CALIBRATION PROCEDURE

NI 5442

This document contains instructions for calibrating the National Instruments PXIe-5442 (NI 5442) arbitrary waveform generator. This calibration procedure is intended for metrology labs. It describes specific programming steps for writing an external calibration procedure for the NI 5442.

Refer to `ni.com/calibration` for additional information about calibration solutions from National Instruments.

Contents

Conventions

The following conventions are used in this manual:

» — The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box.

This icon denotes a note, which alerts you to important information.

This icon denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash. When this symbol is marked on a product, refer to the *Read Me First*: *Safety and Radio-Frequency Interference* for information about precautions to take.

bold — Bold text denotes items that you must select or click in the software, such as menu items and dialog box options. Bold text also denotes parameter names.

italic — Italic text denotes variables, emphasis, a cross-reference, or an introduction to a key concept. Italic text also denotes text that is a placeholder for a word or value that you must supply.

`monospace` — Text in this font denotes text or characters that you should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames, and extensions.

Software Requirements

Calibrating the NI 5442 requires installing NI-FGEN version 2.5 or later on the calibration system. You can download the NI-FGEN instrument driver from the Instrument Driver Network Web site at `ni.com/idnet`. NI-FGEN supports programming a *Self-Calibration* and an *External Calibration* in the LabVIEW, LabWindows™/CVI™, and C or C++ application development environments (ADEs). When you install NI-FGEN, you only need to install support for the ADE that you intend to use.

LabVIEW support is in the `niFgen.llb` file, and all calibration functions appear in the NI-FGEN Calibration palette. For LabWindows/CVI users, the NI-FGEN function panel (`niFgen.fp`) provides access to the available functions.

Calibration functions are LabVIEW VIs or C function calls in NI-FGEN. In this document, the LabVIEW VI or NI-FGEN LabVIEW property is listed in the instructions. Tables for each step show both the configuration of the VI or property and the C function. The C function calls are valid for supported C or C++ compilers. Many of the functions use constants defined in the `niFgen.h` file. To use these constants in C or C++, include `niFgen.h` in your code when you write the calibration procedure.

For the locations of files you may need to calibrate your device, refer to the *NI-FGEN Instrument Driver Readme*, which is available on the NI-FGEN CD.

Note After you install NI-FGEN, you can access the *NI-FGEN Instrument Driver Readme* and other signal generators documentation at **Start»All Programs»National Instruments»NI-FGEN»Documentation**.

Documentation Requirements

For information about NI-FGEN and the NI 5442, you can consult the following documents:

- *NI Signal Generators Getting Started Guide*—provides instructions for installing and configuring NI signal generators.
- *NI Signal Generators Help*—includes detailed information about the NI 5442 and the NI-FGEN VIs and functions.
- *NI 5442 Specifications*—provides the published specification values for the NI 5442.

These documents are installed with NI-FGEN. You also can find the latest versions of the documentation at `ni.com/manuals`.

Password

"`NI`" is the default password for password-protected operations on your device. This password is required to open an external calibration session.

Calibration Interval

A calibration is required only once every two years; however, the measurement accuracy demands of your application determine how often you should perform external calibration. For more information about designing a calibration procedure to suit your needs, refer to *Appendix A: Calibration Procedure Options*.

Test Equipment

External calibration requires different equipment for each applicable specification. Refer to Table 1 for a list of equipment.

Table 1. Equipment Required for Calibrating the NI 5442

Instrument	Applicable Specification	Minimum Specifications	Recommended Instruments
Digital multimeter (DMM)	DC gain and offset	DC accuracy ≤ ±50 ppm Resolution ≤1 μV	NI PXI-4070 Agilent/HP 34401A Keithley 2000
Male banana to female BNC adapter		—	—
Male BNC to female SMB cable		50 Ω, RG-223	—
Spectrum analyzer or frequency meter	Frequency accuracy	Ability to measure 10 MHz or higher sine waves Frequency accuracy to ±500 ppb	NI PXI-5660 Agilent/HP 8560E Agilent/HP 53131A or HP 53132A with timebase option 001, 010, or 012
Male BNC to female SMB cable		50 Ω, RG-223	—

Test Conditions

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

- Keep connections to the NI 5442 short. Long cables and wires act as antennae, picking up noise that can affect measurements.
- Keep relative humidity between 10 and 90% noncondensing.
- Maintain a temperature between 18 and 28 °C.
- Allow a warm-up time of at least 15 minutes after powering on all hardware, loading the operating system, and, if necessary, enabling the device. Unless manually disabled, the NI-FGEN driver automatically loads with the operating system and enables the device. The warm-up time brings the measurement circuitry of the NI 5442 to a stable operating temperature.
- Ensure that the PXI Express chassis fan speed is set to HI, that the fan filters are clean, and that the empty slots contain filler panels.
- Plug the PXI Express chassis and the calibrator into the same power strip to avoid ground loops.

Calibration Procedures

The calibration process has the following steps:

1. *Initial Setup*—Configure the device in Measurement & Automation Explorer (MAX).
2. *Self-Calibration*—Adjust the self-calibration constants of the device.
3. *Verification*—Verify the existing operation of the device. This step allows you to confirm that the device was operating within its specified range prior to adjustment.
4. *Adjustment*—Adjust the device calibration constants with respect to a known voltage source and update the calibration date and temperature.
5. *Reverification*—Repeat the verification process to ensure that the device is operating within its specifications after adjustment.

Note In some cases, the complete calibration procedure may not be required. Refer to *Appendix A: Calibration Procedure Options* for more information about developing a calibration procedure that fits your needs.

Initial Setup

The device must be configured in MAX to communicate with NI-FGEN.

Complete the following steps to configure a device in MAX.

1. Install NI-FGEN.
2. Power off the computer or chassis that will hold the device and install the device in an available slot. Refer to the *NI Signal Generators Getting Started Guide* for information about installing the hardware.
3. Power on the computer or chassis and launch MAX.
4. Configure the device identifier and select **Self-Test** to ensure that the device is working properly.

Note When a device is configured with MAX, it is assigned a device identifier. This device identifier is used to open an NI-FGEN session.

Note For more information about configuring and testing your device in MAX, refer to the *NI Signal Generators Getting Started Guide*.

Self-Calibration

The NI 5442 can perform self-calibration, which adjusts the gain and offset voltage of the analog path. Self-calibration uses an onboard analog-to-digital converter (ADC) to measure the output voltage.

Note You can calibrate the oscillator frequency and the calibration ADC only through an external adjustment procedure.

You can initiate self-calibration interactively from MAX or from the FGEN Soft Front Panel (SFP). Alternatively, you can initiate self-calibration programmatically using NI-FGEN. The following sections include information about performing a self-calibration within each of these environments.

Note Running self-calibration overwrites the analog output constants measured in external calibration, using the onboard ADC to adjust for changes in temperature and other conditions that may have occurred since the external calibration. Calling the niFgen Restore Last Ext Cal Constants VI or the `niFgen_RestoreLastExtCalConstants` function reverts the device to the analog-output constants from the external calibration.

MAX

To initiate self-calibration from MAX, complete the following steps:

1. Launch MAX.
2. Select **My System»Devices and Interfaces»NI-DAQmx Devices**.
3. Select the NI 5442 to calibrate.
4. Initiate self-calibration in one of the following ways:
   - Click **Self-Calibrate** in the upper right corner of the window.
   - Right-click the device name under Devices and Interfaces, and select **Self-Calibrate** from the drop-down listbox.

FGEN Soft Front Panel

To initiate self-calibration from the FGEN SFP, complete the following steps:

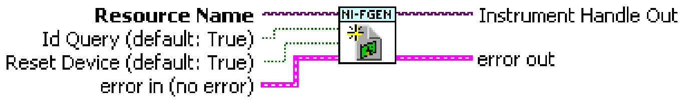
1. Launch the FGEN SFP.
2. Select **Edit»Device Configuration** to launch the Device Configuration dialog box.
3. Select the device that you want to calibrate from the Device drop-down listbox. Click **OK** when finished.
4. Select **Utility»Calibration** to launch the Calibration dialog box.
5. Click **Perform self-calibration**.

NI-FGEN

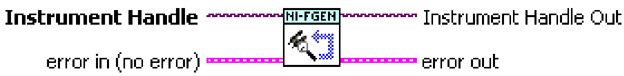
Complete the following steps to programatically perform a self-calibration on the NI 5442 using NI-FGEN:

1. Open an instrument driver session, initialize the device for operation, and return a session handle that will be used to identify the device in future NI-FGEN calls by calling the niFgen Initialize VI.

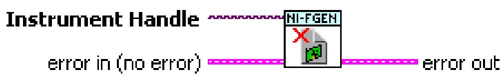
Note Throughout the procedure, refer to C/C++ function call parameters for the LabVIEW input values.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_init` using the following parameters: **resourceName**: The name of the device that you want to verify. This name is the device identifier assigned in MAX. **IDQuery**: `VI_TRUE` **resetDevice**: `VI_TRUE`

2. Initiate self-calibration by calling the niFgen Self Cal VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SelfCal` using the following parameter: **vi**: The session handle returned from `niFgen_init`

3. Close the instrument driver session, destroy the instrument driver session and all of its properties, and release any memory resources NI-FGEN uses by calling the niFgen Close VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_close` using the following parameter: **vi**: The session handle returned from `niFgen_init`

Verification

This section provides instructions for verifying the NI 5442 specifications. Refer to Table 1 for recommendations about choosing an instrument for each test.

Note Always self-calibrate the NI 5442 before beginning a verification procedure.

The steps in the verification procedures describe the code that you use to generate the appropriate signals, as well as the NI-FGEN function calls that you make to verify specifications.

You can verify the following specifications for the NI 5442:

- Oscillator frequency accuracy
- DC gain and offset accuracy

The verification procedure for each of these specifications includes setting up, programming, and closing the application.

Note If any of these tests fail immediately after you perform an external adjustment, verify that you have met the required test conditions before you return the NI 5442 to NI for repair.

Refer to Figure 1 for the names and locations of the NI PXIe-5442 front panel connectors. You can find information about the functions of these connectors in the *NI Signal Generators Getting Started Guide*.

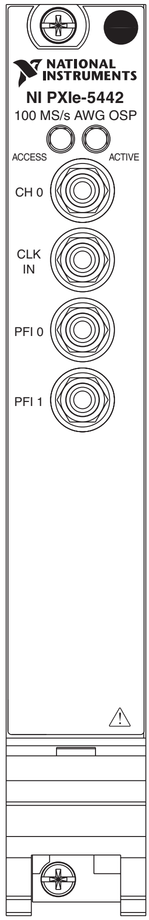


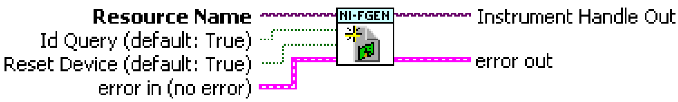
Figure 1. NI PXIe-5442 Front Panel Connectors

Verifying the Oscillator Frequency Accuracy

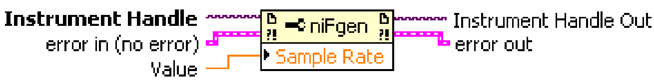
This test verifies the frequency accuracy of the oscillator on the NI 5442. The verification involves generating a 10 MHz sine wave with the NI 5442 and measuring the sine wave frequency with one of the instruments from Table 1.

To verify the frequency accuracy of the oscillator on the NI 5442, complete the following steps:

1. Connect the NI 5442 CH 0 front panel connector to the instrument measuring the frequency accuracy with a male BNC to female SMB cable.
2. Open an instrument driver session, initialize the device for operation, and return a session handle that will be used to identify the device in future NI-FGEN calls by calling the niFgen Initialize VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_init` using the following parameters: **resourceName**: The name of the device that you want to verify. This name is the device identifier assigned in MAX. **IDQuery**: `VI_TRUE` **resetDevice**: `VI_TRUE`

3. Set the sample rate by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Sample Rate**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ SetAttribute ViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_ARB_SAMPLE_ RATE` **value**: `100,000,000`

4. Set the arbitrary waveform gain by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Gain**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ SetAttribute ViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_ARB_GAIN` **value**: `1`

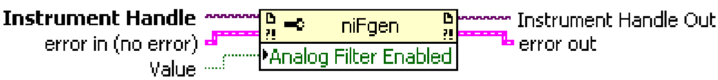
Note You can adjust the gain value based on which measuring device you use.

5. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

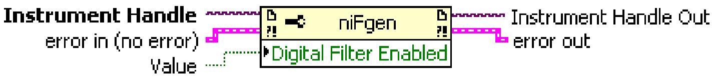
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_OFFSET` **value**: `0`

Note You can adjust the offset value based on which measuring device you use.

6. Enable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_TRUE`

7. Enable the digital filter state by calling the niFgen Property Node and selecting **Output Attributes»Digital Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
 	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_DIGITAL_FILTER_ENABLED` **value**: `VI_TRUE`

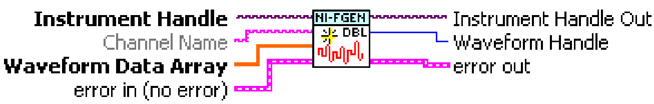
8. Set the digital filter interpolation factor by calling the niFgen Property Node and selecting **Output Attributes»Digital Filter Interpolation Factor**.

LabVIEW Block Diagram	C/C++ Function Call
 	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_DIGITAL_FILTER_INTERPOLATION_FACTOR` **value**: `4`

9. Generate an array of waveform samples. The waveform should have 10 samples per cycle with a total of 500 samples and 50 sine wave cycles. The 100 MS/s sample rate with 10 samples per cycle results in a 10 MHz sine wave waveform.

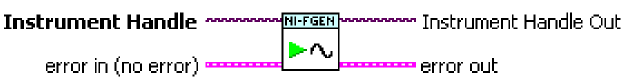
Note The sample values in this waveform must fall between –1.0 and 1.0 (representation: double).

10. Create an arbitrary waveform by calling the niFgen Create Waveform (DBL) instance of the niFgen Create Waveform (poly) VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_CreateWaveformF64</code> using the following parameters:

vi: The session handle returned from <code>niFgen_init</code>
channelName: <code>"0"</code>
numberOfSamples: The size in samples (<code>500</code>) of the waveform you created in step 9
wfmData[]: The array of waveform samples that you created in step 9</td></tr>
</table>

11. Initiate waveform generation by calling the niFgen Initiate Generation VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_InitiateGeneration</code> using the following parameter:

vi: The session handle returned from <code>niFgen_init</code></td></tr>
</table>

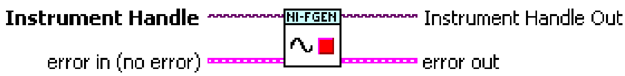
12. Measure the frequency of the NI 5442 output signal.

    A frequency error of 45 Hz for a 10 MHz signal corresponds to an error of 4.5 ppm. This limit accounts for the initial accuracy and the frequency deviation caused by temperature and aging. Refer to Table 2 for frequency ranges.

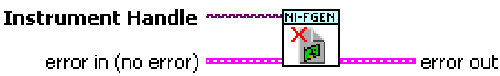
Table 2. Frequency Ranges

Calibration Test Limit		Published Specifications ±25 ppm	
Low (Hz)	**High (Hz)**	**Low (Hz)**	**High (Hz)**
9,999,955	10,000,045	9,999,750	10,000,250

13. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_AbortGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

14. Close the instrument driver session, destroy the instrument driver session and all of its properties, and release any memory resources NI-FGEN uses by calling the niFgen Close VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_close` using the following parameter: **vi**: The session handle returned from `niFgen_init`

Verifying the DC Gain and Offset Accuracy

This test verifies the DC gain and offset accuracy of the NI 5442 into a high-impedance load by generating a number of DC voltages and offsets, measuring the voltages with a DMM, and comparing the NI 5442 results to the error limits.

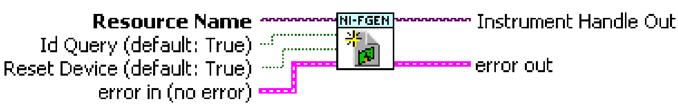
The DC gain and offset accuracy verification procedure has three subprocedures that verify the following specifications:

- Main analog path gain
- Main analog path offset
- Direct path gain

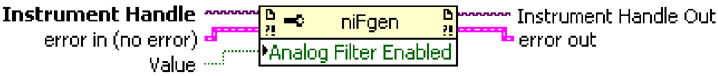
Verifying the Main Analog Path Gain

To verify the gain of the NI 5442 Main analog path, complete the following steps:

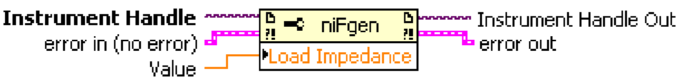
1. Connect the NI 5442 CH 0 front panel connector to a DMM to measure DC gain and offset accuracy.
2. Open an instrument driver session, initialize the device for operation, and return a session handle that will be used to identify the device in future NI-FGEN calls by calling the niFgen Initialize VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_init` using the following parameters: **resourceName**: The name of the device that you want to verify. This name is the device identifier assigned in MAX. **IDQuery**: `VI_TRUE` **resetDevice**: `VI_TRUE`

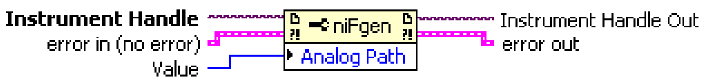
3. Disable the analog filter state by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_FALSE`

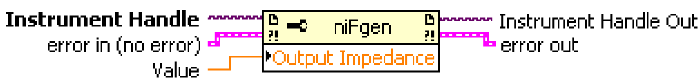
4. Set the load impedance by calling the niFgen Property Node and selecting **Output Attributes»Load Impedance**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_LOAD_IMPEDANCE` **value**: `10,000,000,000`

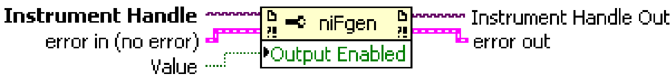
5. Select the Main analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_MAIN_ANALOG_PATH`

6. Set the output impedance by calling the niFgen Property Node and selecting **Basic Operation»Output Impedance**.

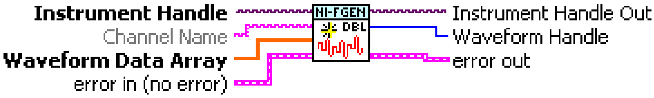
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE` **value**: `50.00`

7. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_TRUE`

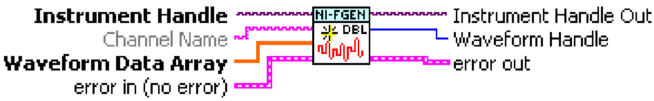
8. Create an array of waveform samples for the positive full-scale DC waveform. This array should contain 500 samples with each sample having the value `1.0` (representation: double).

9. Create an arbitrary waveform by calling the niFgen Create Waveform (DBL) instance of the niFgen Create Arbitrary Waveform (poly) VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_CreateWaveformF64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **numberOfSamples**: The size in samples (`500`) of the waveform you created in step 8 **wfmData[]**: The array of waveform samples that you created in step 8

10. Create an array of waveform samples for the negative full-scale DC waveform. This array should contain 500 samples with each sample having the value `-1.0` (representation: double).

11. Create an arbitrary waveform by calling the niFgen Create Waveform (DBL) instance of the niFgen Create Arbitrary Waveform (poly) VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_CreateWaveformF64</code> using the following parameters:

vi: The session handle returned from <code>niFgen_init</code>
channelName: <code>"0"</code>
numberOfSamples: The size in samples (<code>500</code>) of the waveform that you created in step 10
wfmData[]: The array of waveform samples that you created in step 10</td></tr>
</table>

12. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_SetAttributeViReal64</code> using the following parameters:

vi: The session handle returned from <code>niFgen_init</code>
channelName: <code>"0"</code>
attributeId: <code>NIFGEN_ATTR_ARB_OFFSET</code>
value: <code>0</code></td></tr>
</table>

Steps 13 through 23 use the values listed in Table 3. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

Table 3. Values for Verifying the Gain of the Main Analog Path

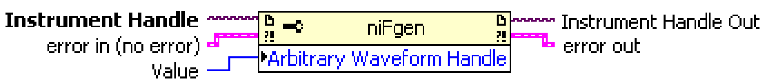
Iteration	Gain	Ideal Positive Full-Scale (Volts)	Ideal Negative Full-Scale (Volts)	Calibration Test Limit (Volts)	Published Specification (Volts)
1	2.000000	2.000000	–2.000000	±0.003700	±0.008500
2	1.650000	1.650000	–1.650000	±0.003140	±0.007100
3	1.250000	1.250000	–1.250000	±0.002500	±0.005500
4	0.850000	0.850000	–0.850000	±0.001860	±0.003900
5	0.600000	0.600000	–0.600000	±0.001460	±0.002900
6	0.415000	0.415000	–0.415000	±0.001164	±0.002160
7	0.300000	0.300000	–0.300000	±0.000980	±0.001700
8	0.205000	0.205000	–0.205000	±0.000828	±0.001320
9	0.150000	0.150000	–0.150000	±0.000740	±0.001100
10	0.105000	0.105000	–0.105000	±0.000668	±0.000920
11	0.075000	0.075000	–0.075000	±0.000620	±0.00080
12	0.055000	0.055000	–0.055000	±0.000588	±0.000720
13	0.037500	0.037500	–0.037500	±0.000560	±0.000650
14	0.026000	0.026000	–0.026000	±0.000542	±0.000604
15	0.018500	0.018500	– 0.018500	±0.000530	±0.000574
16	0.013000	0.013000	– 0.013000	±0.000521	±0.000552
17	0.009000	0.009000	– 0.009000	±0.000514	±0.000536
18	0.006500	0.006500	– 0.006500	±0.000510	±0.000526

Notes: *Error Positive Full-Scale Value* = (*Measured Positive Full-Scale Value*) – (*Ideal Positive Full-Scale Value*)
Error Negative Full-Scale Value = (*Measured Negative Full-Scale Value*) – (*Ideal Negative Full-Scale Value*)

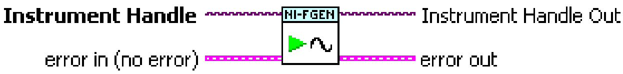
13. Set the arbitrary waveform gain by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Gain**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_GAIN` **value**: The *Gain* value listed in Table 3 for the current iteration

14. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 9 (positive full-scale handle)

15. Initiate waveform generation by calling the niFgen Initiate Generation VI.

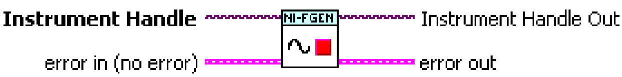
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Initiate Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

16. Measure the NI 5442 DC output voltage. This voltage is the *Measured Positive Full-Scale Value*.
17. Determine the error for positive full scale using the following formula:

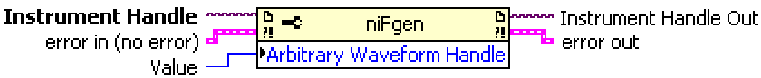
Error Positive Full-Scale =
(*Measured Positive Full-Scale Value*) – (*Ideal Positive Full-Scale Value*)

Compare this error to the *Published Specification* or the *Calibration Test Limit* for the current iteration from Table 3. Refer to *Appendix A: Calibration Procedure Options* for information about the uses of the published specifications and the calibration test limits.

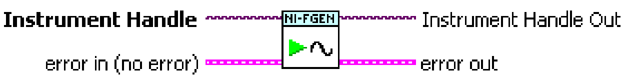
18. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Abort Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

19. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 11 (negative full-scale handle)

20. Initiate waveform generation by calling the niFgen Initiate Generation VI.

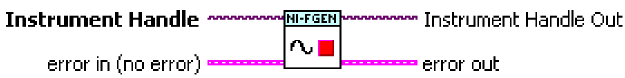
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitiateGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

21. Measure the NI 5442 DC output voltage. This voltage is the *Measured Negative Full-Scale Value*.
22. Determine the error for negative full scale using the following formula:

Error Negative Full-Scale =
(*Measured Negative Full-Scale Value*) – (*Ideal Negative Full-Scale Value*)

Compare this error to the *Published Specification* or the *Calibration Test Limit* for the current iteration from Table 3. Refer to *Appendix A: Calibration Procedure Options* for information about the uses of the published specifications and the calibration test limits.

23. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Abort Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

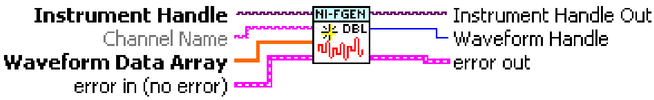
24. If any of the errors are greater than the *Calibration Test Limit*, perform an external adjustment.

Verifying the Main Analog Path Offset

To continue the verification of the DC Gain and Offset accuracy, verify the offset of the NI 5442 Main analog path by completing the following steps:

1. Create an array of waveform samples for the mid-scale DC waveform (0 VDC). This array should contain 500 samples with each sample having the value `0.0` (representation: double).

2. Create an arbitrary waveform by calling the niFgen Create Waveform (DBL) instance of the niFgen Create Waveform (poly) VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_CreateWaveformF64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **numberOfSamples**: The size in samples (`500`) of the waveform that you created in step 1 **wfmData[]**: The array of waveform samples that you created in step 1

3. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 2 (mid-scale handle)

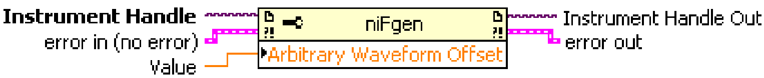
Steps 4 through 14 use the values listed in Table 4. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

Table 4. Values for Verifying the Offset of the Main Analog Path

Iteration	Gain	Ideal Positive Offset (Volts)	Ideal Negative Offset (Volts)	Calibration Test Limit (Volts)	Published Specification (Volts)
1	2.000000	1.000000	–1.000000	±0.004000	±0.009000
2	1.650000	0.825000	–0.825000	±0.003388	±0.007513
3	1.250000	0.625000	–0.625000	±0.002688	±0.005813
4	0.850000	0.425000	–0.425000	±0.001988	±0.004113
5	0.600000	0.300000	–0.300000	±0.001550	±0.003050
6	0.415000	0.207500	–0.207500	±0.001226	±0.002264
7	0.300000	0.150000	–0.150000	±0.001025	±0.001775
8	0.205000	0.102500	–0.102500	±0.000859	±0.001371
9	0.150000	0.075000	–0.075000	±0.000763	±0.001138
10	0.105000	0.052500	–0.052500	±0.000684	±0.000946
11	0.075000	0.037500	–0.037500	±0.000631	±0.000819
12	0.055000	0.027500	–0.027500	±0.000596	±0.000734
13	0.037500	0.018750	–0.018750	±0.000566	±0.000659
14	0.026000	0.013000	–0.013000	±0.000546	±0.000611
15	0.018500	0.009250	–0.009250	±0.000532	±0.000579
16	0.013000	0.006500	–0.006500	±0.000523	±0.000555
17	0.009000	0.004500	–0.004500	±0.000516	±0.000538
18	0.006500	0.003250	–0.003250	±0.000511	±0.000528

Notes: *Error Positive Offset Value* = (*Measured Positive Offset Value*) – (*Ideal Positive Offset Value*)
Error Negative Offset Value = (*Measured Negative Offset Value*) – (*Ideal Negative Offset Value*)

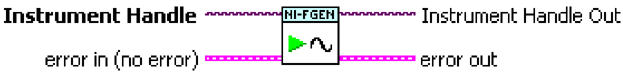
4. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_OFFSET` **value**: The *Ideal Positive Offset* value listed in Table 4 for the current iteration

5. Set the arbitrary waveform gain by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Gain**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_GAIN` **value**: The *Gain* value listed in Table 4 for the current iteration

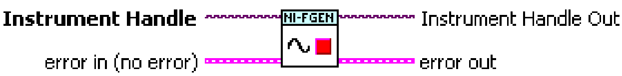
6. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitiateGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

7. Measure the positive NI 5442 DC output voltage. This value is the *Measured Positive Offset Value*.
8. Determine the error for positive offset using the following formula:

   Error Positive Offset =
   (*Measured Positive Offset Value*) – (*Ideal Positive Offset Value*)

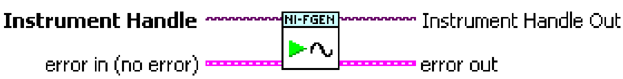
   Compare this error to the *Published Specification* or the *Calibration Test Limit* for the current iteration from Table 4. Refer to *Appendix A: Calibration Procedure Options* for information about the uses of the published specifications and the calibration test limits.
9. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_AbortGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

10. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_OFFSET` **value**: The *Ideal Negative Offset* value listed in Table 4 for the current iteration

11. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitiateGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

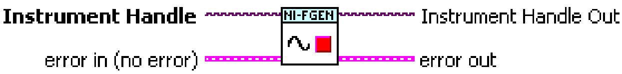
12. Measure the negative NI 5442 DC output voltage. This value is the *Measured Negative Offset Value*.
13. Determine the error for negative offset using the following formula:

$$\textit{Error Negative Offset} = (\textit{Measured Negative Offset Value}) - (\textit{Ideal Negative Offset Value})$$

Compare this error to the *Published Specification* or the *Calibration Test Limit* for the current iteration from Table 4. Refer to Appendix A:

Calibration Procedure Options for information about the uses of the published specifications and the calibration test limits.

14. Abort waveform generation by calling the niFgen Abort Generation VI.

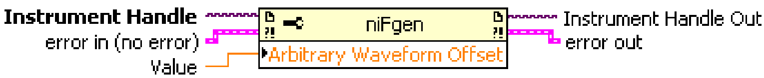
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Abort Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

15. If any of the errors are greater than the *Calibration Test Limit*, perform an external adjustment.

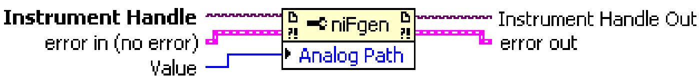
Verifying the Gain of the Direct Path

To continue the verification of the DC Gain and Offset accuracy, verify the gain of the NI 5442 Direct path by completing the following steps:

1. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ SetAttribute ViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_ARB_OFFSET` **value**: `0`

2. Select the Direct analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_DIRECT_ANALOG_PATH`

Steps 3 through 14 use the values listed in Table 5. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

Table 5. Values for Verifying the Gain of the Direct Analog Path

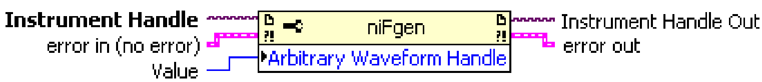
Iteration	Gain	Ideal Positive Full-Scale (Volts)	Ideal Negative Full-Scale (Volts)	Offset Limit (Volts)	Calibration Test Limit (Volts)	Published Specification (Volts)
1	1.000000	1.000000	–1.000000	±0.025000	±0.001600	±0.004000
2	0.950000	0.950000	–0.950000	±0.025000	±0.001520	±0.003800
3	0.900000	0.900000	–0.900000	±0.025000	±0.001440	±0.003600
4	0.850000	0.850000	–0.850000	±0.025000	±0.001360	±0.003400
5	0.800000	0.800000	–0.800000	±0.025000	±0.001280	±0.003200
6	0.750000	0.750000	–0.750000	±0.025000	±0.001200	±0.003000
7	0.710000	0.710000	–0.710000	±0.025000	±0.001136	±0.002840

Notes: *Offset* = ((*Measured Positive Full-Scale Value*) + (*Measured Negative Full-Scale Value*))/2
Error Positive Full-Scale Value = (*Measured Positive Full-Scale Value*) – *Offset* – (*Ideal Positive Full-Scale Value*)
Error Negative Full-Scale Value = (*Measured Negative Full-Scale Value*) – *Offset* – (*Ideal Negative Full-Scale Value*)

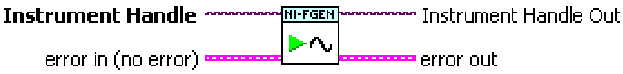
3. Set the arbitrary waveform gain by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Gain**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_GAIN` **value**: The *Gain* value listed in Table 5 for the current iteration

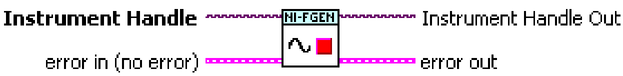
4. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 9 of the *Verifying the Main Analog Path Gain* section (positive full-scale handle)

5. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Initiate Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

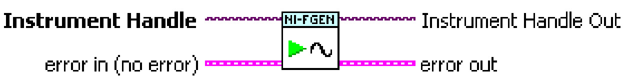
6. Measure the NI 5442 positive DC output voltage. This value is the *Measured Positive Full-Scale Value*.
7. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Abort Generation` using the following parameter: **vi**: The session handle returned from `niFgen_init`

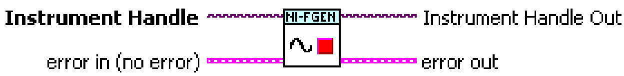
8. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_init` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 11 of the *Verifying the Main Analog Path Gain* section (negative full-scale handle)

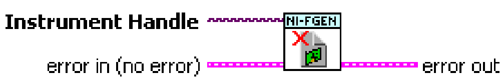
9. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitiateGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

10. Measure the NI 5442 negative DC output voltage. This value is the *Measured Negative Full-Scale Value*.
11. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_AbortGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_init`

12. Average the *Measured Positive Full-Scale Value* and *Measured Negative Full-Scale Value* to calculate the *Offset*.
13. Verify that the *Offset* is less than or equal to the *Offset Limit* listed in Table 5 for the current iteration.
14. Subtract the *Offset* and the *Ideal Full-Scale Value* from the *Measured Full-Scale Value* to get the *Error Full-Scale Value* for both the positive and negative settings, respectively.
15. If any of the errors are greater than the *Calibration Test Limit* listed in Table 5, perform an external adjustment.
16. Close the instrument driver session, destroy the instrument driver session and all of its properties, and release any memory resources NI-FGEN uses by calling the niFgen Close VI.

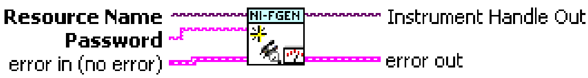
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_close` using the following parameter: **vi**: The session handle returned from `niFgen_init`

Note The offset is not adjustable for the Direct path.

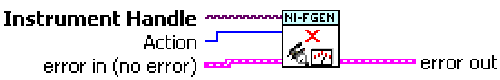
Updating the Calibration Date and Temperature

If the NI 5442 passes verification within the calibration test limits and you do not want to perform an adjustment, you can update the calibration date and onboard calibration temperature without making any adjustments by completing the following steps:

1. Open an NI-FGEN external calibration session by calling the niFgen Init Ext Cal VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Init ExtCal` using the following parameters: **resourceName**: The name of the device that you want to verify. This name is the device identifier assigned in MAX. **password**: `"NI"` (default)

2. Close the instrument driver session and save the calibration date and temperature to the onboard EEPROM by calling the niFgen Close Ext Cal VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Close ExtCal` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **action**: `NIFGEN_VAL_ EXT_CAL_COMMIT`. This setting stores the date and temperature of the system at the time of calibration.

Adjustment

If the NI 5442 successfully passes all verification procedures within the calibration test limits, NI nevertheless recommends adjustment to guarantee its published specifications for the next two years. If you choose not to adjust your device, refer to the *Updating the Calibration Date and Temperature* section for instructions for updating the calibration date and temperature without performing an adjustment. If the NI 5442 is not within the calibration test limits for each verification procedure, perform the adjustment procedure to improve the accuracy of the NI 5442. Refer to *Appendix A: Calibration Procedure Options* for more information about which procedures to perform.

The external calibration procedure adjusts the analog output, the oscillator frequency, and the onboard calibration ADC. *Analog output adjustment* characterizes the DC gains and the offsets of the analog path to ensure the analog voltage accuracy. *Oscillator frequency adjustment* characterizes the onboard oscillator to ensure frequency accuracy. *Calibration ADC adjustment* characterizes the onboard ADC gain and offset so that self-calibration results in an accurately calibrated device.

To perform an adjustment, create an external calibration session by calling the niFgen Init Ext Cal VI or the `nFgen_InitExtCal` function. The external calibration session adjusts a set of calibration constants that are determined during the calibration procedure and stored in the device onboard memory when the session is closed. NI-FGEN uses these calibration constants during a standard NI-FGEN session to ensure that the device operates within its specifications. You must close an external calibration session by calling the niFgen Close Ext Cal VI or the `niFgen_CloseExtCal` function.

The following figure shows the programming flow for an external calibration procedure.

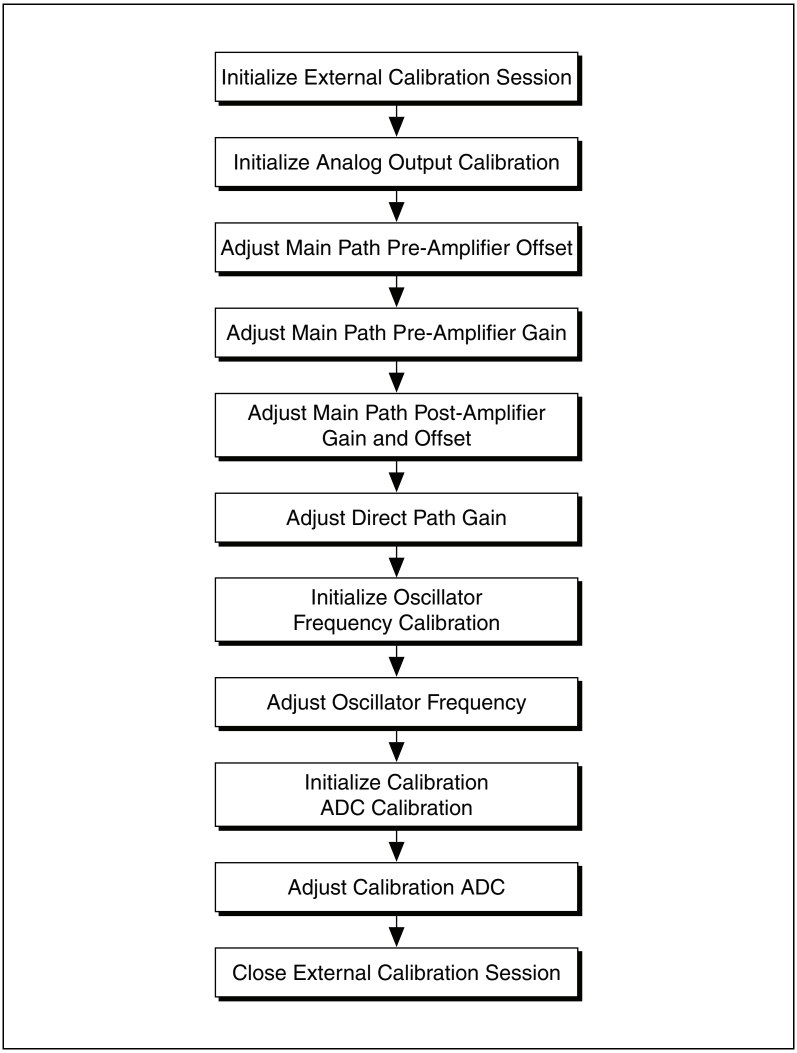
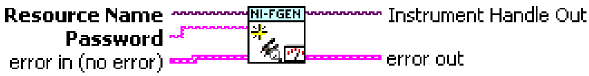


Figure 2. NI 5442 External Calibration Procedure Programming Flow

Initializing the External Calibration Session

Open an NI-FGEN external calibration session by calling the niFgen Init Ext Cal VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitExtCal` using the following parameters: **resourceName**: The name of the device that you want to verify. This name is the device identifier assigned in MAX. **password**: `"NI"` (default)

Adjusting the Analog Output

The analog output adjustment procedure has several subprocedures that adjust the following characteristics:

- Main path preamplifier offset
- Main path preamplifier gain
- Main path postamplifier gain and offset
- Direct path gain

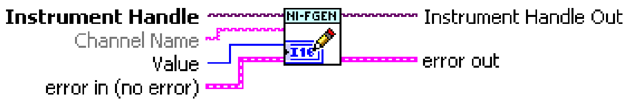
In each of these sub-procedures, the device is in several configurations and takes several output measurements. You then pass these measurements to NI-FGEN to determine the calibration constants for the device.

Initializing Analog Output Calibration

1. Initialize analog output calibration by calling the niFgen Initialize Analog Output Calibration VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitializeAnalogOutputCalibration` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

2. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `0`

3. Select the fixed Low-Gain analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH`

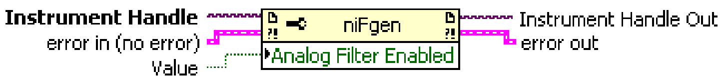
4. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,000`

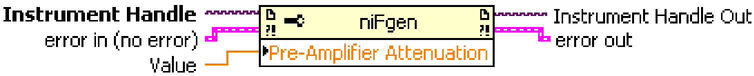
5. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	**C/C++ Function Call**
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `32,767`

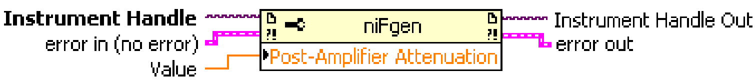
6. Disable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	**C/C++ Function Call**
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_FALSE`

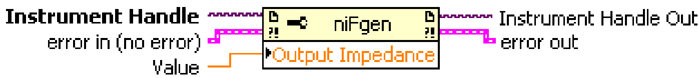
7. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: `0`

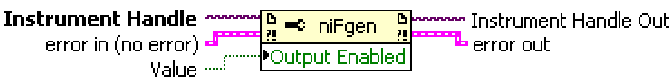
8. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION` **value**: `0`

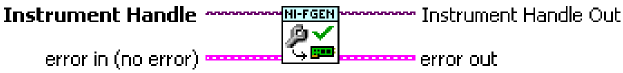
9. Set the output impedance by calling the niFgen Property Node and selecting **Basic Operation»Output Impedance**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE` **value**: `50.00`

10. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

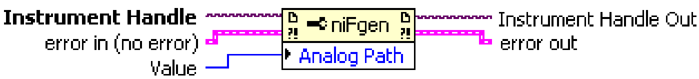
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_TRUE`

11. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

Adjusting the Main Path Preamplifier Offset

1. Select the fixed Low-Gain analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

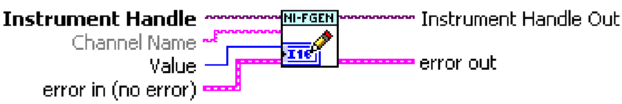
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH`

2. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_SetAttributeViReal64</code> using the following parameters:

vi: The session handle returned from <code>niFgen_InitExtCal</code>
channelName: <code>"0"</code>
attributeId: <code>NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION</code>
value: <code>0</code></td></tr>
</table>

3. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_WriteBinary16AnalogStaticValue</code> using the following parameters:

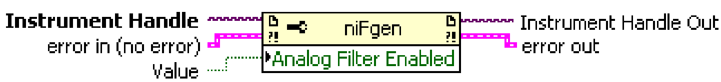
vi: The session handle returned from <code>niFgen_InitExtCal</code>
channelName: <code>"0"</code>
value: <code>0</code></td></tr>
</table>

Steps 4 through 7 use the values listed in Table 6. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

Table 6. Attributes and Values for Main Path Preamplifier Offset

Iteration	Analog Filter Enable	Preamplifier Attenuation	Current Configuration
1	`VI_FALSE`	0	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_0DB`
2	`VI_FALSE`	3	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_3DB`
3	`VI_FALSE`	6	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_6DB`
4	`VI_FALSE`	9	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_9DB`
5	`VI_FALSE`	12	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_12DB`
6	`VI_TRUE`	0	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_0DB`
7	`VI_TRUE`	3	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_3DB`
8	`VI_TRUE`	6	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_6DB`
9	`VI_TRUE`	9	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_9DB`
10	`VI_TRUE`	12	`NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_12DB`

4. Set the analog filter state by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: The *Analog Filter Enable* value for the current iteration from Table 6

5. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: The *Preamplifier Attenuation* value for the current iteration from Table 6

6. Complete the following steps to take voltage measurements at the NI 5442 CH 0 front panel connector into a high-impedance load:

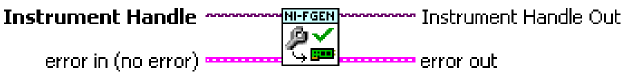
    a. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,000`

b. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `50,000`

c. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

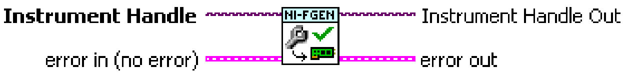
d. Wait 500 ms for the output to settle.

e. Use the DMM to measure the device output voltage. This measurement is `measurement 0`, which is used in step 7.

f. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `1,000`

g. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

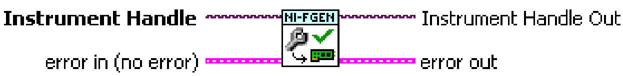
h. Wait 500 ms for the output to settle.

i. Use the DMM to measure the device output voltage. This measurement is `measurement 1`, which is used in step 7.

j. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `15,000`

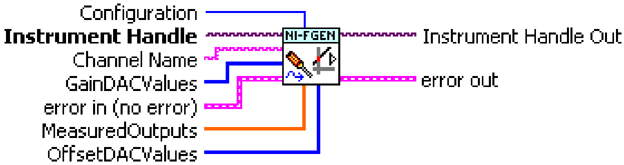
k. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

l. Wait 500 ms for the output to settle.

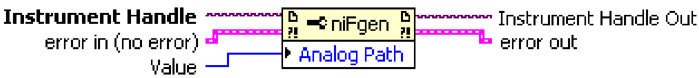
m. Use the DMM to measure the device output voltage. This measurement is `measurement 2`, which is used in step 7.

7. Adjust the preamplifier Main path offset by calling the niFgen Cal Adjust Main Path Pre Amp Offset VI.

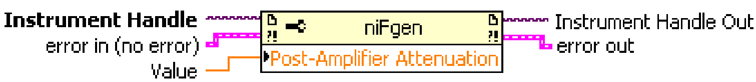
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_CalAdjustMainPathPreAmpOffset` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **configuration**: The *Current Configuration* value for the current iteration from Table 6 **gainDACValues**: `{2000, 1000}` **offsetDACValues**: `{50000, 15000}` **measuredOutputs**: `{measurement 0, measurement 1, measurement 2}`

Adjusting the Main Path Preamplifier Gain

1. Select the fixed Low-Gain analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH`

2. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION` **value**: `0`

3. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

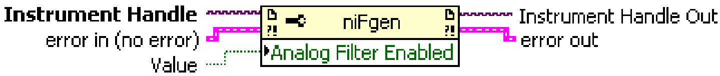
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `32,000`

Steps 4 through 7 use the values listed in Table 7. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

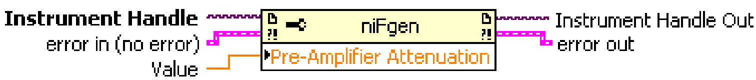
Table 7. Attributes and Values for Main Path Preamplifier Gain

Iteration	Analog Filter Enable	Preamplifier Attenuation	Current Configuration
1	VI_FALSE	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_0DB
2	VI_FALSE	3	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_3DB
3	VI_FALSE	6	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_6DB
4	VI_FALSE	9	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_9DB
5	VI_FALSE	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_12DB
6	VI_TRUE	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_0DB
7	VI_TRUE	3	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_3DB
8	VI_TRUE	6	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_6DB
9	VI_TRUE	9	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_9DB
10	VI_TRUE	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_ON_12DB

4. Set the analog filter state by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: The *Analog Filter Enable* value for the current iteration from Table 7

5. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

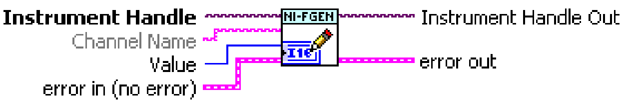
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: The *Preamplifier Attenuation* value for the current iteration from Table 7

6. Complete the following steps to take voltage measurements at the NI 5442 CH 0 front panel connector into a high-impedance load:

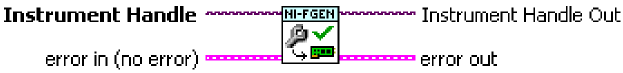
   a. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `1,500`

   b. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `25,233`

c. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

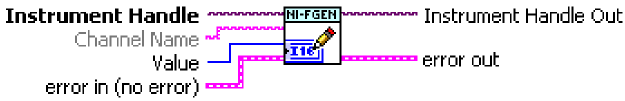
d. Wait 500 ms for the output to settle.

e. Use the DMM to measure the device output voltage. This measurement is `measurement 0`, which is used in step 7.

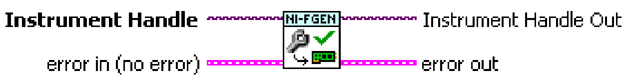
f. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,000`

g. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `-29,232`

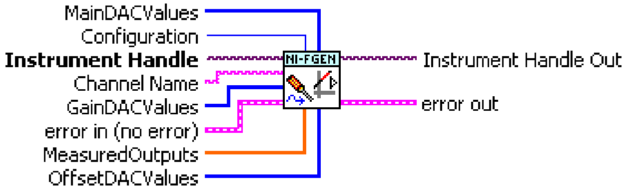
h. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

i. Wait 500 ms for the output to settle.

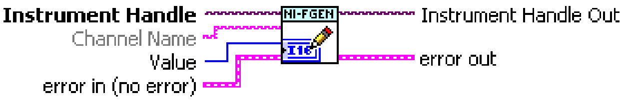
j. Use the DMM to measure the voltage output by the device. This measurement is `measurement 1`, which is used in step 7.

7. Adjust the preamplifier Main path gain and offset by calling the niFgen Cal Adjust Main Path Pre Amp Gain VI.

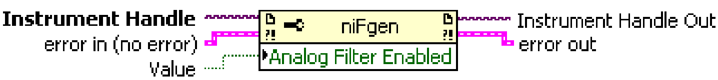
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Cal AdjustMainPath PreAmpGain` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **configuration**: The *Current Configuration* value for the current iteration from Table 7 **mainDACValues**: `{25233, –29232}` **gainDACValues**: `{1500, 2000}` **offsetDACValues**: `{32000}` **measuredOutputs**: `{measurement 0, measurement 1}`

Adjusting the Main Path Postamplifier Gain and Offset

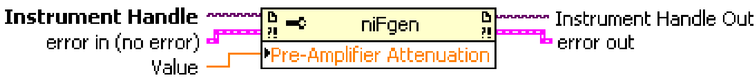
1. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `0`

2. Disable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_FALSE`

3. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: `0`

4. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,000`

5. Select the analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

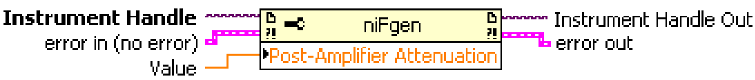
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH`

Steps 6 through 8 use the values listed in Table 8. Complete these steps for the row corresponding to Iteration 1, then repeat them for each of the remaining iterations.

Table 8. Attributes and Values for the Main Path Postamplifier Gain and Offset

Iteration	Postamplifier Attenuation	Current Configuration
1	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_0DB
2	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_12DB
3	24	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_24DB
4	36	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_36DB

6. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

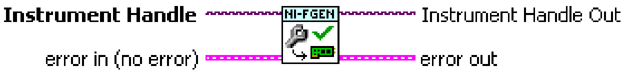
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION` **value**: The *Postamplifier Attenuation* value for the current iteration from Table 8

7. Complete the following steps to take voltage measurements at the NI 5442 CH 0 front panel connector into a high-impedance load:

   a. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `50,000`

b. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

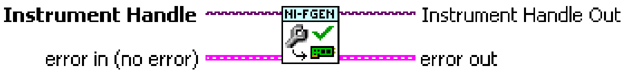
c. Wait 500 ms for the output to settle.

d. Use the DMM to measure the device output voltage. This measurement is `measurement 0`, which is used in step 8.

e. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `15,000`

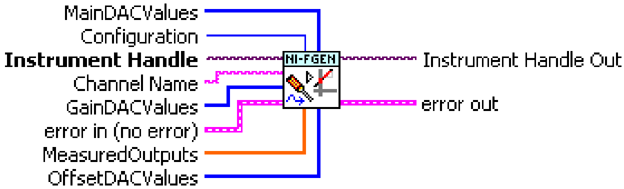
f. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

g. Wait 500 ms for the output to settle.

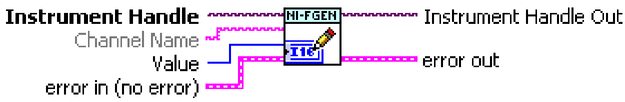
h. Use the DMM to measure the device output voltage. This measurement is `measurement 1`, which is used in step 8.

8. Adjust the postamplifier Main path gain and offset by calling the niFgen Cal Adjust Main Path Post Amp Gain And Offset VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Cal AdjustMainPath PostAmpGainAnd Offset` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **configuration**: The *Current Configuration* value for the current iteration from Table 8 **mainDACValues**: `{0, 0}` **gainDACValues**: `{2000}` **offsetDACValues**: `{50000, 15000}` **measuredOutputs**: `{measurement 0, measurement 1}`

Adjusting the Direct Path Gain

1. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `0`

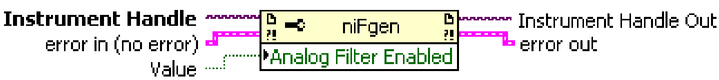
2. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,000`

3. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `32,767`

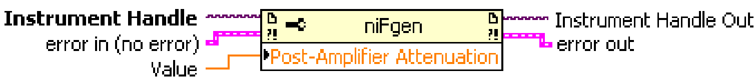
4. Disable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_FALSE`

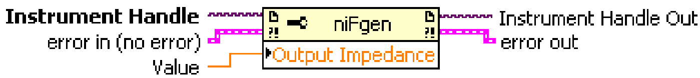
5. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: `0`

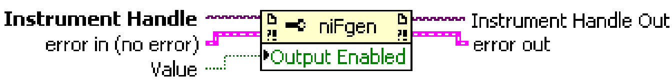
6. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION` **value**: `0`

7. Set the output impedance by calling the niFgen Property Node and selecting **Basic Operation»Output Impedance**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE` **value**: `50.00`

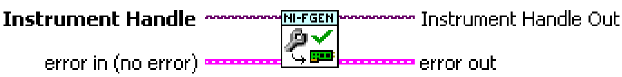
8. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_TRUE`

9. Select the Direct analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

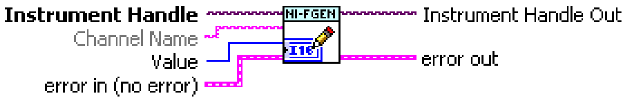
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_DIRECT_ANALOG_PATH`

10. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

11. Complete the following steps to take voltage measurements at the NI 5442 CH 0 front panel connector into a high-impedance load:

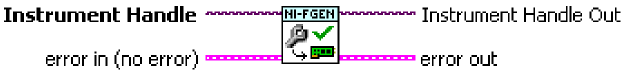
    a. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `32,767`

    b. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `1,800`

c. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

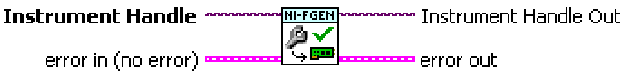
d. Wait 500 ms for the output to settle.

e. Use the DMM to measure the device output voltage. This measurement is `measurement 0`, which is used in step 12.

f. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Set AttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_GAIN_DAC_VALUE` **value**: `2,600`

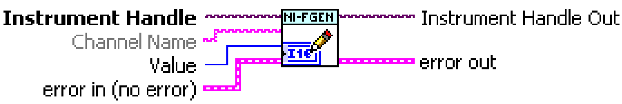
g. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

h. Wait 500 ms for the output to settle.

i. Use the DMM to measure the device output voltage. This measurement is `measurement 1`, which is used in step 12.

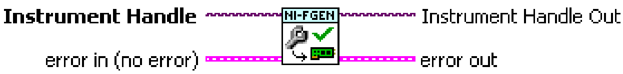
j. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `-32,767`

k. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `1,500`

l. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

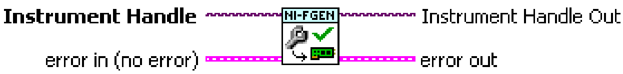
m. Wait 500 ms for the output to settle.

n. Use the DMM to measure the device output voltage. This measurement is `measurement 2`, which is used in step 12.

o. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `2,300`

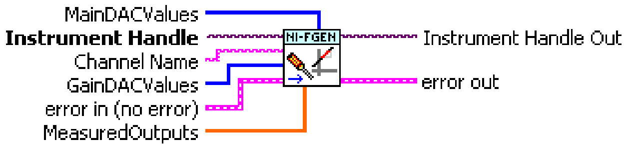
p. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

q. Wait 500 ms for the output to settle.

r. Use the DMM to measure the device output voltage. This measurement is `measurement 3`, which is used in step 12.

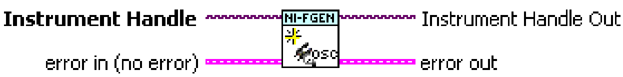
12. Adjust the Direct path gain by calling the niFgen Cal Adjust Direct Path Gain VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ CalAdjustDirectPath Gain` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **configuration**: `{32767, -32767}` **gainDACValues**: `{1800, 2600, 1500, 2300}` **offsetDACValues**: `{50000, 15000}` **measuredOutputs**: `{measurement 0, measurement 1, measurement 2, measurement 3}`

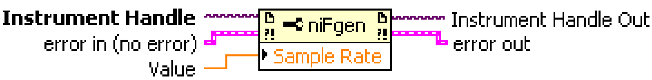
Adjusting the Oscillator Frequency

Adjusting the oscillator frequency involves generating a sine wave at a desired frequency, and then iteratively measuring the frequency, passing the measured value to NI-FGEN so that the oscillator can be adjusted, and then remeasuring the resulting frequency. This process is repeated until the difference between the desired and measured frequency falls within the desired 4.5 ppm tolerance. The adjustment ensures the frequency accuracy of the onboard oscillator.

1. Initialize oscillator frequency calibration by calling the niFgen Initialize Oscillator Frequency Calibration VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ InitializeOscillatorFrequencyCalibration` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

2. Set the sample rate by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Sample Rate**.

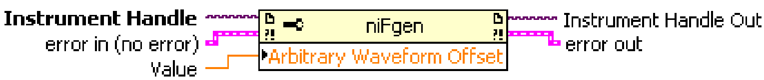
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_ SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_ARB_SAMPLE_ RATE` **value**: `100,000,000`

3. Set the arbitrary waveform gain by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Gain**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_GAIN` **value**: `1`

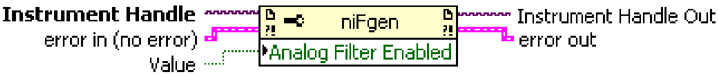
Note You can adjust this value based on which measurement device you use.

4. Set the arbitrary waveform offset by calling the niFgen Property Node and selecting **Arbitrary Waveform Output»Arbitrary Waveform Offset**.

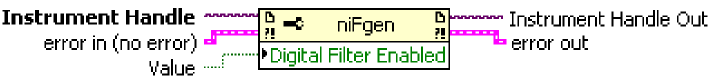
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_OFFSET` **value**: `0`

Note You can adjust this value based on which measurement device you use.

5. Enable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_TRUE`

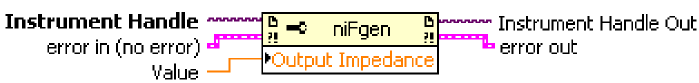
6. Enable the digital filter by calling the niFgen Property Node and selecting **Output Attributes»Digital Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_DIGITAL_FILTER_ENABLED` **value**: `VI_TRUE`

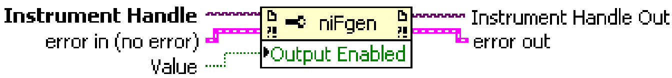
7. Set the digital filter interpolation factor by calling the niFgen Property Node and selecting **Output Attributes»Digital Filter Interpolation Factor**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_DIGITAL_FILTER_INTERPOLATION_FACTOR` **value**: `4`

8. Set the output impedance by calling the niFgen Property Node and selecting **Basic Operation»Output Impedance**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE` **value**: `50.00`

9. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_SetAttributeViBoolean</code> using the following parameters:

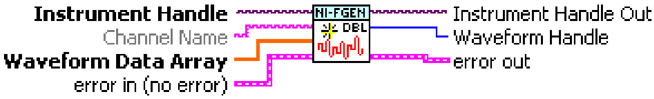
vi: The session handle returned from <code>niFgen_InitExtCal</code>
channelName: <code>"0"</code>
attributeId: <code>NIFGEN_ATTR_OUTPUT_ENABLED</code>
value: <code>VI_TRUE</code></td></tr>
</table>

10. Generate an array of waveform samples.

    Each waveform should have 10 samples per cycle, with a total of 500 samples and 50 sine wave cycles. The 100 MS/s sample rate with 10 samples per cycle results in a 10 MHz sine wave waveform.

Note The sample values of this waveform must fall between –1.0 and 1.0.

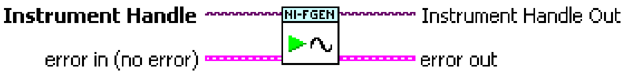
11. Create an arbitrary waveform by calling the niFgen Create Waveform (DBL) instance of the niFgen Create Waveform (poly) VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_CreateWaveformF64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **numberOfSamples**: The size in samples (`500`) of the waveform you created in step 10 **wfmData[]**: The array of waveform samples that you created in step 10

12. Set the waveform handle by calling the niFgen Property Node and selecting **Arbitrary Waveform» Arbitrary Waveform Handle**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE` **value**: The **waveformHandle** from step 11 (sine waveform handle)

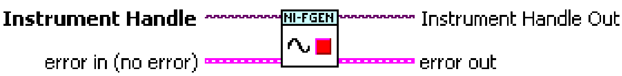
13. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Initiate Generation` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

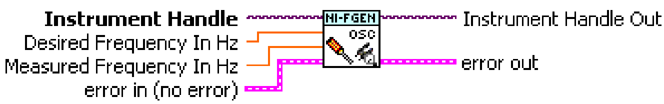
14. Measure the frequency of the generated waveform. This value is the `measured frequency`, which is used in step 15.
15. Repeat steps 15a through 15d for as long as the difference between the measured frequency and the desired frequency (10 MHz) is greater than the tolerance (4.5 ppm).

    The measured frequency should converge on the desired frequency. If the measured frequency does not converge on the desired frequency within 16 iterations, a problem may exist with your measurement device or the NI 5442.

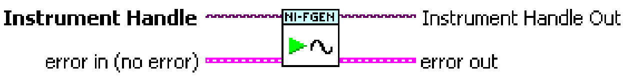
    a. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Abort Generation` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

b. Adjust the oscillator frequency by calling the niFgen Cal Adjust Oscillator Frequency VI.

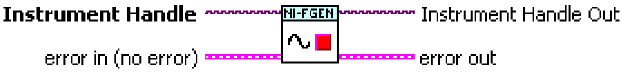
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Cal AdjustOscillator Frequency` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **desiredFrequencyInHz**: `10,000,000` **measuredFrequencyInHz**: The measured frequency value (in Hz)

c. Initiate waveform generation by calling the niFgen Initiate Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Initiate Generation` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`.

d. Measure the frequency of the generated waveform. This value is the `measured frequency`.

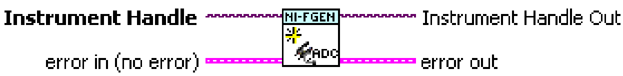
16. Abort waveform generation by calling the niFgen Abort Generation VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_AbortGeneration` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

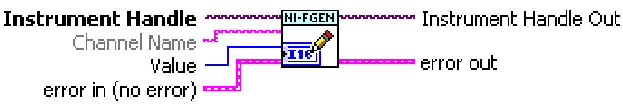
Adjusting the Calibration ADC

The NI 5442 has an onboard calibration ADC that is used during self-calibration. Adjusting the calibration ADC involves characterizing the gain and offset associated with the ADC so that performing self-calibration results in an accurately calibrated device.

1. Initialize ADC calibration by calling niFgen Initialize Cal ADC Calibration VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_InitializeCalADCCalibration` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

2. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `0`

3. Select the fixed Low-Gain analog path by calling the niFgen Property Node and selecting **Output Attributes»Analog Path**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_PATH` **value**: `NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH`

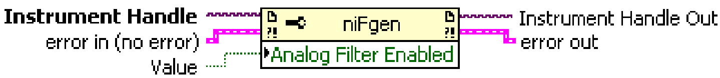
4. Set the gain DAC value by calling the niFgen Property Node and selecting **Calibration»Gain DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_GAIN_DAC_VALUE` **value**: `1,700`

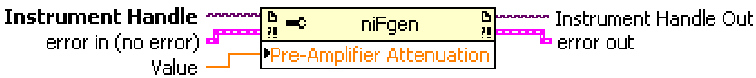
5. Set the offset DAC value by calling the niFgen Property Node and selecting **Calibration»Offset DAC Value**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OFFSET_DAC_VALUE` **value**: `32,767`

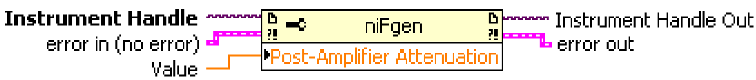
6. Disable the analog filter by calling the niFgen Property Node and selecting **Output Attributes»Analog Filter Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_ANALOG_FILTER_ENABLED` **value**: `VI_FALSE`

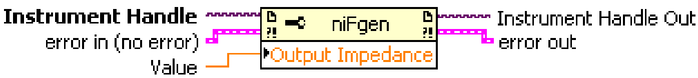
7. Set the preamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Pre-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION` **value**: `0`

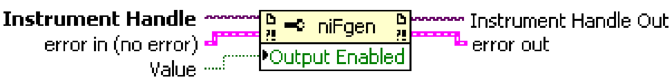
8. Set the postamplifier attenuation by calling the niFgen Property Node and selecting **Calibration»Post-Amplifier Attenuation**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION` **value**: `0`

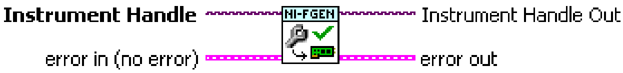
9. Set the output impedance by calling the niFgen Property Node and selecting **Basic Operation»Output Impedance**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViReal64` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE` **value**: `50.00`

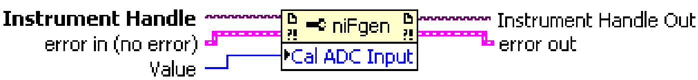
10. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_TRUE`

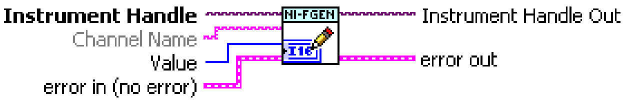
11. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

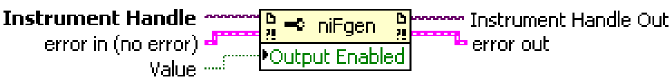
12. Wait 500 ms for the output to settle.
13. Set the calibration ADC input by calling the niFgen Property Node and selecting **Calibration»Cal ADC Input**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `""` (empty string) **attributeId**: `NIFGEN_ATTR_CAL_ADC_INPUT` **value**: `NIFGEN_VAL_ANALOG_OUTPUT`

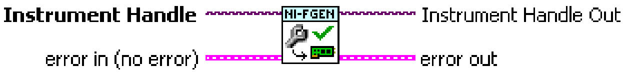
14. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Write Binary16Analog StaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `27,232`

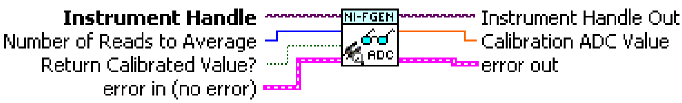
15. Disable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Set AttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ ATTR_OUTPUT_ENABLED` **value**: `VI_FALSE`

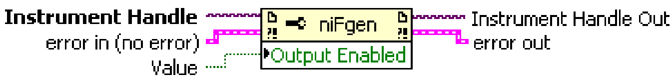
16. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

17. Wait 500 ms for the output to settle.
18. Measure the analog output voltage with the onboard calibration ADC by calling the niFgen Read CAL ADC VI.

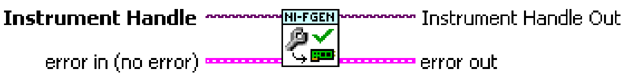
LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Read CalADC` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **numberOfReads ToAverage**: `3` **returnCalibratedValue**: `VI_FALSE` **calADCValue**: Returns a ViReal64 variable. The variable passed by reference through this parameter receives the voltage measured by the onboard ADC. This value is `cal ADC measurement 0`, which is used in step 32.

19. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_SetAttributeViBoolean</code> using the following parameters:

vi: The session handle returned from <code>niFgen_InitExtCal</code>
channelName: <code>"0"</code>
attributeId: <code>NIFGEN_ATTR_OUTPUT_ENABLED</code>
value: <code>VI_TRUE</code></td></tr>
</table>

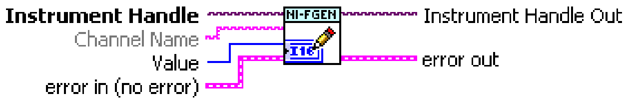
20. Commit the attribute values to the device by calling the niFgen Commit VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_Commit</code> using the following parameter:

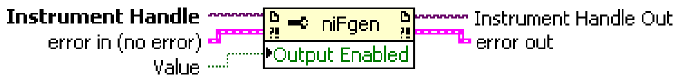
vi: The session handle returned from <code>niFgen_InitExtCal</code></td></tr>
</table>

21. Wait 500 ms for the output to settle.
22. Use the DMM to measure the NI 5442 output voltage directly into the DMM into a high-impedance load. This measurement is `external measurement 0`, which is used in step 32.

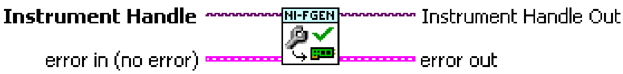
23. Set the main DAC value by calling the niFgen Write Binary 16 Analog Static Value VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_WriteBinary16AnalogStaticValue` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **value**: `10,232`

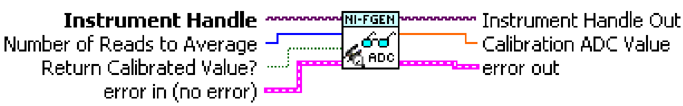
24. Disable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_FALSE`

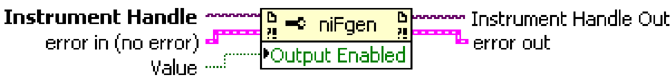
25. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

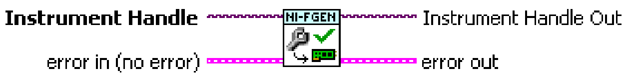
26. Wait 500 ms for the output to settle.
27. Measure the analog output voltage with the onboard calibration ADC by calling the niFgen Read CAL ADC VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Read CalADC` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **numberOfReadsToAverage**: `3` **returnCalibratedValue**: `VI_FALSE` **calADCValue**: Returns a ViReal64 variable. The variable passed by reference through this parameter receives the voltage measured by the onboard ADC. This value is `cal ADC measurement 1`, which is used in step 32.

28. Enable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViBoolean` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `"0"` **attributeId**: `NIFGEN_ATTR_OUTPUT_ENABLED` **value**: `VI_TRUE`

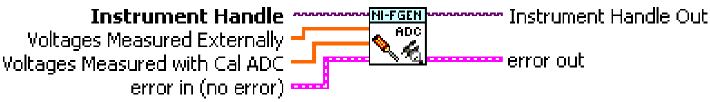
29. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

30. Wait 500 ms for the output to settle.

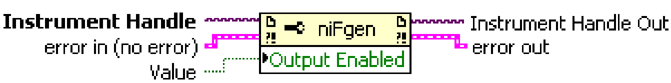
31. Use the DMM to measure the NI 5442 output voltage directly into the DMM (into a high-impedance load). This value is `external measurement 1`, which is used in step 32.

32. Adjust the ADC calibration by calling the niFgen Cal Adjust Cal ADC VI.

<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_Cal AdjustCalADC</code> using the following parameters:

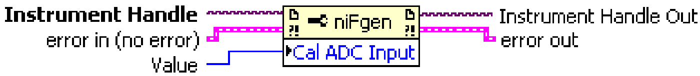
vi: The session handle returned from <code>niFgen_InitExtCal</code>
voltagesMeasured Externally: <code>{external measurement 0, external measurement 1}</code>
voltagesMeasured WithCalADC: <code>{cal ADC measurement 0, cal ADC measurement 1}</code></td></tr>
</table>

33. Disable the analog output by calling the niFgen Property Node and selecting **Basic Operation»Output Enabled**.

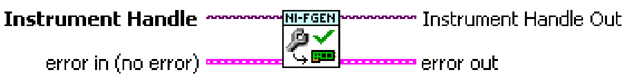
<table>
<tr><th>LabVIEW Block Diagram</th><th>C/C++
Function Call</th></tr>
<tr><td>
</td><td>Call <code>niFgen_Set AttributeViBoolean</code> using the following parameters:

vi: The session handle returned from <code>niFgen_InitExtCal</code>
channelName: <code>"0"</code>
attributeId: <code>NIFGEN_ ATTR_OUTPUT_ENABLED</code>
value: <code>VI_FALSE</code></td></tr>
</table>

34. Set the calibration ADC input by calling the niFgen Property Node and selecting **Calibration»Cal ADC Input**.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_SetAttributeViInt32` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **channelName**: `""` (empty string) **attributeId**: `NIFGEN_ATTR_CAL_ADC_INPUT` **value**: `NIFGEN_VAL_GROUND`

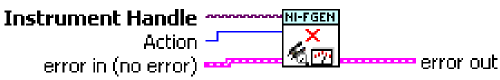
35. Commit the attribute values to the device by calling the niFgen Commit VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_Commit` using the following parameter: **vi**: The session handle returned from `niFgen_InitExtCal`

Closing the External Adjustment Session

When you have completed all the adjustment stages, you must close the external adjustment session to store the new calibration constants in the onboard EEPROM.

Close the instrument driver session and save the calibration date and temperature by calling the niFgen Close Ext Cal VI.

LabVIEW Block Diagram	C/C++ Function Call
	Call `niFgen_CloseExtCal` using the following parameters: **vi**: The session handle returned from `niFgen_InitExtCal` **action**: If the external adjustment procedure completed without any errors, use `NIFGEN_VAL_EXT_CAL_COMMIT`. This function stores the new calibration constants, updated calibration dates, and updated calibration temperatures in the onboard EEPROM. If any errors occurred during the external adjustment procedure, or if you want to abort the operation, use `NIFGEN_VAL_EXT_CAL_ABORT`. This function then discards the new calibration constants and does not change any of the calibration data stored in the onboard EEPROM.

Appendix A: Calibration Procedure Options

External Calibration

External calibration involves both verification and adjustment. Verification is the process of testing the device to ensure that the output accuracy is within certain specifications. Adjustment is the process of measuring and compensating for device performance to improve the output accuracy. A properly verified device is guaranteed to meet or exceed its published specifications for the duration of the calibration interval.

You can use the two sets of test limits provided in this document (the calibration test limits and the published specifications) to perform a verification that determines whether an adjustment process should be performed or, if an adjustment has already been performed, to ensure that it was successful.

If all the output errors determined during verification fall within the calibration test limits, the device is guaranteed to meet or exceed its published specifications for a full calibration interval (two years). For this reason, you must verify against the calibration test limits when performing verification after adjustment.

Published specification values are less restrictive than the calibration test limits. If all the output errors determined during verification fall within the published specifications, but not within the calibration test limits, the device currently meets its published specifications. The device will meet published specifications for the rest of the current calibration interval, but may not remain within these specifications for another two years. In this case, you can perform an adjustment to improve the output accuracy or reset the calibration interval. However, if some output errors determined during verification do not fall within the published specifications, perform an adjustment to restore the device operation to its published specifications.

The *Complete Calibration* section describes the recommended calibration procedure. The *Optional Calibration* section describes alternative procedures that allow you to skip adjustment if the device already meets its calibration test limits or published specifications.

Complete Calibration

Performing a complete calibration is the recommended method of calibration, as it guarantees that the NI 5442 meets or exceeds its published specifications for a two-year calibration interval. At the end of the complete calibration procedure, verify that the output error falls within the calibration test limits. Figure 3 shows the programming flow for complete calibration.

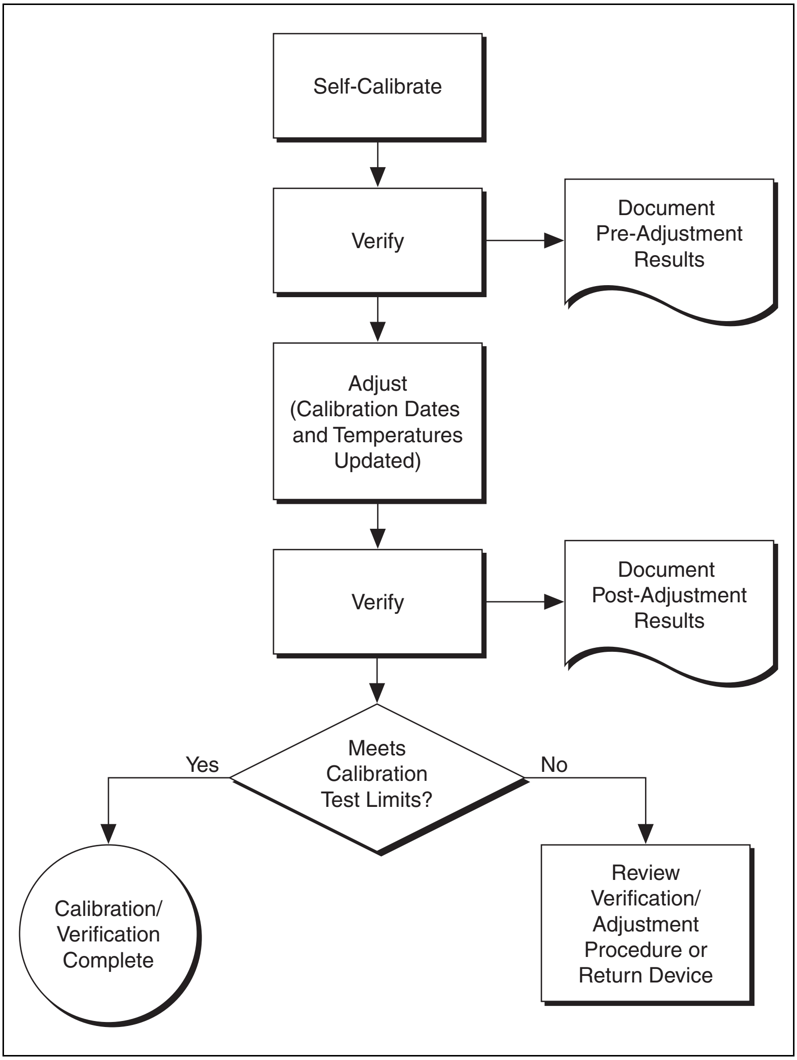


Figure 3. Complete Calibration Programming Flow

Optional Calibration

You can choose to skip the adjustment steps of the calibration procedure if the output error is within the calibration test limits or the published specifications during the first verification. If all the output errors determined during the first verification fall within the calibration test limits, the device is guaranteed to meet or exceed its published specifications for a full calibration interval. In this case, you can update the calibration date, effectively resetting the calibration interval, without actually performing an adjustment. Refer to the *Updating the Calibration Date and Temperature* section for more information about this process.

If all the output errors determined during the first verification fall within the published specifications, but not within the calibration test limits, adjustment is also optional. However, you cannot update the calibration date because the device will not necessarily operate within the published specifications for an additional two years.

Note Regardless of the results of the first verification, if you choose to perform an adjustment, you must verify that the output error falls within the calibration test limits at the end of the calibration procedure.

Refer to Figure 4 for a visual representation of the programming flow for the optional calibration.

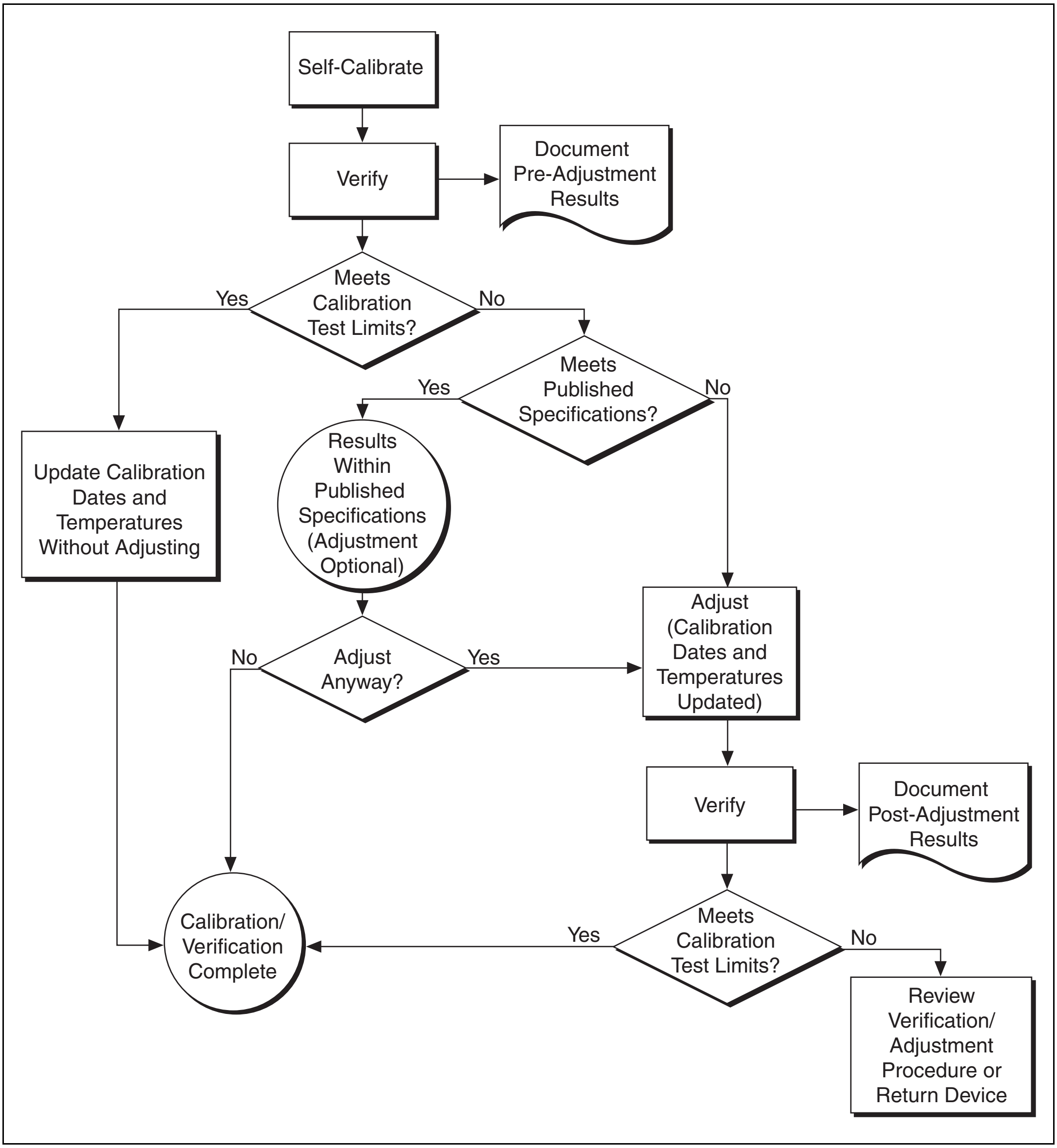


Figure 4. Optional Calibration Programming Flow

Appendix B: Calibration Utilities

NI-FGEN supports several calibration utilities that allow you to perform the following functions:

- Retrieve information about adjustments performed on the NI 5442.
- Restore an external calibration.
- Change the external calibration password.
- Store small amounts of information in the onboard EEPROM.

Note You can retrieve some data using MAX or the FGEN SFP; you can retrieve all the data using NI-FGEN.

MAX

To retrieve data using MAX, complete the following steps:

1. Launch MAX.
2. Navigate to **My System»Devices and Interfaces»NI-DAQmx Devices** and select the device from which you want to retrieve information.
3. Select the **Calibration Tab** on the lower right corner. You should see information about the last calibration dates and temperature for both external and self-calibration.

FGEN SFP

To retrieve data using the FGEN SFP, complete the following steps:

1. Launch the FGEN SFP.
2. Navigate to **Edit»Device Configuration** and select the device from which you want to retrieve information using the Device Configuration dialog box.
3. Navigate to **Edit»Device Configuration Utility»Calibration** to launch the Calibration dialog box. You should see information about the last calibration dates for both external and self-calibration.

NI-FGEN

NI-FGEN provides a full complement of calibration utility VIs and functions. Refer to the *NI Signal Generators Help* for the complete VI and function references. The following VIs are the niFgen Calibration Utility VIs:

- niFgen Get Self Cal Supported
- niFgen Restore Last Ext Cal Constants
- niFgen Get Ext Cal Recommended Interval
- niFgen Get Self Cal Last Date and Time
- niFgen Get Self Cal Last Temp
- niFgen Read Current Temp
- niFgen Get Ext Cal Last Date and Time
- niFgen Get Ext Cal Last Temp
- niFgen Get Cal User Defined Info
- niFgen Set Cal User Defined Info
- niFgen Change Ext Cal Password

The following functions are the niFgen Calibration Utility functions:

- `niFgen_GetSelfCalSupported`
- `niFgen_GetSelfCalLastDateAndTime`
- `niFgen_GetExtCalLastDateAndTime`
- `niFgen_GetSelfCalLastTemp`
- `niFgen_GetExtCalLastTemp`
- `niFgen_GetExtCalRecommendedInterval`
- `niFgen_ChangeExtCalPassword`
- `niFgen_SetCalUserDefinedInfo`
- `niFgen_GetCalUserDefinedInfo`
- `niFgen_GetCalUserDefinedInfoMaxSize`
- `niFgen_ReadCurrentTemperature`
- `niFgen_RestoreLastExtCalConstants`

Where to Go for Support

The National Instruments Web site is your complete resource for technical support. At `ni.com/support` you have access to everything from troubleshooting and application development self-help resources to email and phone assistance from NI Application Engineers.

National Instruments corporate headquarters is located at 11500 North Mopac Expressway, Austin, Texas, 78759-3504. National Instruments also has offices located around the world to help address your support needs. For telephone support in the United States, create your service request at `ni.com/support` and follow the calling instructions or dial 512 795 8248. For telephone support outside the United States, contact your local branch office:

Australia 1800 300 800, Austria 43 662 457990-0,
Belgium 32 (0) 2 757 0020, Brazil 55 11 3262 3599,
Canada 800 433 3488, China 86 21 5050 9800,
Czech Republic 420 224 235 774, Denmark 45 45 76 26 00,
Finland 358 (0) 9 725 72511, France 01 57 66 24 24,
Germany 49 89 7413130, India 91 80 41190000, Israel 972 3 6393737,
Italy 39 02 41309277, Japan 0120-527196, Korea 82 02 3451 3400,
Lebanon 961 (0) 1 33 28 28, Malaysia 1800 887710,
Mexico 01 800 010 0793, Netherlands 31 (0) 348 433 466,
New Zealand 0800 553 322, Norway 47 (0) 66 90 76 60,
Poland 48 22 3390150, Portugal 351 210 311 210, Russia 7 495 783 6851,
Singapore 1800 226 5886, Slovenia 386 3 425 42 00,
South Africa 27 0 11 805 8197, Spain 34 91 640 0085,
Sweden 46 (0) 8 587 895 00, Switzerland 41 56 2005151,
Taiwan 886 02 2377 2222, Thailand 662 278 6777,
Turkey 90 212 279 3031, United Kingdom 44 (0) 1635 523545

372397A-01 Jan08