CALIBRATION PROCEDURE

NI 5412

Contents

NATIONAL INSTRUMENTS™

Introduction

This document contains instructions for calibrating the NI 5412 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 5412.

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

Software

Calibrating the NI 5412 requires installing NI-FGEN version 2.3 or later on the calibration system. You can download NI-FGEN from the National Instruments web site at `ni.com/updates`. NI-FGEN supports programming the *Password* and the *External Calibration Procedures* in the LabVIEW, LabWindows™/CVI™, and C application development environments (ADEs). When you install NI-FGEN, you need to install support only for the ADE that you intend to use.

NI-FGEN 2.3 or later includes all the functions and attributes necessary for calibrating the NI 5412. For LabWindows/CVI, the NI-FGEN function panel (`niFgen.fp`) provides help about the functions available. LabVIEW support is in the `niFgen.llb` file, and all calibration VIs appear in the Functions palette.

Calibration functions are C function calls or LabVIEW VIs in NI-FGEN. In this document, the C function call is shown first, followed by the corresponding LabVIEW VI or NI-FGEN LabVIEW property node, in parentheses. The C function calls are valid for any compiler capable of calling a 32-bit DLL. Many of the functions use constants defined in the `niFgen.h` file. To use these constants in C, you must include `niFgen.h` in your code when you write the calibration procedure. Refer to Table 1 for file locations.

Table 1. Calibration File Locations

File Name and Location	Description
`IVI\Bin\niFgen_32.dll`	The NI-FGEN library, which provides the functionality for calibrating the NI 5412.
`IVI\Lib\msc\niFgen.lib` `IVI\Lib\bc\niFgen.lib`	Allows you to create applications that call functions in the `niFgen_32.dll`: • For Microsoft Visual C/C++, link to `msc\niFgen.lib`. • For LabWindows/CVI, link to the library appropriate to your current compatibility mode (`msc` for Microsoft Visual C/C++).
`IVI\Include\niFgen.h`	A header file for the accessible functions in the `niFgen_32.dll`. You must include this file in any C code that you write to call these functions.

Table 1. Calibration File Locations (Continued)

File Name and Location	Description
`<LabVIEW>\instr.lib\niFgen\niFgen.llb` (LabVIEW)	Contains VIs that correspond to the functions in the `niFgen_32.dll`.
`IVI\Drivers\niFgen\niFgen.fp` (CVI)	Contains the function panels for the functions in `niFgen32.dll`.

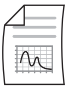
The calibration process is described in the *Password* and the *External Calibration Procedures* sections, including step-by-step instructions on calling the appropriate calibration functions.

Documentation

Consult the following documents for information about the NI PXI-5412, NI-FGEN, and your application software. All documents are available on `ni.com`, and the Help files install with the software.

NI Signal Generators Getting Started Guide

Contains: NI-FGEN installation, hardware installation, and hardware programming



NI PXI-5412 Specifications

Contains: NI PXI-5412 specifications and calibration interval

NI-FGEN Readme

Contains: Operating system and application software support in NI-FGEN

NI Signal Generators Help

Contains: Detailed information about NI PXI-5412.

LabVIEW Help

Contains: LabVIEW programming concepts and reference information about NI-FGEN VIs and functions

NI-FGEN C Reference Help

Contains: Reference information for NI-FGEN C functions and NI-FGEN C properties

Password

The default calibration password is `NI`.

Calibration Interval

National Instruments recommends a calibration interval of one year for the NI PXI-5412. You should adjust the recommended calibration interval based on the measurement accuracy demands of your application.

Test Equipment

External calibration requires different equipment for each applicable specification. Refer to Table 2 for a list of equipment.

Table 2. Equipment Required for Calibrating the NI 5412

Instrument	Applicable Specification	Minimum Specifications	Recommended Instrument
Digital multimeter (DMM)	AC accuracy, DC gain and offset	DCV accuracy: ≤0.05% DC input impedance: ≥1 GΩ ACV accuracy: ≤0.16% AC input impedance: ≥1 MΩ Bandwidth: ≥100 kHz	NI PXI-4070 Agilent HP 34401A Keithley 2000
Banana(m)-to-BNC(f) adapter		—	—
BNC(m)-to-SMB(f) cable		50 Ω, RG-223	—

Table 2. Equipment Required for Calibrating the NI 5412 (Continued)

Instrument	Applicable Specification	Minimum Specifications	Recommended Instrument
Spectrum analyzer, frequency meter, or signal source analyzer	Frequency accuracy	Ability to measure 10 MHz or greater sine waves Frequency accuracy to $\leq$500 ppb	NI PXI-5660 Agilent HP 8560E Agilent HP 53131A or HP 53132A with timebase option 001, 010, or 012 Rohde & Schwarz (R&S) FSUP
BNC(m)-to-SMB(f) cable		50 Ω, RG-223	—
Power meter/sensor	Frequency response (flatness)	VSWR: (50 kHz to 120 MHz) $\leq$ 1.11 Relative power accuracy: $\leq$ 0.022 dB	R&S NRP-Z91
Type N(f)-to-SMB plug adapter		VSWR: 1.3	Pasternak PE9316

Test Conditions

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

- Keep connections to the NI 5412 short.
- Keep relative humidity below 80%.
- Maintain a temperature between 18 °C and 28 °C.
- Observe the 15 minute warm-up time.

Self-Calibration Procedures

The NI 5412 can perform self-calibration, which adjusts the gain and offset of the main analog path. Self-calibration uses only an onboard ADC to measure the output voltage. You can implement self-calibration on the NI 5412 by following procedures similar to the *Verifying the DC Gain and Offset Accuracy* and the *Adjusting the Analog Output* procedures. However, output impedance, oscillator frequency, and the calibration ADC are not adjusted during self-calibration.

You can initiate self-calibration interactively from Measurement & Automation Explorer (MAX) or from the FGEN Soft Front Panel (SFP). Alternatively, you can initiate self-calibration programmatically using NI-FGEN.

MAX

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

1. Launch MAX.
2. Select **My System»Devices and Interfaces»PXI System** from the tree control.
3. Select the device that you want to calibrate.
4. Initiate self-calibration in one of the following ways:
   - Click **Self-Calibrate** in the upper right corner.
   - Right-click the device name and select **Self-Calibrate** from the drop-down menu.

FGEN Soft Front Panel

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

1. Select the device that you want to calibrate using the Device Configuration dialog box (**Edit»Device Configuration**).
2. Open the Calibration dialog box (**Utility»Calibration**).
3. Click **Perform self-calibration**.

NI-FGEN

To self-calibrate the NI 5412 programmatically using NI-FGEN, complete the following steps:

1. Call `niFgen_init` (niFgen Initialize VI) to open an NI-FGEN session using the following parameters:
   - **resourceName**: The name of the device that you want to calibrate. You can find this name under Devices and Interfaces in MAX.
   - **IDQuery**: `VI_TRUE`
   - **resetDevice**: `VI_TRUE`
   - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.
2. Call `niFgen_SelfCal` (niFgen Self Cal VI) using the following parameter:
   - **vi**: The session handle returned from `niFgen_init`
3. Call `niFgen_close` (niFgen Close VI) to close the NI-FGEN session using the following parameter:
   - **vi**: The session handle returned from `niFgen_init`

External Calibration Options

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. You can use verification to ensure that the adjustment process was successful or to determine if the adjustment process needs to be performed.

Adjustment is the process of measuring and compensating for device performance to improve the output accuracy. Performing an adjustment updates the calibration date, resetting the calibration interval. The device is guaranteed to meet or exceed its published specifications for the duration of the calibration interval.

This document provides two sets of test limits for most verification stages, the *calibration test limits* and the *published specifications*. The calibration test limits are more restrictive than the published specifications. 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.

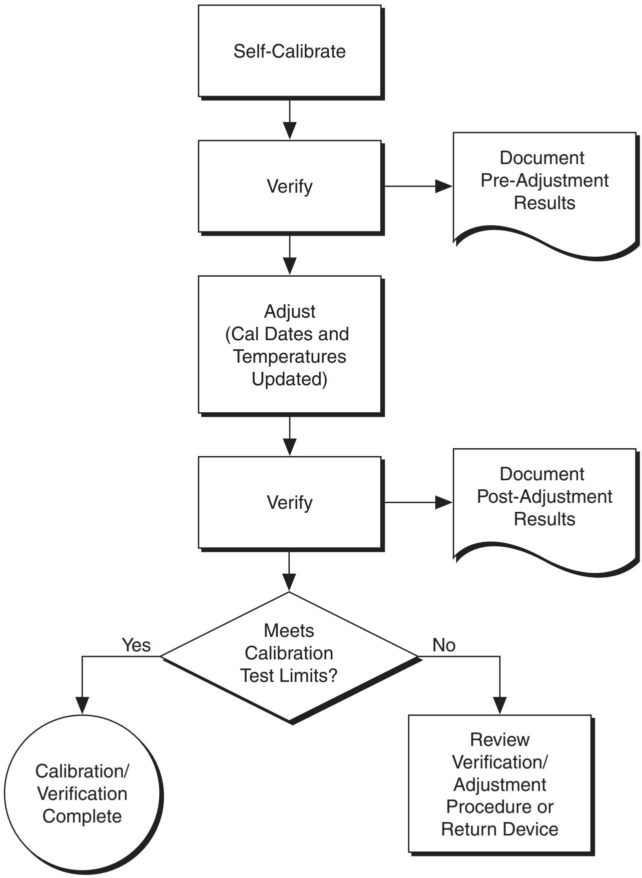
If all the output errors determined during verification fall within the published specifications, but not within the calibration test limits, the device meets its published specifications. However, the device may not remain within these specifications for another two years. The device will meet published specifications for the rest of the current calibration interval. In this case, you can perform an adjustment if you want to improve the output accuracy or reset the calibration interval. If some output errors determined during verification do not fall within the published specifications, you must 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 alternate 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 way to guarantee that the NI 5412 meets or exceeds its published specifications for a two-year calibration interval. At the end of the complete calibration procedure, you verify that the output error falls within the calibration test limits. Figure 1 shows the programming flow for complete calibration.

Figure 1. Complete Calibration Programming Flow



Optional Calibration

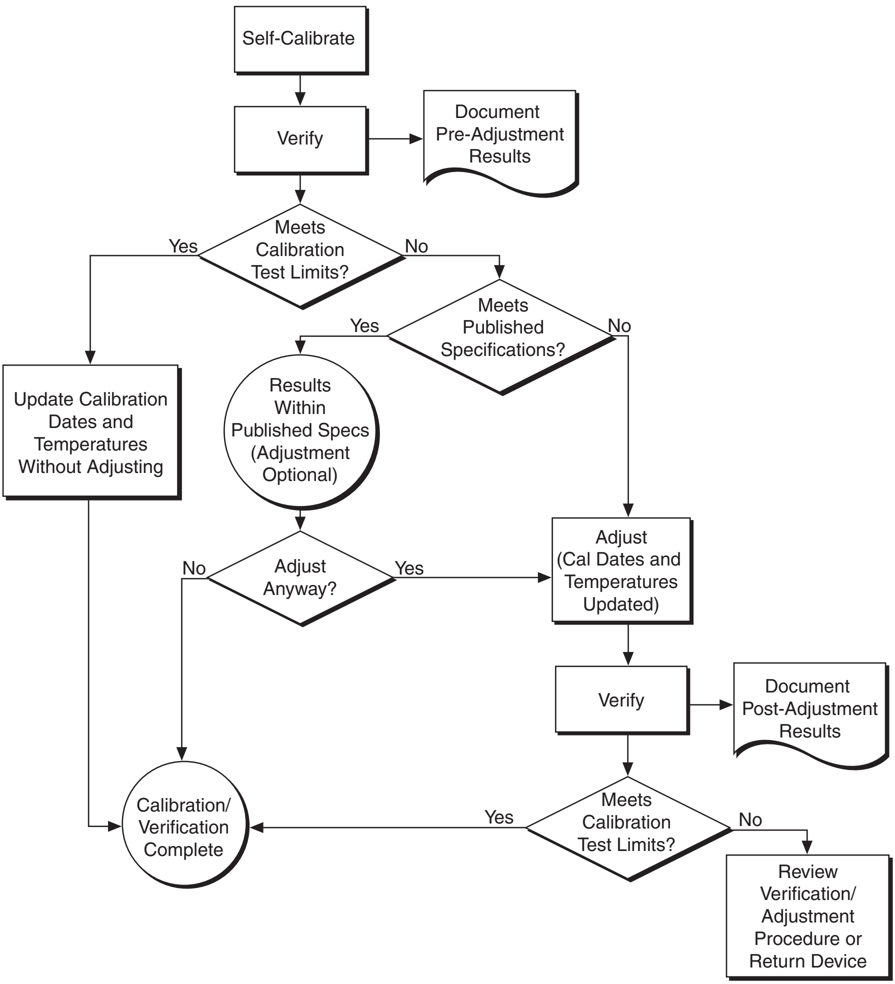
You can choose to skip the adjustment steps of the calibration procedure if the output error is within the calibration test limits or the published specifications during the first verification. If all the output errors determined during the first verification fall within the calibration test limits, 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 *Adjusting the NI 5412* section for more information.

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 cannot 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.

Figure 2 shows the programming flow for the optional calibration.

Figure 2. Optional Calibration Programming Flow



External Calibration Procedures

The complete external calibration procedure consists of self-calibrating, verifying the performance of the NI 5412, adjusting the calibration constants, and verifying again after the adjustments. In some cases, the complete calibration procedure may not be required. Refer to the *External Calibration Options* section for more information.

The external calibration procedure automatically stores the calibration date to allow traceability.

Writing the Calibration Procedure

Before you begin to write the calibration program, review the programming flowcharts in Figures 1 and 2.

Calibration Procedures in LabVIEW

To write calibration procedures in LabVIEW, you must use the VIs included in the `niFgen.llb` file. After installation, these VIs appear within the NI-FGEN Calibration palette under **Functions» Instrument I/O»Instrument Drivers»NI-FGEN»Calibration**.

Calibration Procedures in LabWindows/CVI

To write calibration procedures in LabWindows/CVI, you must use the function panels included in the `niFgen.fp` file. After installation, you can locate the calibration functions under the Calibration class node.

Calibration Procedures in C

To write calibration procedures in C, you must include the `niFgen.h` file in the code that calls the calibration functions, and you must link the `niFgen.lib` file into the build of your executable.

Verifying NI 5412 Specifications

Note Always self-calibrate the NI 5412 before beginning a verification procedure.

This section provides instructions for verifying the NI 5412 specifications and for updating the calibration cycle.

Verification determines whether the device is performing within its specifications prior to external adjustment. Verification and external adjustment together compose a complete calibration. To verify that the NI 5412 still meets its specifications, you must use NI-FGEN to control the NI 5412.

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 5412:

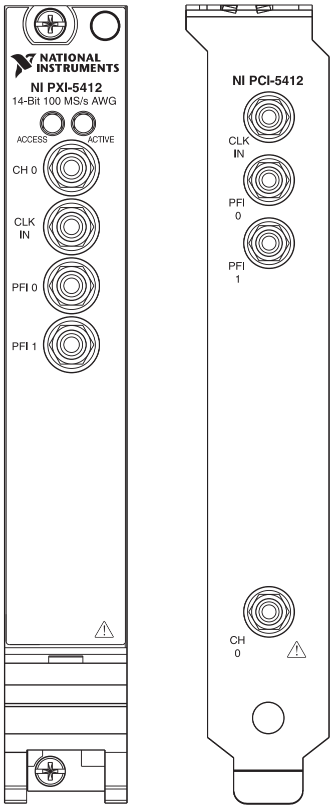
- Oscillator frequency accuracy
- DC gain and offset accuracy
- AC accuracy
- Flatness

The verification procedure for each of these specifications includes setting up, programming, and cleaning up.

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 5412 to NI for repair.

Refer to Table 2 for information about which instrument to use for verifying each specification. Refer to Figure 3 for the names and locations of the NI PXI-5412 and the NI PCI-5412 front panel connectors.

Figure 3. NI PXI-5412 and NI PCI-5412 Front Panel Connectors



Verifying the Oscillator Frequency Accuracy

This test verifies the frequency accuracy of the oscillator on the NI 5412. The verification involves generating a 10 MHz sine wave with the NI 5412 and measuring the sine wave frequency with one of the instruments from Table 2.

To verify the frequency accuracy of the oscillator on the NI 5412, complete the following steps:

1. Connect the NI 5412 CH 0 front panel connector to the instrument measuring the frequency accuracy with a male BNC to female SMB cable.
2. Call `niFgen_init` (niFgen Initialize VI) using the following parameters:
   - **resourceName**: The name of the device that you want to verify. You can find this name under Devices and Interfaces in MAX.
   - **IDQuery**: `VI_TRUE`
   - **resetDevice**: `VI_TRUE`
   - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.
3. Call `niFgen_SetAttributeViReal64` to set the sample rate (niFgen property node: **Arbitrary Waveform Output»Sample Rate**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_ARB_SAMPLE_RATE`
   - **value**: `100000000`
4. Call `niFgen_SetAttributeViReal64` to set the gain (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Gain**) 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. Call `niFgen_SetAttributeViReal64` to set the offset (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Offset**) 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. Call `niFgen_SetAttributeViBoolean` to set the digital filter state (niFgen property node: **Output Attributes»Digital Filter Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_DIGITAL_FILTER_ENABLED`
   - **value**: `VI_TRUE`
7. Call `niFgen_SetAttributeViReal64` to set the digital filter interpolation factor (niFgen property node: **Output Attributes» Digital Filter Interpolation Factor**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_DIGITAL_FILTER_INTERPOLATION_FACTOR`
   - **value**: `4`
8. 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. Because you set the sample rate to 100 MS/s and because you are using 10 samples per cycle, the resulting waveform is a 10 MHz sine wave.

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

9. Call `niFgen_CreateArbWaveform` (niFgen Create Arbitrary Waveform VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **wfmSize**: 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.
   - **wfmHandle**: The variable passed by reference through this parameter receives the value (waveform handle) that identifies the waveform created by this function.
10. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`

11. Measure the frequency out of the NI 5412.

    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 3 for frequency ranges.

Table 3. Frequency Ranges

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

12. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
13. Call `niFgen_close` (niFgen Close VI) to close the instrument driver session, to destroy the instrument driver session and all of its properties, and to release any memory resources NI-FGEN uses. Use 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 5412 into a high-impedance load by generating a number of DC voltages and offsets, measuring the voltage with a DMM, and comparing the NI 5412 to the error limits.

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

- Main analog path gain
- Main analog path offset

Verifying the Main Analog Path Gain

To verify the gain of the NI 5412 main analog path, complete the following steps:

1. Connect the NI 5412 CH 0 front panel connector to the DMM for measuring DC gain and offset accuracy.
2. Call `niFgen_init` (niFgen Initialize VI) using the following parameters:
    - **resourceName**: The name of the device that you want to verify. You can find this name under Devices and Interfaces in MAX.
    - **IDQuery**: `VI_TRUE`
    - **resetDevice**: `VI_TRUE`
    - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.

3. Call `niFgen_SetAttributeViReal64` to set the load impedance (niFgen property node: **Output»Load Impedance**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_LOAD_IMPEDANCE`
   - **value**: `10000000000`
4. Call `niFgen_SetAttributeViInt32` to set the analog path (niFgen property node: **Output Attributes»Analog Path**) 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`
5. Call `niFgen_SetAttributeViReal64` to set the output impedance (niFgen property node: **Basic Operation»Output Impedance**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE`
   - **value**: `50`
6. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
   - **value**: `VI_TRUE`
7. 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).
8. Call `niFgen_CreateArbWaveform` (niFgen Create Arbitrary Waveform VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **wfmSize**: The size in samples (`500`) of the waveform you created in step 7.
   - **wfmData**: The array of waveform samples that you created in step 7.
   - **wfmHandle**: The variable passed by reference through this parameter receives the value (waveform handle) that identifies the waveform created by this function (positive full-scale handle).
9. 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).

10. Call `niFgen_CreateArbWaveform` (niFgen Create Arbitrary Waveform VI) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **wfmSize**: The size in samples (`500`) of the waveform that you created in step 9.
    - **wfmData**: The array of waveform samples that you created in step 9.
    - **wfmHandle**: The variable passed by reference through this parameter receives the value (waveform handle) that identifies the waveform created by this function (negative full-scale handle).
11. Call `niFgen_SetAttributeViReal64` to set the offset (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Offset**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_ARB_OFFSET`
    - **value**: `0`

Repeat steps 12 through 22 for each of the 24 iterations listed in Table 4, changing the *Gain* value for each iteration.

Table 4. 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	12.000000	12.000000	-12.000000	±0.019700	±0.048500
2	10.000000	10.000000	-10.000000	±0.016500	±0.040500
3	7.000000	7.000000	-7.000000	±0.011700	±0.028500
4	5.000000	5.000000	-5.000000	±0.008500	±0.020500
5	3.500000	3.500000	-3.500000	±0.006100	±0.014500
6	2.500000	2.500000	-2.500000	±0.004500	±0.010500
7	2.000000	2.000000	-2.000000	±0.003700	±0.008500
8	1.650000	1.650000	-1.650000	±0.003140	±0.007100
9	1.250000	1.250000	-1.250000	±0.002500	±0.005500
10	0.850000	0.850000	-0.850000	±0.001860	±0.003900
11	0.600000	0.600000	-0.600000	±0.001460	±0.002900
12	0.415000	0.415000	-0.415000	±0.001164	±0.002160

Table 4. Values for Verifying the Gain of the Main Analog Path (Continued)

Iteration	Gain	Ideal Positive Full-Scale (Volts)	Ideal Negative Full-Scale (Volts)	Calibration Test Limit (Volts)	Published Specification (Volts)
13	0.300000	0.300000	-0.300000	±0.000980	±0.001700
14	0.205000	0.205000	-0.205000	±0.000828	±0.001320
15	0.150000	0.150000	-0.150000	±0.000740	±0.001100
16	0.105000	0.105000	-0.105000	±0.000668	±0.000920
17	0.075000	0.075000	-0.075000	±0.000620	±0.000800
18	0.055000	0.055000	-0.055000	±0.000588	±0.000720
19	0.037500	0.037500	-0.037500	±0.000560	±0.000650
20	0.026000	0.026000	-0.026000	±0.000542	±0.000604
21	0.018500	0.018500	-0.018500	±0.000530	±0.000574
22	0.013000	0.013000	-0.013000	±0.000521	±0.000552
23	0.009000	0.009000	-0.009000	±0.000514	±0.000536
24	0.006500	0.006500	-0.006500	±0.000510	±0.000526
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*)					

12. Call `niFgen_SetAttributeViReal64` to set the gain (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Gain**) 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.
13. Call `niFgen_SetAttributeViInt32` to choose the positive full-scale DC waveform (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Handle**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE`
    - **value**: The **wfmHandle** from step 8 (positive full-scale handle).

14. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
15. Measure the DC voltage output of the NI 5412. This value is the *Measured Positive Full-Scale Value*.
16. 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* listed in Table 4.
17. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
18. Call `niFgen_SetAttributeViInt32` to choose the negative full-scale DC waveform (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Handle**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE`
    - **value**: The **wfmHandle** from step 10 (negative full-scale handle).
19. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
20. Measure the DC voltage out of the NI 5412. This value is the *Measured Negative Full-Scale Value*.
21. Calculate the error for negative full-scale using the following formula:

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

    Compare this error to the *Published Specification* or the *Calibration Test Limit* listed in Table 4.
22. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
23. If any of the errors are greater than the *Calibration Test Limit*, perform an external adjustment.

Verifying the Main Analog Path Offset

To verify the offset of the NI 5412 main analog path, complete 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. Call `niFgen_CreateArbWaveform` (niFgen Create Arbitrary Waveform VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **wfmSize**: 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.
   - **wfmHandle**: The variable passed by reference through this parameter receives the value (waveform handle) that identifies the waveform created by this function (mid-scale handle).
3. Call `niFgen_SetAttributeViInt32` to choose the mid-scale handle DC waveform (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Handle**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE`
   - **value**: The **wfmHandle** from step 2 (mid-scale handle).

Repeat steps 4 through 14 for each of the 24 iterations listed in Table 5, changing the *Ideal Positive Offset*, *Ideal Negative Offset*, and *Gain* values for each iteration.

Table 5. 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	12.000000	6.000000	-6.000000	±0.021500	±0.051500
2	10.000000	5.000000	-5.000000	±0.018000	±0.043000
3	7.000000	3.500000	-3.500000	±0.012750	±0.030250
4	5.000000	2.500000	-2.500000	±0.009250	±0.021750
5	3.500000	1.750000	-1.750000	±0.006625	±0.015375
6	2.500000	1.250000	-1.250000	±0.004875	±0.011125
7	2.000000	1.000000	-1.000000	±0.004000	±0.009000
8	1.650000	0.825000	-0.825000	±0.003388	±0.007513
9	1.250000	0.625000	-0.625000	±0.002688	±0.005813

Table 5. Values for Verifying the Offset of the Main Analog Path (Continued)

Iteration	Gain	Ideal Positive Offset (Volts)	Ideal Negative Offset (Volts)	Calibration Test Limit (Volts)	Published Specification (Volts)
10	0.850000	0.425000	-0.425000	±0.001988	±0.004113
11	0.600000	0.300000	-0.300000	±0.001550	±0.003050
12	0.415000	0.207500	-0.207500	±0.001226	±0.002264
13	0.300000	0.150000	-0.150000	±0.001025	±0.001775
14	0.205000	0.102500	-0.102500	±0.000859	±0.001371
15	0.150000	0.075000	-0.075000	±0.000763	±0.001138
16	0.105000	0.052500	-0.052500	±0.000684	±0.000946
17	0.075000	0.037500	-0.037500	±0.000631	±0.000819
18	0.055000	0.027500	-0.027500	±0.000596	±0.000734
19	0.037500	0.018750	-0.018750	±0.000566	±0.000659
20	0.026000	0.013000	-0.013000	±0.000546	±0.000611
21	0.018500	0.009250	-0.009250	±0.000532	±0.000579
22	0.013000	0.006500	-0.006500	±0.000523	±0.000555
23	0.009000	0.004500	-0.004500	±0.000516	±0.000538
24	0.006500	0.003250	-0.003250	±0.000511	±0.000528
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. Call `niFgen_SetAttributeViReal64` to set the offset (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Offset**) 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 5 for the current iteration.

5. Call `niFgen_SetAttributeViReal64` to set the gain (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Gain**) 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.
6. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
   - **vi**: The session handle returned from `niFgen_init`
7. Measure the positive DC voltage out of the NI 5412. This value is the *Measured Positive Offset Value*.
8. Calculate 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* listed in Table 5.
9. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
   - **vi**: The session handle returned from `niFgen_init`
10. Call `niFgen_SetAttributeViReal64` to set the offset (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Offset**) 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 5 for the current iteration.
11. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
12. Measure the negative DC voltage out of the NI 5412. This value is the *Measured Negative Offset Value*.
13. Calculate the error for positive offset using the following formula:

    Error Negative Offset = (*Measured Negative Offset Value*) - (*Ideal Negative Offset Value*)

    Compare this error to the *Published Specification* or the *Calibration Test Limit* listed in Table 5.

14. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform 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 AC Voltage Amplitude Absolute Accuracy

This test verifies the AC voltage amplitude absolute accuracy of the NI 5412 using a DMM. To verify the AC accuracy of the NI 5412, complete the following steps:

1. Connect the NI 5412 CH 0 front panel connector to the DMM using the SMB(m)-to-BNC(f) cable and the BNC(f)-to-banana(m) adapter. Connect the SMB connector of the cable to CH 0 of the NI 5412. Connect the BNC connector of the cable to the BNC(f)-to-banana(m) adapter. Connect the banana adapter to the DMM, maintaining the correct polarity.
2. Call `niFgen_init` (niFgen Initialize VI) using the following parameters:
    - **resourceName**: The name of the device that you want to verify. You can find this name under Devices and Interfaces in MAX.
    - **IDQuery**: `VI_TRUE`
    - **resetDevice**: `VI_TRUE`
    - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.
3. Call `niFgen_ConfigureChannels` (niFgen Configure Channels VI) using the following parameters:
    - **Channels**: `"0"`
    - **vi**: The session handle returned from `niFgen_init`
4. Call `niFgen_ConfigureSampleRate` (niFgen Set Sample Rate VI) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **Sample Rate**: 100 Ms/s (`100000000`)
5. Call `niFgen_SetAttributeViReal64` to set the load impedance (niFgen property node: **Output»Load Impedance**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_LOAD_IMPEDANCE`
    - **value**: `1000000`
6. Call `niFgen_ConfigureOutputMode` (niFgen Configure Output Mode VI) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **Output Mode**: `NIFGEN_VAL_OUTPUT_ARB` (Arbitrary Waveform)

7. Call `niFgen_CreateWaveformF64` (niFgen Create Waveform (DBL) VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **wfmSize**: The size in samples (`2000`) of the waveform.
   - **wfmData**: The array of waveform samples (double representation).
   - **wfmHandle**: A pointer to a waveform. The variable passed by reference through this parameter acts as a handle to the waveform and can be used for setting the active waveform, changing the data in the waveform, building sequences of waveforms, or deleting the waveform when it is no longer needed.

Note In LabVIEW, you must call the niFgen Util Create Waveform Data VI to generate a single cycle sine wave with 2,000 samples and an amplitude of 1. Wire the output of the niFgen Util Create Waveform Data VI to the **Waveform Data Array** input of the niFgen Create Waveform (DBL) VI.

8. Configure the DMM using the following settings:
   - Function: AC voltage
   - Range: Refer to Table 6
   - Input impedance: 1 MΩ
   - Average readings: 4
   - Digits: 6.5

Note These values assume you are using an NI 4070 DMM. For other DMMs, use the range closest to the values listed in step 8. The input impedance should be equal to or greater than the values indicated in Table 2, *Equipment Required for Calibrating the NI 5412*.

9. Repeat steps 10 through 15 for each of the 24 iterations listed in Table 6, changing the *Gain* and *DMM Range (VRMS)* values for each iteration.

Table 6. Values for Verifying the AC Voltage Amplitude Absolute Accuracy

Iteration	Gain	DMM Range (VRMS)	Expected Amplitude (VRMS)	Test Limit (-VRMS)	Test Limit (+VRMS)
1	12.000000	50	8.485281	-0.085560	0.170413
2	10.000000	50	7.071068	-0.0714178	0.142128
3	7.000000	5	4.949747	-0.050205	0.099702
4	5.000000	5	3.535534	-0.036062	0.071418
5	3.500000	5	2.474874	-0.025456	0.050205

Table 6. Values for Verifying the AC Voltage Amplitude Absolute Accuracy (Continued)

Iteration	Gain	DMM Range (VRMS)	Expected Amplitude (VRMS)	Test Limit (-VRMS)	Test Limit (+VRMS)
6	2.500000	5	1.767767	-0.018385	0.036062
7	2.000000	5	1.414214	-0.014849	0.028991
8	1.650000	5	1.166726	-0.012374	0.024042
9	1.250000	5	0.883883	-0.009546	0.018385
10	0.850000	5	0.601041	-0.006718	0.012728
11	0.600000	0.5	0.424264	-0.004950	0.009192
12	0.415000	0.5	0.293449	-0.003642	0.006576
13	0.300000	0.5	0.212132	-0.002828	0.004950
14	0.205000	0.5	0.144957	-0.0021587	0.003606
15	0.150000	0.5	0.106066	-0.001768	0.002828
16	0.105000	0.5	0.074246	-0.001450	0.002192
17	0.075000	0.5	0.053033	-0.001237	0.001768
18	0.055000	0.5	0.038809	-0.001096	0.001485
19	0.037500	0.5	0.026517	-0.003359	0.006010
20	0.026000	0.5	0.018385	-0.0008910	0.001075
21	0.018500	0.5	0.013081	-0.000838	0.000969
22	0.013000	0.5	0.009192	-0.000799	0.000891
23	0.009000	0.5	0.006364	-0.000771	0.000834
24	0.006500	0.5	0.004596	-0.000753	0.000799

10. Call `niFgen_SetAttributeViReal64` (niFgen property node: **Arbitrary Waveform»Gain**) to set the gain 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 6 for the current iteration

11. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
12. Wait 5 seconds for the output of the NI 5412 to settle.
13. Measure and record the output voltage amplitude with the DMM. This value is the measured amplitude, *measuredVRMS*.
14. Calculate the peak amplitude error using the following equation:

$$expected\ VRMS - measured\ VRMS = error$$

15. Compare the output error to the test limits in Table 6 for the current iteration.
16. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the current generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
17. If any of the errors are greater than the test limits, perform an external adjustment.

Verifying Frequency Response (Flatness)

This test verifies the frequency response (flatness) of the NI 5412 using a power meter. To verify the flatness of the NI 5412, complete the following steps:

1. Connect the NI 5412 CH 0 front panel connector to the power meter using the required adapter.
2. Call `niFgen_init` (niFgen Initialize VI) using the following parameters:
    - **resourceName**: The name of the device that you want to verify. You can find this name under Devices and Interfaces in MAX.
    - **IDQuery**: `VI_TRUE`
    - **resetDevice**: `VI_TRUE`
    - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.
3. Call `niFgen_ConfigureChannels` (niFgen Configure Channels VI) using the following parameters:
    - **Channels**: `"0"`
    - **vi**: The session handle returned from `niFgen_init`
4. Call `niFgenSetAttributeViBoolean` (niFgen property node: **Output»Output Enabled**) to disable the NI 5412 output. Use the following parameters:
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_FALSE`
    - **vi**: The session handle returned from `niFgen_init`

5. Call `niFgenCommit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
   - **vi:** The session handle returned from `niFgen_init`
6. Null the power meter according to the power meter documentation.
7. Configure the power meter using the following settings:
   - Average: 16
   - Measure watts
   - Channel 0 power sensor connected to the NI 5412
   - High accuracy
8. Call `niFgen_ConfigureSampleRate` (niFgen Set Sample Rate VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **Sample Rate**: 100 MS/s (`100000000`)
9. Call `niFgen_CreateWaveformF64` (niFgen Create Waveform (DBL) VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_init`
   - **channelName**: `"0"`
   - **wfmSize**: The size in samples of the waveform.
   - **wfmArray**: The array of waveform samples (double representation).
   - **wfmHandle**: A pointer to a waveform. The variable passed by reference through this parameter acts as a handle to the waveform and can be used for setting the active waveform, changing the data in the waveform, building sequences of waveforms, or deleting the waveform when it is no longer needed.

Note In LabVIEW, you must call the Sine Pattern VI to create an array of waveform samples with an **amplitude** of 1, and **samples** and **cycles** that correspond to the current iteration in Table 7. Wire the **Sinusoidal Pattern** output of the Sine Pattern VI to the **Waveform Data Array** input of the niFgen Create Waveform (DBL) VI.

Table 7. NI 5412 Setup for Frequency Response (Flatness) Verification

Iteration	Frequency (Hz)	Number of Samples	Number of Cycles	Published Specification	
				Low-Gain Amplifier	High-Gain Amplifier
1	50,000	2,000	1	-1 dB to +1 dB	-1 dB to +1 dB
2	1,000,000	1,000	10	-1 dB to +1 dB	-1 dB to +1 dB
3	2,000,000	1,000	20	-1 dB to +1 dB	-1 dB to +1 dB
4	3,000,000	1,000	30	-1 dB to +1 dB	-1 dB to +1 dB

Table 7. NI 5412 Setup for Frequency Response (Flatness) Verification

Iteration	Frequency (Hz)	Number of Samples	Number of Cycles	Published Specification	
				Low-Gain Amplifier	High-Gain Amplifier
5	4,000,000	1,000	40	-1 dB to +1 dB	-1 dB to +1 dB
6	5,000,000	1,000	50	-1 dB to +1 dB	-1 dB to +1 dB
7	6,000,000	1,000	60	-1 dB to +1 dB	-1 dB to +1 dB
8	20,000,000	1,000	200	-3 dB to 0 dB	-3 dB to 0 dB

10. Call `niFgen_ConfigureArbWaveform` (niFgen Configure Arbitrary Waveform VI) using the following parameters:
    - **channelName**: `"0"`
    - **wfmHandle**: The waveform handle returned from `niFgen_CreateWaveformF64`.
    - **Gain**: `1`
    - **Offset**: `0`
    - **vi**: The session handle returned from `niFgen_init`
11. Call `niFgen_SetAttributeViBoolean` (niFgen property node: **Output»Output Enabled**) to enable the NI 5412 output using the following parameters:
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_TRUE`
    - **vi**: The session handle returned from `niFgen_init`
12. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`
13. Allow the power meter to stabilize for 10 seconds.
14. Measure and record the reference (50 kHz) power (W_{ref}) of the positive output in Watts.
15. Configure the NI 5412 and power meter frequency according to the next iteration in Table 7.
16. Using the recorded power values, calculate the deviation from the reference (50 kHz) power using the following equation:

$$Flatness(dB) = 10\log\left(\frac{W_f}{W_{ref}}\right)$$

17. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the current generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_init`

18. Repeat steps 9 through 17 for each iteration in Table 7.
19. If any of the errors are greater than the test limits, perform an external adjustment.

Note To verify the flatness for the High-Gain Amplifier path, repeat steps 1 through 18, but set the Gain to 3 in step 10.

Adjusting the NI 5412

If the NI 5412 successfully passes all verification within the calibration test limits, adjustment is recommended, but not required, to guarantee its published specifications for the next two years. If the NI 5412 was not within the calibration test limits for each verification procedure, perform the adjustment procedure to improve the accuracy of the NI 5412. Refer to the *External Calibration Options* section to determine which procedures to perform.

An adjustment is required only once every two years. The adjustment procedure automatically updates the calibration date and temperature in the EEPROM of the NI 5412.

If the NI 5412 passed verification within the calibration test limits and you do not want to do an adjustment, you can update the calibration date and onboard calibration temperature without making any adjustments by completing the following steps:

1. Call `niFgen_InitExtCal` (niFgen Init Ext Cal VI) to open an NI-FGEN external calibration session using the following parameters:
   - **resourceName**: The name of the device you want to calibrate. You can find this name under Devices and Interfaces in MAX.
   - **password**: The password required to open an external calibration session. If this password has not been changed since manufacturing, the password is `"NI"`.
   - **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the external calibration session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.
2. Call `niFgen_CloseExtCal` (niFgen Close Ext Cal VI) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **action**: `NIFGEN_VAL_EXT_CAL_COMMIT`

The external calibration procedure adjusts the analog output, the oscillator frequency, and the calibration ADC. Analog output adjustment characterizes the DC gains and the offsets of the analog path to ensure the analog output voltage accuracy. Adjusting the oscillator frequency adjusts 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.

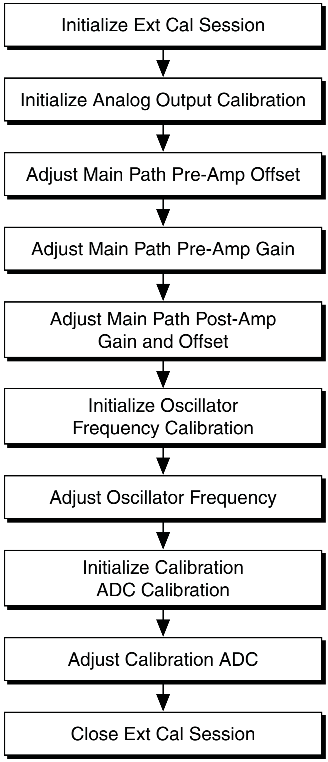
You cannot perform an external calibration using a standard NI-FGEN session. You must create an external calibration session using `niFgen_InitExtCal` (niFgen Init Ext Cal VI). An external calibration session allows you to use NI-FGEN functions and attributes that are specifically for external calibration, while still allowing you to use all the standard NI-FGEN functions and attributes with the external calibration session.

Along with the standard NI-FGEN attributes, the external calibration session uses 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 using `niFgen_CloseExtCal` (niFgen Close Ext Cal VI).

Figure 4 shows the programming flow for an external calibration procedure.

Figure 4. NI 5412 External Calibration Procedure



Initializing the External Calibration Session

Call `niFgen_InitExtCal` (niFgen Init Ext Cal VI) to open an NI-FGEN external calibration session using the following parameters:

- **resourceName**: The name of the device you want to calibrate. This name can be found under Devices and Interfaces in MAX.
- **password**: The password required to open an external calibration session. If this password has not been changed since manufacturing, the password is `"NI"`.

- **vi**: A pointer to a ViSession. The variable passed by reference through this parameter receives the value that identifies the external calibration session created by this function. This value acts as the session handle and is passed as the first parameter to all subsequent NI-FGEN functions.

Adjusting the Analog Output

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

- Main path pre-amplifier offset
- Main path pre-amplifier gain
- Main path post-amplifier gain and offset

In each of these subprocedures, you put the device in several configurations and take several output measurements. You then pass these measurements to NI-FGEN, which determines the calibration constants for the device.

Initializing Analog Output Calibration

1. Call `niFgen_InitializeAnalogOutputCalibration` (niFgen Initialize Analog Output Calibration VI) using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
2. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **value**: `0`
3. Call `niFgen_SetAttributeViInt32` to set the analog path value (niFgen property node: **Output Attributes»Analog Path**) 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. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
    - **value**: `2000`
5. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`

- **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
- **value**: `32767`

6. Call `niFgen_SetAttributeViReal64` to set the pre-amplifier attenuation (niFgen property node: **Calibration»Pre-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION`
   - **value**: `0`
7. Call `niFgen_SetAttributeViReal64` to set the post-amplifier attenuation (niFgen property node: **Calibration»Post-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION`
   - **value**: `0`
8. Call `niFgen_SetAttributeViReal64` to set the output impedance (niFgen property node: **Basic Operation»Output Impedance**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE`
   - **value**: `50`
9. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
   - **value**: `VI_TRUE`
10. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`

Adjusting the Main Path Pre-Amplifier Offset

1. Call `niFgen_SetAttributeViInt32` to set the analog path value (niFgen property node: **Output Attributes»Analog Path**) 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. Call `niFgen_SetAttributeViReal64` to set the post-amplifier attenuation (niFgen property node: **Calibration»Post-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION`
   - **value**: `0`
3. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **value**: `0`

Repeat steps 4 through 6 for each of the 5 iterations listed in Table 8, changing the *Pre-Amplifier Attenuation* and *Current Configuration* values for each iteration.

Table 8. Attributes and Values for Main Path Pre-Amplifier Offset

Iteration	Pre-Amplifier Attenuation	Current Configuration
1	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_0DB
2	3	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_3DB
3	6	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_6DB
4	9	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_9DB
5	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_12DB

4. Call `niFgen_SetAttributeViReal64` to set the pre-amplifier attenuation (niFgen property node: **Calibration»Pre-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION`
   - **value**: The *Pre-Amplifier Attenuation* value for the current iteration from Table 8
5. Take the following voltage measurements at the NI 5412 CH 0 front panel connector into a high-impedance load:
   a. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
      - **vi**: The session handle returned from `niFgen_InitExtCal`
      - **channelName**: `"0"`
      - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
      - **value**: `2000`

b. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
   - **value**: `50000`
c. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage output by the device. This measurement is `measurement 0`, which is used in step 6.
f. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
   - **value**: `1000`
g. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage output by the device. This measurement is `measurement 1`, which is used in step 6.
j. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
   - **value**: `15000`
k. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage output of the device. This measurement is `measurement 2`, which is used in step 6.

6. Call `niFgen_CalAdjustMainPathPreAmpOffset` (niFgen Cal Adjust Main Path Pre Amp Offset VI) 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.
   - **gainDACValues**: An array containing two elements—the two values (`2000`, `1000`) that you set as the gain DAC in the order that you measured them.
   - **offsetDACValues**: An array containing two elements—the two values (`50000`, `15000`) that you set as the offset DAC in the order that you measured them.
   - **measuredOutputs**: An array containing three elements—the three output voltages (`measurement 0`, `measurement 1`, `measurement 2`) that you measured in the order that you measured them.

Adjusting the Main Path Pre-Amplifier Gain

1. Call `niFgen_SetAttributeViInt32` to set the analog path value (niFgen property node: **Output Attributes»Analog Path**) 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. Call `niFgen_SetAttributeViReal64` to set the post-amplifier attenuation (niFgen property node: **Calibration»Post-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION`
   - **value**: `0`
3. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
   - **value**: `32000`

Repeat steps 4 through 6 for each of the 5 iterations listed in Table 9, changing the *Pre-Amplifier Attenuation* and *Current Configuration* values for each iteration.

Table 9. Attributes and Values for Main Path Pre-Amplifier Gain

Iteration	Pre-Amplifier Attenuation	Current Configuration
1	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_0DB
2	3	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_3DB
3	6	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_6DB
4	9	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_9DB
5	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_FILTER_OFF_12DB

4. Call `niFgen_SetAttributeViReal64` to set the pre-amplifier attenuation (niFgen property node: **Calibration»Pre-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION`
   - **value**: The *Pre-Amplifier Attenuation* value for the current iteration from Table 9.
5. Call the following functions and take voltage measurements at the NI 5412 CH 0 front panel connector into a high-impedance load:
   a. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
      - **vi**: The session handle returned from `niFgen_InitExtCal`
      - **channelName**: `"0"`
      - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
      - **value**: `1500`

   b. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
      - **vi**: The session handle returned from `niFgen_InitExtCal`
      - **channelName**: `"0"`
      - **value**: `25233`

   c. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage output by the device. This measurement is `measurement 0`, which is used in step 6.

f. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
   - **value**: `2000`

g. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **value**: `-29232`

h. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 6.

6. Call `niFgen_CalAdjustMainPathPreAmpGain` (niFgen Cal Adjust Main Path Pre Amp Gain VI) 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 9.
   - **mainDACValues**: An array containing two elements—the two values (`25233`, `-29232`) that you set for the main DAC in the order that you measured them.
   - **gainDACValues**: An array containing two elements—the two values (`1500`, `2000`) that you set for the gain DAC in the order that you measured them.
   - **offsetDACValues**: An array containing one element—the value (`32000`) that you set for the offset DAC.
   - **measuredOutputs**: An array containing two elements—the two output voltages (`measurement 0`, `measurement 1`) that you measured in the order that you measured them.

Adjusting the Main Path Post-Amplifier Gain and Offset

1. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **value**: `0`

2. Call `niFgen_SetAttributeViReal64` to set the pre-amplifier attenuation (niFgen property node: **Calibration»Pre-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION`
   - **value**: `0`
3. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
   - **value**: `2000`

Repeat steps 4 through 7 for each of the eight iterations listed in Table 10, changing the *Post-Amplifier Attenuation* and *Current Configuration* values for each iteration.

Table 10. Attributes and Values for the Main Path Post-Amplifier Gain and Offset

Iteration	Analog Path	Post-Amp Attenuation	Current Configuration
1	NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_0DB
2	NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_12DB
3	NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH	24	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_24DB
4	NIFGEN_VAL_FIXED_LOW_GAIN_ANALOG_PATH	36	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_LOW_GAIN_36DB
5	NIFGEN_VAL_FIXED_HIGH_GAIN_ANALOG_PATH	0	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_HIGH_GAIN_0DB
6	NIFGEN_VAL_FIXED_HIGH_GAIN_ANALOG_PATH	12	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_HIGH_GAIN_12DB
7	NIFGEN_VAL_FIXED_HIGH_GAIN_ANALOG_PATH	24	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_HIGH_GAIN_24DB
8	NIFGEN_VAL_FIXED_HIGH_GAIN_ANALOG_PATH	36	NIFGEN_VAL_CAL_CONFIG_MAIN_PATH_HIGH_GAIN_36DB

4. Call `niFgen_SetAttributeViInt32` to set the analog path value (niFgen property node: **Output Attributes»Analog Path**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_ANALOG_PATH`
    - **value**: The *Analog Path* value for the current iteration from Table 10.
5. Call `niFgen_SetAttributeViReal64` to set the post-amplifier attenuation (niFgen property node: **Calibration»Post-Amplifier Attenuation**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION`
    - **value**: The *Post-Amplifier Attenuation* value for the current iteration from Table 10.
6. Call the following functions and take voltage measurements at the NI 5412 CH 0 front panel connector into a high-impedance load:
    a. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
        - **vi**: The session handle returned from `niFgen_InitExtCal`
        - **channelName**: `"0"`
        - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
        - **value**: `50000`

    b. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage generated by the device. This measurement is `measurement 0`, which is used in step 7.

    e. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
        - **vi**: The session handle returned from `niFgen_InitExtCal`
        - **channelName**: `"0"`
        - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
        - **value**: `15000`

    f. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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 voltage generated by the device. This measurement is `measurement 1`, which is used in step 7.

7. Call `niFgen_CalAdjustMainPathPostAmpGainAndOffset` (niFgen Cal Adjust Main Path Post Amp Gain And Offset VI) 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 10.
   - **mainDACValues**: An array containing two elements—the values (`0`, `0`) that you set on the main DAC.
   - **gainDACValues**: An array containing one element—the value (`2000`) that you set on the gain DAC.
   - **offsetDACValues**: An array containing two elements—the two values (`50000`, `15000`) that you set on the offset DAC in order.
   - **measuredOutputs**: An array containing two elements—the two output voltages (`measurement 0`, `measurement 1`) that you measured in order.

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 tolerance, which is 4.5 ppm. This adjustment ensures the frequency accuracy of the onboard oscillator.

1. Call `niFgen_InitializeOscillatorFrequencyCalibration` (niFgen Initialize Oscillator Frequency Calibration VI) using the following parameter:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
2. Call `niFgen_SetAttributeViReal64` to set the sample rate (niFgen property node: **Arbitrary Waveform Output»Sample Rate**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_ARB_SAMPLE_RATE`
   - **value**: `100000000`
3. Call `niFgen_SetAttributeViReal64` to set the gain (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Gain**) 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 measuring device you use.

4. Call `niFgen_SetAttributeViReal64` to set the offset (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Offset**) 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 measuring device you use.

5. Call `niFgen_SetAttributeViBoolean` to set the digital filter state (niFgen property node: **Output Attributes»Digital Filter Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_DIGITAL_FILTER_ENABLED`
   - **value**: `VI_TRUE`
6. Call `niFgen_SetAttributeViReal64` to set the digital filter interpolation factor (niFgen property node: **Output Attributes» Digital Filter Interpolation Factor**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_DIGITAL_FILTER_INTERPOLATION_FACTOR`
   - **value**: `4`
7. Call `niFgen_SetAttributeViReal64` to set the output impedance (niFgen property node: **Basic Operation»Output Impedance**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE`
   - **value**: `50`
8. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
   - **value**: `VI_TRUE`
9. 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. Because you set the sample rate to 100 MS/s and because there are 10 samples per cycle, the resulting waveform is a 10 MHz sine wave. The sample values of this waveform must fall between -1.0 and 1.0.

10. Call `niFgen_CreateArbWaveform` (niFgen Create Arbitrary Waveform VI) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **wfmSize**: The size in samples (`500`) of the waveform you created in step 9.
    - **wfmData**: The array of waveform samples that you created in step 9.
    - **wfmHandle**: The variable passed by reference through this parameter receives the value (waveform handle) that identifies the waveform created by this function.
11. Call `niFgen_SetAttributeViInt32` to choose the sine waveform (niFgen property node: **Arbitrary Waveform Output»Arbitrary Waveform Handle**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_init`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_ARB_WAVEFORM_HANDLE`
    - **value**: The **wfmHandle** from step 10 (sine waveform handle).
12. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
13. Measure the frequency of the generated waveform. This value is the *Measured Frequency*, which is used in step 14.
14. Repeat steps 14a through 14d 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 5412.

    a. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
        - **vi**: The session handle returned from `niFgen_InitExtCal`

    b. Call `niFgen_CalAdjustOscillatorFrequency` (niFgen Cal Adjust Oscillator Frequency VI) using the following parameters:
        - **vi**: The session handle returned from `niFgen_InitExtCal`
        - **desiredFrequencyInHz**: The desired frequency (`10000000`) of the generated sinusoid in Hz.
        - **measuredFrequencyInHz**: The `measured frequency` of the generated sinusoid in Hz.

    c. Call `niFgen_InitiateGeneration` (niFgen Initiate Generation VI) to initiate the waveform 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*.

15. Call `niFgen_AbortGeneration` (niFgen Abort Generation VI) to abort the waveform generation using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`

Adjusting the Calibration ADC

The NI 5412 has an onboard calibration ADC that is used during self-calibration. Adjusting the calibration ADC involves characterizing the gain and offset associated with this ADC so that performing self-calibration results in an accurately calibrated device.

1. Call `niFgen_InitializeCalADCCalibration` (niFgen Initialize Cal ADC Calibration VI) using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
2. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **value**: `0`
3. Call `niFgen_SetAttributeViInt32` to set the analog path value (niFgen property node: **Output Attributes»Analog Path**) 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. Call `niFgen_SetAttributeViInt32` to set the gain DAC value (niFgen property node: **Calibration»Gain DAC Value**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_GAIN_DAC_VALUE`
    - **value**: `1700`
5. Call `niFgen_SetAttributeViInt32` to set the offset DAC value (niFgen property node: **Calibration»Offset DAC Value**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OFFSET_DAC_VALUE`
    - **value**: `32767`
6. Call `niFgen_SetAttributeViReal64` to set the pre-amplifier attenuation (niFgen property node: **Calibration»Pre-Amplifier Attenuation**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_PRE_AMPLIFIER_ATTENUATION`
    - **value**: `0`

7. Call `niFgen_SetAttributeViReal64` to set the post-amplifier attenuation (niFgen property node: **Calibration»Post-Amplifier Attenuation**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_POST_AMPLIFIER_ATTENUATION`
   - **value**: `0`
8. Call `niFgen_SetAttributeViReal64` to set the output impedance (niFgen property node: **Basic Operation»Output Impedance**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_IMPEDANCE`
   - **value**: `50`
9. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
   - **vi**: The session handle returned from `niFgen_InitExtCal`
   - **channelName**: `"0"`
   - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
   - **value**: `VI_TRUE`
10. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
11. Wait 500 ms for the output to settle.
12. Call `niFgen_SetAttributeViInt32` to set the calibration ADC input (niFgen property node: **Calibration»Cal ADC Input**) 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`
13. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **value**: `27232`

14. Call `niFgen_SetAttributeViBoolean` to disable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_FALSE`
15. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
16. Wait 500 ms for the output to settle.
17. Call `niFgen_ReadCalADC` (niFgen Read CAL ADC VI) to measure the analog output voltage with the onboard calibration ADC using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **numberOfReadsToAverage**: `3`
    - **returnCalibratedValue**: `VI_FALSE`
    - **calADCValue**: 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 31.
18. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_TRUE`
19. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
20. Wait 500 ms for the output to settle.
21. Use the DMM to measure the NI 5412 voltage output directly into the DMM into a high-impedance load. This measurement is `external measurement 0`, which is used in step 31.
22. Call `niFgen_WriteBinary16AnalogStaticValue` (niFgen Write Binary 16 Analog Static Value VI) to set the main DAC value using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **value**: `10232`

23. Call `niFgen_SetAttributeViBoolean` to disable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_FALSE`
24. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
25. Wait 500 ms for the output to settle.
26. Call `niFgen_ReadCalADC` (niFgen Read CAL ADC VI) to measure the analog output voltage with the onboard calibration ADC using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **numberOfReadsToAverage**: `3`
    - **returnCalibratedValue**: `VI_FALSE`
    - **calADCValue**: A ViReal64 variable. The variable passed by reference through this parameter receives the voltage measured by the onboard calibration ADC. This value is `cal ADC measurement 1`, which is used in step 31.
27. Call `niFgen_SetAttributeViBoolean` to enable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_TRUE`
28. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device using the following parameter:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
29. Wait 500 ms for the output to settle.
30. Use the DMM to measure the NI 5412 voltage output directly into the DMM (into a high-impedance load). This measurement is `external measurement 1`, which is used in step 31.
31. Call `niFgen_CalAdjustCalADC` (niFgen Cal Adjust Cal ADC VI) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **voltagesMeasuredExternally**: An array containing two elements—the two voltages (`external measurement 0`, `external measurement 1`) that you measured with the DMM in the order that you measured them.
    - **voltagesMeasuredWithCalADC**: An array containing two elements—the two voltages (`cal ADC measurement 0`, `cal ADC measurement 1`) that you measured with the onboard calibration ADC in the order that you measured them.

32. Call `niFgen_SetAttributeViBoolean` to disable the analog output (niFgen property node: **Basic Operation»Output Enabled**) using the following parameters:
    - **vi**: The session handle returned from `niFgen_InitExtCal`
    - **channelName**: `"0"`
    - **attributeID**: `NIFGEN_ATTR_OUTPUT_ENABLED`
    - **value**: `VI_FALSE`
33. Call `niFgen_SetAttributeViInt32` to set the calibration ADC input (niFgen property node: **Calibration»Cal ADC Input**) 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`
34. Call `niFgen_Commit` (niFgen Commit VI) to commit the attribute values to the device 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.

Call `niFgen_CloseExtCal` (niFgen Close Ext Cal VI) 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, 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 discards the new calibration constants and does not change any of the calibration data stored in the onboard EEPROM.

Calibration Utilities

NI-FGEN supports several calibration utilities that allow you to retrieve information about adjustments performed on the NI 5412, restore an external calibration, change the external calibration password, and store small amounts of information in the onboard EEPROM. You can retrieve some data using MAX or the FGEN SFP; however, you can retrieve all the data using NI-FGEN.

MAX

To retrieve data using MAX, complete the following steps:

1. Launch MAX.
2. Select the device from which you want to retrieve information from **My System»Devices and Interfaces»PXI System**.
3. Select the **Calibration Tab** on the lower right corner.

   You should see information about the last 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. Select the device from which you want to retrieve information using the Device Configuration dialog box (**Edit»Device Configuration**).
3. Open the Calibration dialog box (**Utility»Calibration**).

   You should see information about the last dates of both external and self-calibration.

NI-FGEN

NI-FGEN provides a full complement of calibration utility functions and VIs. Refer to the *NI Signal Generators Help* for the complete function reference, including the following utility functions:

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

Where to Go for Support

The National Instruments website 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.

Visit ni.com/services for NI Factory Installation Services, repairs, extended warranty, and other services.

Visit ni.com/register to register your National Instruments product. Product registration facilitates technical support and ensures that you receive important information updates from NI.

A Declaration of Conformity (DoC) is our claim of compliance with the Council of the European Communities using the manufacturer's declaration of conformity. This system affords the user protection for electromagnetic compatibility (EMC) and product safety. You can obtain the DoC for your product by visiting ni.com/certification. If your product supports calibration, you can obtain the calibration certificate for your product at ni.com/calibration.

National Instruments corporate headquarters is located at 11500 North Mopac Expressway, Austin, Texas, 78759-3504. National Instruments also has offices located around the world. For telephone support in the United States, create your service request at ni.com/support or dial 1 866 ASK MYNI (275 6964). For telephone support outside the United States, visit the Worldwide Offices section of ni.com/niglobal to access the branch office websites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

371461D-01 Oct15