Computer-Based Instruments

NI-DSA Software User Manual

Version 1.2

September 2001 Edition
Part Number 370380A-01

Worldwide Technical Support and Product Information

`ni.com`

National Instruments Corporate Headquarters

11500 North Mopac Expressway   Austin, Texas 78759-3504   USA   Tel: 512 683 0100

Worldwide Offices

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 284 5011,
Canada (Calgary) 403 274 9391, Canada (Montreal) 514 288 5722, Canada (Ottawa) 613 233 5949,
Canada (Québec) 514 694 8521, Canada (Toronto) 905 785 0085, China (Shanghai) 021 6555 7838,
China (ShenZhen) 0755 3904939, Czech Republic 02 2423 5774, Denmark 45 76 26 00, Finland 09 725 725 11,
France 01 48 14 24 24, Germany 089 741 31 30, Greece 30 1 42 96 427, Hong Kong 2645 3186,
India 91805275406, Israel 03 6120092, Italy 02 413091, Japan 03 5472 2970, Korea 02 596 7456,
Malaysia 603 9596711, Mexico 001 800 010 0793, Netherlands 0348 433466, New Zealand 09 914 0488,
Norway 32 27 73 00, Poland 0 22 528 94 06, Portugal 351 1 726 9011, Russia 095 2387139,
Singapore 2265886, Slovenia 386 3 425 4200, South Africa 11 805 8197, Spain 91 640 0085,
Sweden 08 587 895 00, Switzerland 056 200 51 51, Taiwan 02 2528 7227, United Kingdom 01635 523545

For further support information, see the *Technical Support Resources* appendix. To comment on the documentation, send e-mail to `techpubs@ni.com`.

Important Information

Conventions

The following conventions are used in this manual:

<>	Angle brackets that contain numbers separated by an ellipsis represent a range of values associated with a bit or signal name—for example, DBIO<3..0>.
[ ]	Square brackets enclose optional items—for example, [`response`].
»	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.
♦	The ♦ symbol indicates that the following text applies only to a specific product, a specific operating system, or a specific software version.
💡	This icon denotes a tip, which alerts you to advisory information.
📝	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.
bold	Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. Bold text also denotes parameter names.
ICP	ICP® is an abbreviation for Integrated Circuit Piezoelectric. ICP-type products operate using a constant current source and return the output signal in the form of voltage modulation on the same line as the current source.
italic	Italic text denotes variables, emphasis, a cross reference, or an introduction to a key concept. This font 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. This font is also used for the proper names of disk drives, paths, directories, programs, device names, functions, variables, filenames and extensions. This manual also uses this font as a naming convention to

jointly refer to LabVIEW VIs and C-language function calls. For example, `Read Measurement`, when shown in this font, refers both to the LabVIEW NI-DSA Read Measurements VI and to the C function `NIDSA_read_measurement`.

Contents

Chapter 5
Octave Analysis (Add-On) Mode Programming

Chapter 6
Advanced Concepts

1 Introduction to NI-DSA

Thank you for buying a National Instruments Dynamic Signal Analyzer (DSA), which includes NI-DSA, National Instruments' driver for its DSA devices. With NI-DSA, you can program your NI 4551 or NI 4552 to analyze time- and frequency-domain data onboard in real time. This manual shows you how to use your application development environment (ADE) with NI-DSA to program your DSA instrument.

Getting Started with NI-DSA

1. Install your ADE, the NI-DSA driver, and your NI 45*XX*. For installation instructions, refer to *Where to Start with Your NI 45*XX *for PCI Dynamic Signal Analyzer*, which came with your hardware.

Note Read the `Readme.htm` file that was installed with the instrument driver for last-minute changes and updates.

2. Choose the *measurement mode* that is best for your application. For more information about NI-DSA measurement modes, refer to the *NI-DSA Measurement Modes* section.
3. Begin programming your DSA instrument. Detailed programming instructions and examples for each of the measurement modes as shown in this list:
   - For basic NI-DSA programming instructions common to all NI-DSA applications, refer to Chapter 2, *Programming with NI-DSA*.
   - For specific FFT mode programming instructions and examples, refer to Chapter 3, *FFT Mode Programming*.
   - For programming examples and instructions for swept-sine mode and source mode, refer to Chapter 4, *Swept-Sine Mode Programming*.
   - For instructions and examples of octave and level analyzer mode programming, refer to Chapter 5, *Octave Analysis (Add-On) Mode Programming*.

Related Documentation

- National Instruments Application Note 41, *The Fundamentals of FFT-Based Signal Analysis and Measurement*
- The *NIDSA Help for C Programmers* and the *NIDSA Help for Basic Programmers* have detailed information about the NI-DSA functions and parameters. If you have installed NI-DSA, the default location of these help files is **Start»Programs»National Instruments»Ni-dsa**.
- Help for LabVIEW VIs is available in the LabVIEW context help.
- To learn about the electrical and mechanical aspects and features of your National Instruments DSA hardware, refer to the *NI 4551/4552 User Manual*.
- For the latest versions of drivers, manuals, and example programs, visit `ni.com/instruments` for free downloads.

NI-DSA Measurement Modes

NI-DSA has three measurement modes, including:

- FFT analyzer mode
- Swept-sine analyzer mode
- Octave analyzer mode (Real-Time Octave Analyzer add-on)

In addition to the measurement modes, NI-DSA has a source mode for use with only the NI 4551.

The mode you use depends on the measurements you want to take. The following sections describe the modes and their specific features.

Note The Real-Time Octave Analyzer (RTOA) software is available as an add-on for NI-DSA. If you have not obtained and installed the RTOA, you cannot perform the octave and level measurements described in this manual.

FFT Analyzer Mode

Use FFT analyzer mode for the following operations:

- Real-time FFT—analyze two baseband spans and two zoom spans, or four baseband spans
- Auto-power spectrum
- Frequency response
- Cross-power spectrum

- Coherence
- Power spectral density
- Bandpower
- Harmonic analysis—THD, THD+Noise, and SINAD
- Time-domain data acquisition

Swept-Sine Analyzer Mode (NI 4551 Only)

Use swept-sine analyzer mode to make frequency response measurements when you need more accuracy, control, and dynamic range than FFT-based frequency response measurements provide.

Note Swept-sine analyzer mode requires a signal source, so this mode is supported only on the NI 4551 for PCI.

A swept-sine analyzer measures the response of a system one frequency at a time, based on a sequence of frequency points that you can specify. At each point, the source (an analog output channel) generates a sine wave at a constant frequency. The input channels measure only at this frequency. After going through the specified sequence of frequency points, the measurements are complete. In addition, the auto-ranging, auto-level, and auto-resolution features of NI-DSA can automatically adjust the input gain, output amplitude, and step size to optimize the measurements for a particular device under test.

Octave Analyzer Mode (Optional)

Use octave analyzer mode to analyze the frequency content of signals in a way that is analogous to human hearing. Octave analysis is required by many government standards. Features of the octave analyzer mode include:

- Full-, 1/3-, and 1/12-octave measurements
- 25,600 or 51,200 Samples/sec
- A-, B- or C-weighting
- IEC/ANSI compliant
- Averaging modes
  - Linear
  - Exponential
    - Slow
    - Fast

        - Impulse
        - Custom
    - Equal confidence
    - Peak-hold
- Level measurements:
    - Equivalent continuous level
    - Slow
    - Fast
    - Impulse
    - Impulse Eq
    - Peak
    - Custom

In the optional octave analyzer mode, the data are passed through a parallel bank of filters, averaged if needed, then transferred to the host memory.

Source Mode (NI 4551 Only)

With your NI 4551 and NI-DSA in source mode, you can generate signals of the following types:

- Sine (up to two tones)
- Chirp
- Noise—PRN, white noise, pink noise, and band-limited noise
- Arbitrary—the NI 4551 generates an output based on the data (up to 4,096 samples) you place in the output buffer

2 Programming with NI-DSA

This chapter provides instructions for programming with NI-DSA and specific programming examples, which you can modify for your application.

Note This manual uses a naming convention to jointly refer to LabVIEW VIs and C-language function calls. For example, `Read Measurement`, when shown in this font, refers both to the LabVIEW NI-DSA Read Measurements VI and to the C function `NIDSA_read_measurement`.

Basic Concepts

In general, programs built with NI-DSA always perform the same basic types of operations. In this section, the operations are described as steps in general terms. In later sections, specific instructions for individual operating modes are given.

Step 1—Initialize. Establishes communication with the DSA instrument and set its power-on state.

Step 2—Configure. Selects the measurement mode and sets up the instrument to perform specific operations.

Step 3—Control. Starts, stops, and checks the status of instrument operations.

Step 4—Read. Transfers data from the instrument to the host computer.

Step 5—Close. Terminates communication between the software and the instrument and deallocates system resources.

In addition to these operations, there are operations performed using utility functions. Utility functions perform auxiliary operations. Examples of Utility functions include:

- `Reset`
- `Self Test`

- `Revision Query`
- `Error Query`
- `Error Message`

Notes for C Programmers

The constants used in the example programs are defined in the NI-DSA header file, `nidsa.h`. The C code snippets in this manual use these constants for clarity.

VISA data types, defined in `visatype.h`, are also used.

Step 1—Initialize

Establishes a connection between your instrument and your application by using `Initialize`. `Initialize` returns a session number, which your application uses as a handle to communicate other function calls to the instrument. You can also use `Initialize` to query the device name or to reset the instrument.

LabVIEW Example

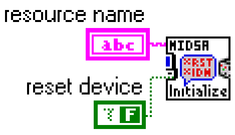

Figure 2-1. Initialize

C Example

```
NIDSA_init (“DAQ::1”, VI_TRUE, VI_FALSE, &vi);
```

Notes on Examples

The resource name is the device number assigned by Measurement & Automation Explorer (MAX). If you have only one DSA instrument installed in your system, the device number is Device 1 by default.

If you have multiple DSA instruments installed, launch MAX and open **Devices and Interfaces** to find your device number.

DSASession is a handle returned by `Initialize`.

Step 2—Configure

Use `Configure DSA Mode` to download the code for FFT, swept-sine, or octave analyzer mode to the digital signal processor (DSP). Then configure the analog front end with the following functions:

- `Set Input Voltage Range`—from ±10 mV to ±42 V in 10 dB increments. To let the instrument automatically select the best range, use `Set Input Auto Range`.
- `Set Input Coupling`—select DC or AC coupling for inputs.

Caution If you are using ICP-type sensors, select AC coupling to protect your instrument from potential damage. ICP-type sensors can cause large DC-offset voltages to occur on the signal inputs.

- `Configure Trigger`—sets the parameters for analog triggering.

Refer to the *NI 4551/4552 User Manual*, Chapter 3, *Hardware Overview* for more information about configuring your analog input channels.

Note Unless otherwise noted, you can use the functions in this section, and most NI-DSA functions, to apply parameters to all input channels, or to apply different parameters to one or any combination of input channels independently of the others.

LabVIEW Example

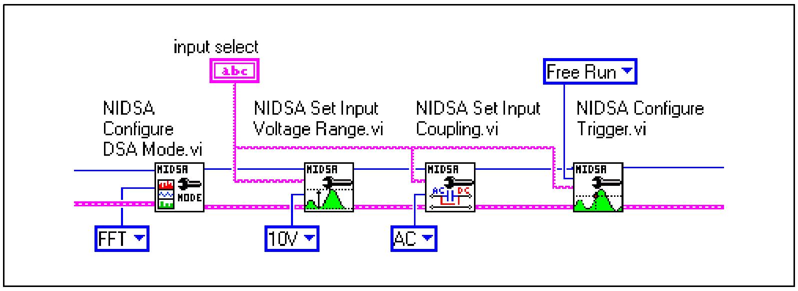
input select
abc
Free Run
NIDSA Configure DSA Mode.vi
NIDSA Set Input Voltage Range.vi
NIDSA Set Input Coupling.vi
NIDSA Configure Trigger.vi
NIDSA
MODE
FFT
10V
AC

Figure 2-2. Configure

C Example

```
//Configure Hardware settings
//Put hardware in FFT mode
NIDSA_configure_dsa_mode (DSASession, FFT_MODE);
//Set all channels for input range of 10 V
NIDSA_set_input_voltage_range (DSASession, "-1", VRANGE_10V);
NIDSA_set_input_coupling (DSASession, "-1", COUP_AC);
NIDSA_configure_trigger (DSASession, FREERUN, TriggerChannel, Level,
Hysteresis, Slope, Delay);
```

Notes on Examples

By default, NI-DSA returns all measurements in volts, but you can have NI-DSA return measurements in the engineering units (EU) that you choose. `Configure EU`, used with the other `Configure EU` functions, lets you choose the units for your measurements and scale them accordingly. More information about using engineering units, refer to Chapter 3, *FFT Mode Programming*, the *Using Engineering Units* section.

You have now configured your hardware for the input signal, and are now ready to configure your measurements.

- To configure FFT measurements, refer to Chapter 3, *FFT Mode Programming*.
- To use NI-DSA for acquiring time-domain signals, refer to Chapter 3, *FFT Mode Programming*, the *Using Capture Mode to Acquire Time-Domain Data* section.
- If you are using an NI 4551 to generate a signal, refer to Chapter 4, *Swept-Sine Mode Programming*, the *Source Mode (NI 4551 Only)* section.
- For swept-sine measurements, refer to Chapter 4, *Swept-Sine Mode Programming*, the *Step 2—Configure the Swept-Sine Analyzer* section.
- For octave and level measurements, refer to Chapter 5, *Octave Analysis (Add-On) Mode Programming*.

Step 3—Read

Read functions are different for each measurement mode. For mode-specific programming instructions, refer to Chapter 3, *FFT Mode Programming*, Chapter 4, *Swept-Sine Mode Programming*, and Chapter 5, *Octave Analysis (Add-On) Mode Programming*.

Step 4—Control

Control functions are different for each measurement mode, but generally start, stop, or change the operation of your DSA instrument. For example, you might call `Restart OLM Averaging` to restart averaging for a particular analyzer channel. Refer to your online help for more information about control functions.

Step 5—Close

Any application you write with NI-DSA should have a close function.

LabVIEW Example

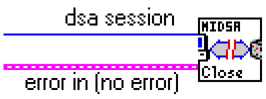

Figure 2-3. Close

C Example

```
NIDSA_close (DSAsession);
```

3 FFT Mode Programming

Figure 3-1 shows the recommended program flow for baseband FFT analysis. Figure 3-2 shows the recommended program flow for zoom FFT analysis. For a detailed discussion of FFT analysis, refer to the *Step 2—Configure the FFT Analyzer* section.

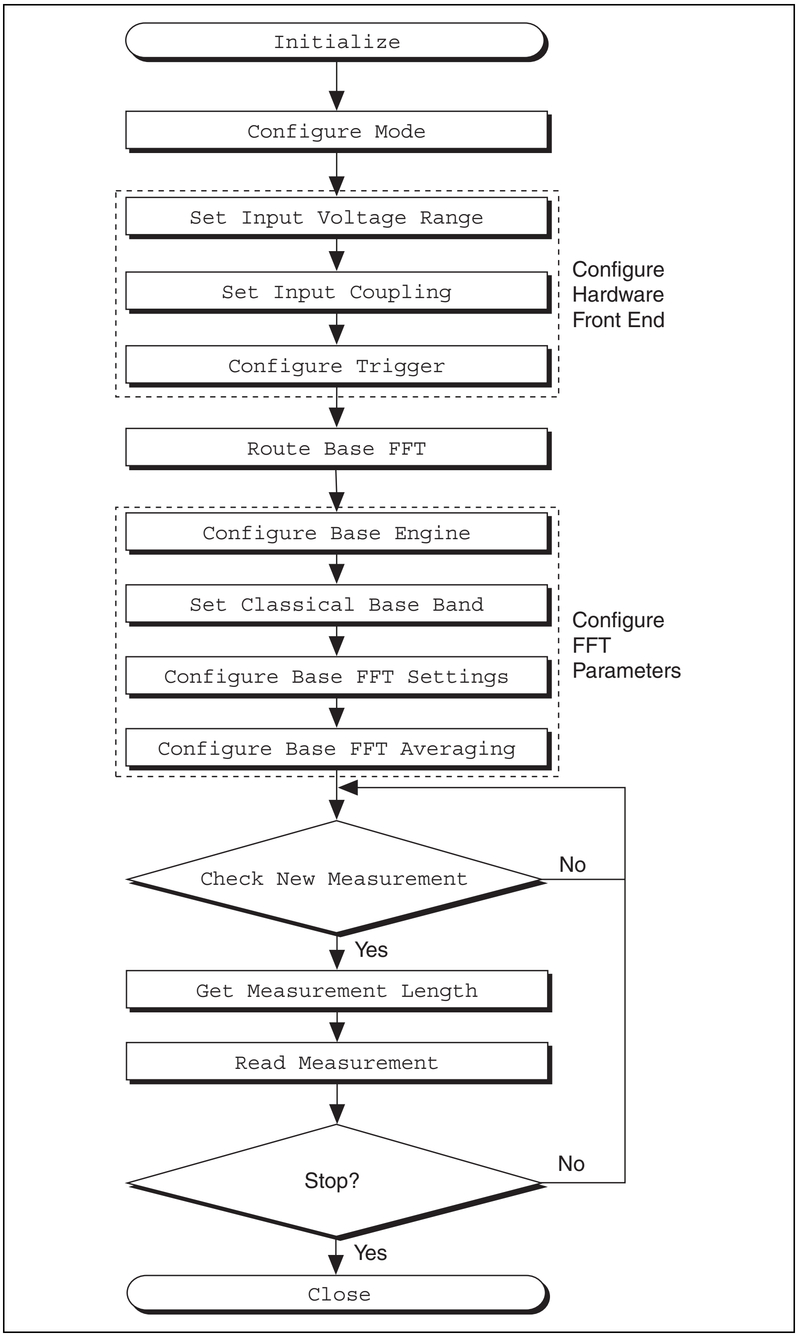

Figure 3-1. Baseband FFT Programming Flowchart

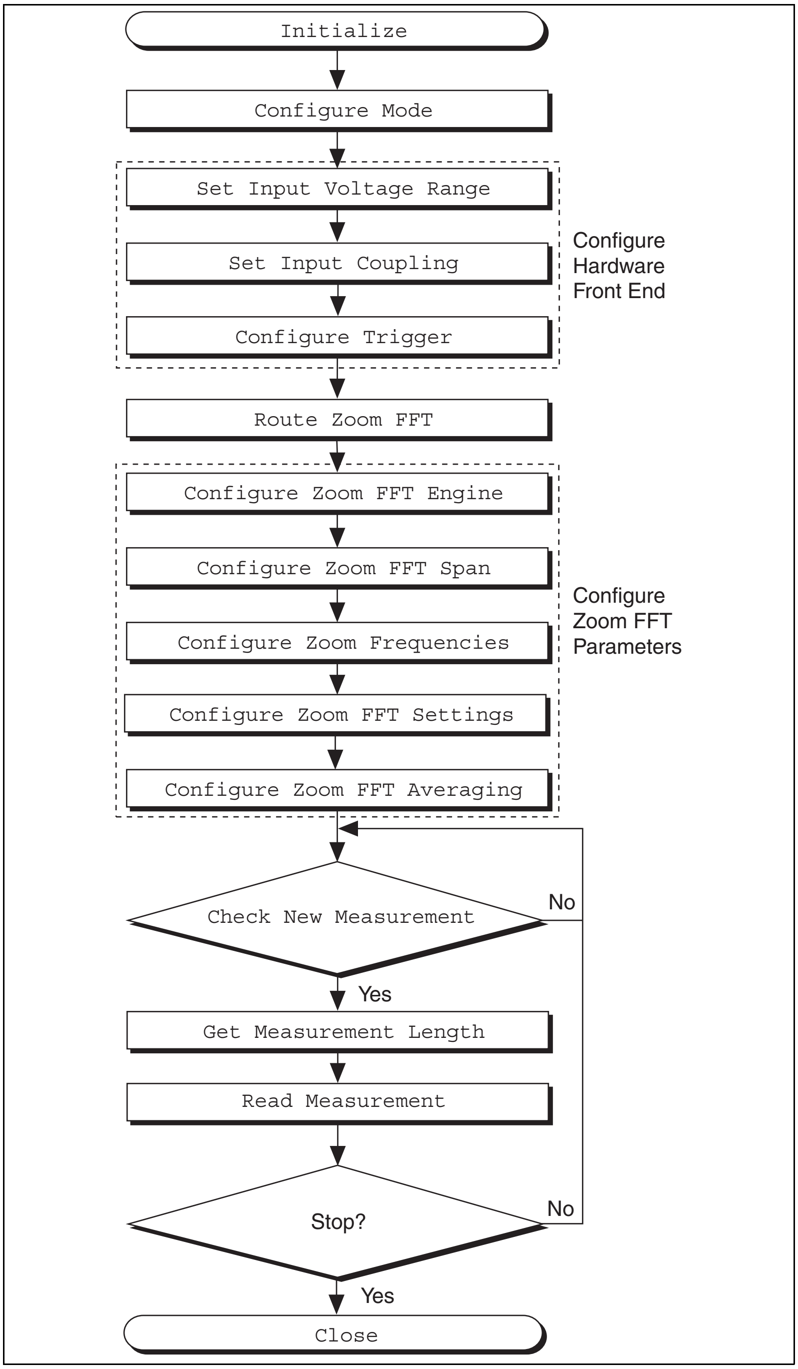

Figure 3-2. Zoom FFT Programming Flowchart

Step 2—Configure the FFT Analyzer

The first step in configuring your FFT measurement is to route the digitized input signal to one or more analyzers using `Route Base` and `Route Zoom`. To configure a baseband FFT, use the following:

- `Configure Base FFT Engine`—sets the type of measurements the analyzer makes, using the following:
  - **Time**—acquires a time waveform
  - **FFT**—acquires a time waveform, then performs FFT
  - **Auto**—acquires time waveform, performs FFT and auto-power measurement
  - **Cross**—acquires time waveform, performs FFT, auto-power, and cross-power measurements
- `Set Classical Baseband Span`—sets the classical baseband span
- `Configure Base FFT Settings`—set the size of the time record, apply a window, sets time increment and phase suppression
- `Configure Base FFT Averaging`—sets the averaging mode, weighting, number, and overload rejection

To set up a zoom FFT, use the following:

- `Configure Zoom FFT Engine`—sets the type of measurements the analyzer makes, using the same parameters as `Configure Base FFT Engine`
- `Configure Zoom FFT Span`—sets the zoom span and the lock-to frequency
- `Configure Zoom Frequencies`—set the start, center, and end frequencies
- `Configure Zoom FFT Settings`—sets the size of the time record, apply a window, set time increment and phase suppression
- `Configure Zoom FFT Averaging`—sets the averaging mode, weighting, number, and overload rejection

LabVIEW Examples

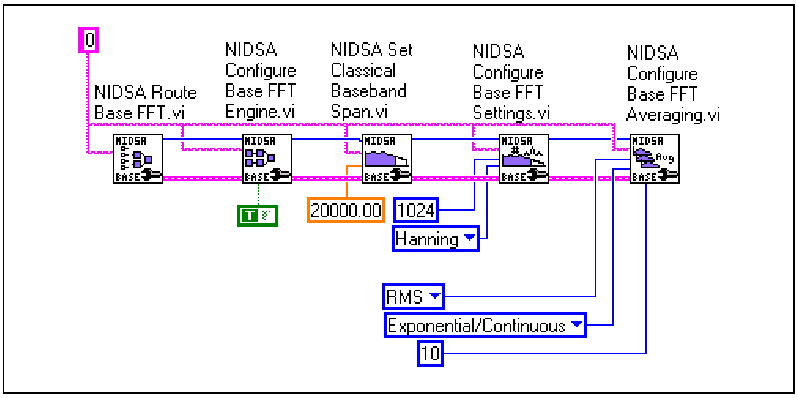
0
NIDSA Route Base FFT.vi
NIDSA Configure Base FFT Engine.vi
NIDSA Set Classical Baseband Span.vi
NIDSA Configure Base FFT Settings.vi
NIDSA Configure Base FFT Averaging.vi
20000.00
1024
Hanning
RMS
Exponential/Continuous
10

Figure 3-3. Configuring Baseband FFT

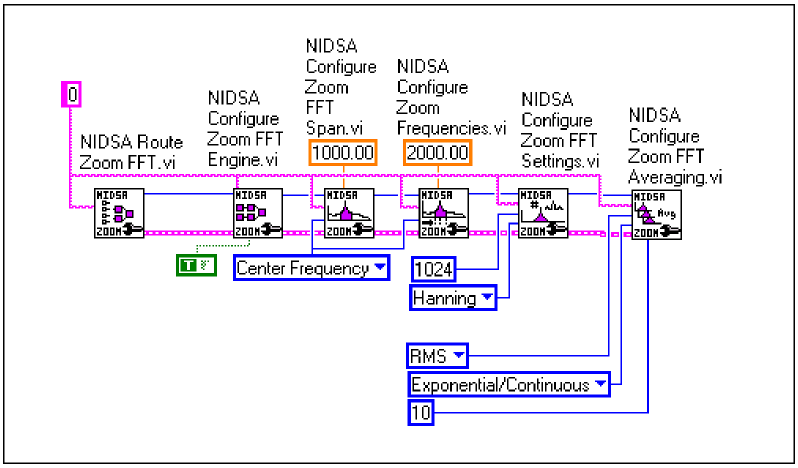
0
NIDSA Route Zoom FFT.vi
NIDSA Configure Zoom FFT Engine.vi
NIDSA Configure Zoom FFT Span.vi
1000.00
NIDSA Configure Zoom Frequencies.vi
2000.00
NIDSA Configure Zoom FFT Settings.vi
NIDSA Configure Zoom FFT Averaging.vi
Center Frequency
1024
Hanning
RMS
Exponential/Continuous
10

Figure 3-4. Configuring Zoom FFT

C Examples

Examples for baseband FFT and zoom FFT are included.

Baseband FFT Example

```
//Configure Baseband FFT Analyzer

//Route channel i to analyzer 0

NIDSA_route_base_fft (DSASession, "0", InputChannel);
```

```
NIDSA_configure_base_fft_engine (gDSASession, "0", VI_FALSE, VI_TRUE, 
VI_FALSE, VI_FALSE);

//Set the frequency span for Baseband FFT analyzer

NIDSA_set_classical_baseband_span (DSASession, "0", BaseBandSpan);

//Set FFT resolution and windowing

NIDSA_configure_base_fft_settings (DSASession, "0", FFTSize, FFTWindow, 100, 
0.0);

//Configure averaging parameters

NIDSA_configure_base_fft_averaging (gDSASession, "0", AvgMode,AvgWeighting, 
NumberOfAvg,VI_FALSE);
```

Zoom FFT Example

```
//Configuring Zoom FFT analyzer

NIDSA_configure_zoom_fft_engine (gDSASession, CHANNEL, 0, 0, 0, 1);

NIDSA_configure_zoom_fft_span (gDSASession, CHANNEL, 1,zoomSpan);

NIDSA_configure_zoom_frequencies (gDSASession, CHANNEL, 1,centerFrequency);

NIDSA_configure_zoom_fft_settings (gDSASession, 
CHANNEL,*(number_of_frequency_bins+lines),window, timeIncr, phaseSuppress);

NIDSA_configure_zoom_fft_averaging (gDSASession, CHANNEL, AvgMode, 
weightingMode, numberOfAvg, overloadReject);
```

Notes on Examples

Routing

You can route signals to two base analyzers and two zoom analyzers with `Route Base FFT` and `Route Zoom FFT`. Each analyzer operates independently of the others, and any input can be routed to any analyzer or combination of analyzers. For example, you can route one input to all four analyzers and perform four different analyses on the same input.

Note If you configure your instrument to perform a 4-channel FFT, `Route Zoom` generates an error. 4-channel FFTs can be performed in baseband only.

Configuring the FFT Engine

Each measurement type parameter of `Configure Base FFT Engine` automatically sets the preceding parameter. The **time** parameter sets the analyzer to simply acquire a waveform. The **FFT** parameter automatically sets the **time** parameter since an FFT cannot be performed without a waveform. Setting **auto** sets the **time** and **FFT** parameters, and those functions are performed, as well as measuring the auto-power spectrum. Setting **cross** sets all of these parameters.

When you set the **cross** parameter, the two baseband channels are *linked* to ensure that the settings for each channel are the same. Setting changes made to channel 0 causes the channel 1 settings to be changed to match channel 0.

- **Time**—Only Time domain information is available
- **FFT**—FFT (Mag + phase info is available)
- **Auto**—Auto-power spectrum (Phase information is lost)
- **Cross**—Cross-power spectrum (FRF measurements)

Note Cross-power spectrum measurements require much more processing than time-domain acquisitions. For the best performance, be sure you choose the mode that provides only the information you need in real time, and reserve other measurements for your computer.

Tip Setting the **Channel** parameter to `-1` selects all channels on your DSA device.

Setting the Span

`Set Classical Baseband Span` can only be used to specify a classical span, one that corresponds to 400 lines for a 1,024-point FFT. You can set the **span** parameter to any value from 1,953.125 to 80,000.000 Hz, but it is coerced to the nearest classical span. Use `Set Baseband Span` to perform an extended FFT.

Note Changing the baseband span of any base channel causes the sampling rate to change, affecting the span of all other channels.

Classical spans are as follows:

- Up to 800 lines in 0–80 KHz span for baseband
- Up to 1,600 line in 0–80 KHz span for zoom analyzer

Acceptable extended spans are:

- Up to 800 lines in 0–95 KHz span for baseband
- Up to 1,600 line in 0–95 KHz span for zoom analyzer

Note NI-DSA automatically sets the hardware sampling rate to optimize antialiasing and real-time performance, based on the span you select.

Configuring the FFT Settings

- Size of the buffer is directly related to the FFT resolution (1,024 pts = 400 lines for baseband/1,024 points = 800 lines for zoom)
- Window—Window to apply to the time domain signal (applied to reduce spectral leakage)
- Time increment—Related to overlapping (overlapping = 100 – time increment)
- Phase suppression— Phase suppression sets the amplitude threshold (magnitude squared) above which the phase of a particular frequency component is computed. This feature is used to disregard the phase information for frequency components with negligible amplitudes (i.e. noise). Setting the value of Phase Suppress equal to 0.0 causes the phase to be computed for all frequency components, no matter how small their magnitude. The value assigned to Phase Suppress must have units of voltage squared. For example, to configure phase suppression for all frequency components with amplitudes less than –40 dBV (0.01V), set Phase Suppress to (0.01 V)2 = 0.0001 V2.

Averaging

Averaging successive measurements tends to improve the accuracy of your measurements. Use `Configure Base FFT Averaging` or `Configure Zoom FFT Averaging` to set the following:

- **Mode**
  - Vector
  - RMS
  - Peak Hold
- **Weighting**
  - Linear
  - Exponential
- **Number of averages**

Step 3—Read FFT Measurements

Before you can read a measurement, you need to know if it has been completed. `Check New Measurement` returns a `1` if a new measurement is ready. You must call `Check New Measurement` before reading the first measurement and before reading any subsequent measurement set.

Reading a New Measurement

To read a new measurement, use the following functions:

- `Get Measurement Length`—returns the number of data points to read, based on the type and length of the measurement, and the set of frequency lines you choose with **freqLineSelector**. Variables for **freqLineSelector** are:
  - Valid lines—returns alias-free lines only
  - All lines—returns all FFT lines
  - Classical lines—returns alias-free classical lines
- `Read Measurement`—reads the measurement from the specified analyzer channel

The valid measurement **type** settings for `Get Measurement Length` and `Read Measurement` are limited to the measurements you specified with `Configure Base FFT Engine`/`Configure Zoom FFT Engine` and `Configure Base FFT Settings`/`Configure Zoom FFT settings`. Each FFT analyzer measurement type and the valid **type** parameters for reading that measurement type is shown in the following list:

- Time measurement
  - Time 0—reads time waveform on analyzer 0
  - Time 1—reads time waveform on analyzer 1
  - Windowed time 0—reads windowed time waveform on analyzer 0
  - Windowed time 1—reads windowed time waveform on analyzer 1
- FFT measurement
  - Base FFT 0—reads baseband FFT on analyzer 0
  - Base FFT 1—reads baseband FFT on analyzer 1
- Auto-power measurement
  - Base auto-power 0—reads baseband auto-power from analyzer 0
  - Base auto-power 1—reads baseband auto-power from analyzer 1

- Cross-power measurement
  - Base cross-power
  - Base freq response
  - Base coherence
  - Base coherent power
  - Zoom FFT 0
  - Zoom FFT 1
  - Zoom auto-power 0
  - Zoom auto-power 1
  - Zoom cross-power
  - Zoom freq response
  - Zoom coherence
  - Zoom coherent power

Additional read measurement parameters that you must set are:

- **startIndex**—starts returning data from this data point, usually 0, to return the entire measurement
- **Length**—number of data points to return
- **View**—which part of the selected measurement to return, from the following choices:
  - Real part
  - Imaginary part
  - Magnitude
  - Magnitude squared
  - Phase
  - Unwrapped phase
- **dbUnits**
  - dB off (linear units)
  - dB on
  - dBm on
- **pkrmsUnits**
  - Peak
  - RMS
  - Peak-to-peak

- **phaseUnits**
  - Degrees
  - Radians
- **x0**—x-axis value of the first data point returned in the y-axis measurement array
- **dx**—x-axis increment for data points returned in the y-axis measurement array

Tip Keep in mind that the units for **x0** and **dx** parameters are seconds for time-domain measurements and hertz for FFTs.

♦ NI 4552

If you configure an NI 4552 for 4-channel time, FFT, or auto-power measurements, you may also use these **type** parameters:

- Time 2
- Time 3
- Windowed time 2
- Windowed time 3
- Base FFT 2
- Base FFT 3
- Base auto-power 2
- Base auto-power 3

LabVIEW Example

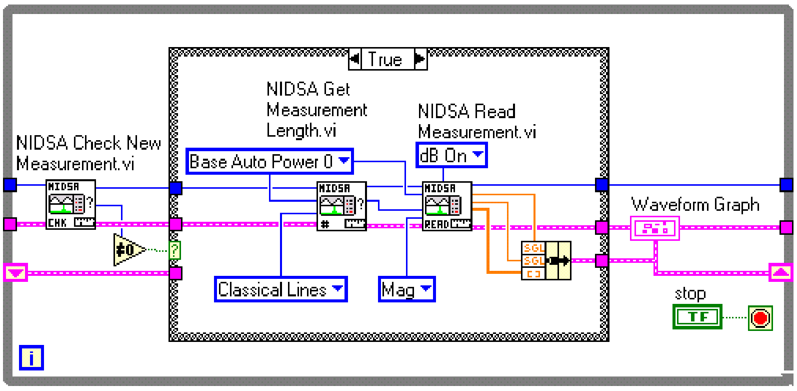
True
NIDSA Get Measurement Length.vi
NIDSA Read Measurement.vi
NIDSA Check New Measurement.vi
Base Auto Power 0
dB On
Classical Lines
Mag
Waveform Graph
stop

Figure 3-5. Read FFT Measurements

C Example

When reading FFT measurements, ideally, you need to use a timer in order to read measurements periodically.

```
//This Function is called periodically (timer tick)

int CVICALLBACK TimerCB (int panel, int control, int event,void
*callbackData, int eventData1, int eventData2){

ViStatus NewMeasurement;
ViInt32 Length;
ViInt32 RealOrComplex;
ViReal32 x0;
ViReal32 dx;
ViReal32 ActualFS;

switch (event){
                    case EVENT_TIMER_TICK:
                    //Check if a new measurement is ready
                    NIDSA_check_new_measurement (DSASession,
                    &NewMeasurement);
                    if (!NewMeasurement)
                    return 0;
                    //Read the new measurement
                    NIDSA_get_measurement_length (DSASession,
                    BaseFFT0,FREQ_LINES_CLASSICAL, &Length,
                    &RealOrComplex);


                    //Read the measurement performed by Analyzer BaseFFT0 /
                    Get Magnitude in dB / Use RMS units and Degrees
                    NIDSA_read_measurement (DSASession, BaseFFT0,0, Length,
                    MAGVIEW, DBON,RMSUNIT, DEGREES, &x0,
                    &dx,gFFTMeasurement);
                    break;
                    }
}
```

Step 4—Control

Checks the status of ongoing measurement and error handling.

Status functions return the status of the instrument, such as whether or not an overload has occurred at the input, whether the instrument is operating in real-time, or if the time that has elapsed since averaging was begun. (Example: `Get OLM Status`)

`Check Status`—returns status values for up to seven conditions. The primary status indicators are as follows:

- Real time—indicates whether the onboard DSP can keep up with the rate of data acquisition
- Main error—indicates an error on the DSA device
- Input overload status—indicates an overload condition on an input channel. The input level exceeded the maximum voltage allowed.
- Wait on trigger—indicates that the DSA instrument is waiting for a trigger condition to be met. Refer to the *NI 4551/4552 User Manual*, Chapter 3, *Hardware Overview* for more information about triggering.
- Linear averaging status
- Get base FFT average complete
- Get zoom FFT average complete
- Error handling

- Error message
- Reset

Using Capture Mode to Acquire Time-Domain Data

When you are using the FFT mode of NI-DSA, you can obtain time-domain waveforms from `Read Measurement`. This function returns time-domain blocks whose size scales with the number of FFT lines you are calculating. For instance, a 400-line classical, or 475-line extended FFT, corresponds to 1,024 time domain data points.

For many applications, this type of acquisition is sufficient. However, if your FFT analyzer falls out of real time, you lose one or more of these time-domain blocks. In other words, the time-domain data set contains a gap. Capture mode provides an alternative method that guarantees gapless time-domain data. Capture mode is most often used to stream data to disk.

The API for capture mode is quite similar to the standard NI-DAQ operations in LabVIEW. When using capture mode, call `Configure Capture Buffer` before the acquisition begins. This call sets aside a buffer space in RAM to hold time domain data as the DSA device acquires it. If you are acquiring data from a known length of time that does not exceed a few seconds, you can use the one-shot capture option. If you need to acquire more than a few hundred thousand data points, or do not know the acquisition time in advance, perform a continuous capture. The continuous mode allocates a circular buffer.

After you have configured all the analysis parameters for the DSA device, call `Control Capture`, setting **Capture Mode Start**. At this point, the RAM buffer you set aside begins filling with data.

To pull data out of this buffer into you application, call `Read Capture Single Chan` or `Read Capture Multi Chan`. If you are performing a continuous capture operation, it is important to read this data from the buffer periodically to prevent the buffer from overflowing. If you fail to read the data often enough and the number of points in buffer exceeds its capacity, NI-DSA returns an error.

NI-DSA ships with a LabVIEW example program that illustrates how to use capture mode to stream gap-free time-domain data to a file for offline processing.

Using Engineering Units

Measurements made with NI-DSA are expressed in volts (V) by default. In many industries or applications, it is more appropriate to express measurements in other units. Sound pressure, for example, is usually measured in pascals (Pa). The NI-DSA engineering units functions (those with `EU` in their names) let you configure, read, and display measurements in eight standard units including Pa, g, and m/s^2, or in any custom unit you specify.

Configure Engineering Units

- Configure EU—**EU Mode** enables or disables engineering units for selected channels
- `Configure EU Label`—selects the label according to the type of sensor you are using (to match the units you are using)
  - **EU Label Select**—selects V, Pa, g, m/s, in/s, m/s^2, in/s^2, or a custom label
  - **EU Label String**—if **EU Label Select** = custom, can contain the label text, for example: f/s
- `Configure EU Scale`
  - **EU scale**—sets the scale factor for EU Scale Type, depending on the sensitivity of your sensor
  - **EU scale type**—V/EU, EU/V, or dB (EU/V)
- `Configure EU dB Reference`—enters the reference level for your measurements

For example, assume you are using a microphone as your transducer to measure sound pressure, which is usually expressed in Pa. The sensitivity of the microphone is 20 mV/Pa. You call:

- `Configure EU Label` to set the label to Pa
- `Configure EU Scale` to set the scale factor to 0.020 (20 mV = 0.020 V) and the EU scale to V/EU.

With these settings, a 1 V signal from the microphone is presented as a sound pressure of 94 dB, so you set the dB reference to 20 µPa (0.00002 Pa).

LabVIEW Example

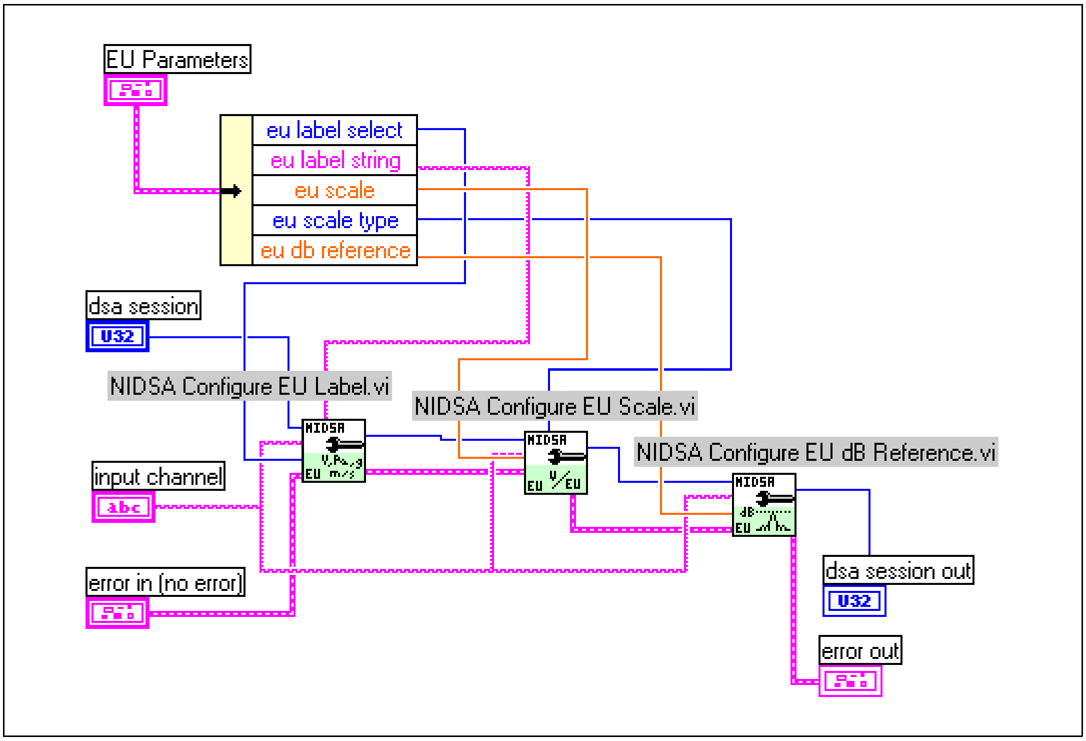

Figure 3-6. Configuring Engineering Units

C Example

```
//Configuring engineering units
//A microphone is connected on Channel 0 (Units = Pa)
NIDSA_configure_eu_label (DSASession, "0", EU_LABEL_PA, "EU");

//Microphone sensitivity = 100 mV/Pa
NIDSA_configure_eu_scale (DSASession, "0", 0.1, EU_SCALE_V_EU);

//Reference level = 20µPa
NIDSA_configure_eu_dbref (DSASession, "0", 20e-6);
```

4 Swept-Sine Mode Programming

You can use swept-sine mode to perform frequency-response measurements with a higher dynamic range than is usually possible with FFT mode. For information about how swept-sine measurements are performed, refer to the *Swept-Sine* section of Chapter 6, *Advanced Concepts*.

Note Only the NI 4551 can be programmed in swept-sine mode since this mode requires the instrument to generate stimuli.

You can find a swept-sine programming example, `455x Swept Sine Mode.vi`, on your NI-DSA CD. Figure shows the recommended program flow for swept-sine mode programming.

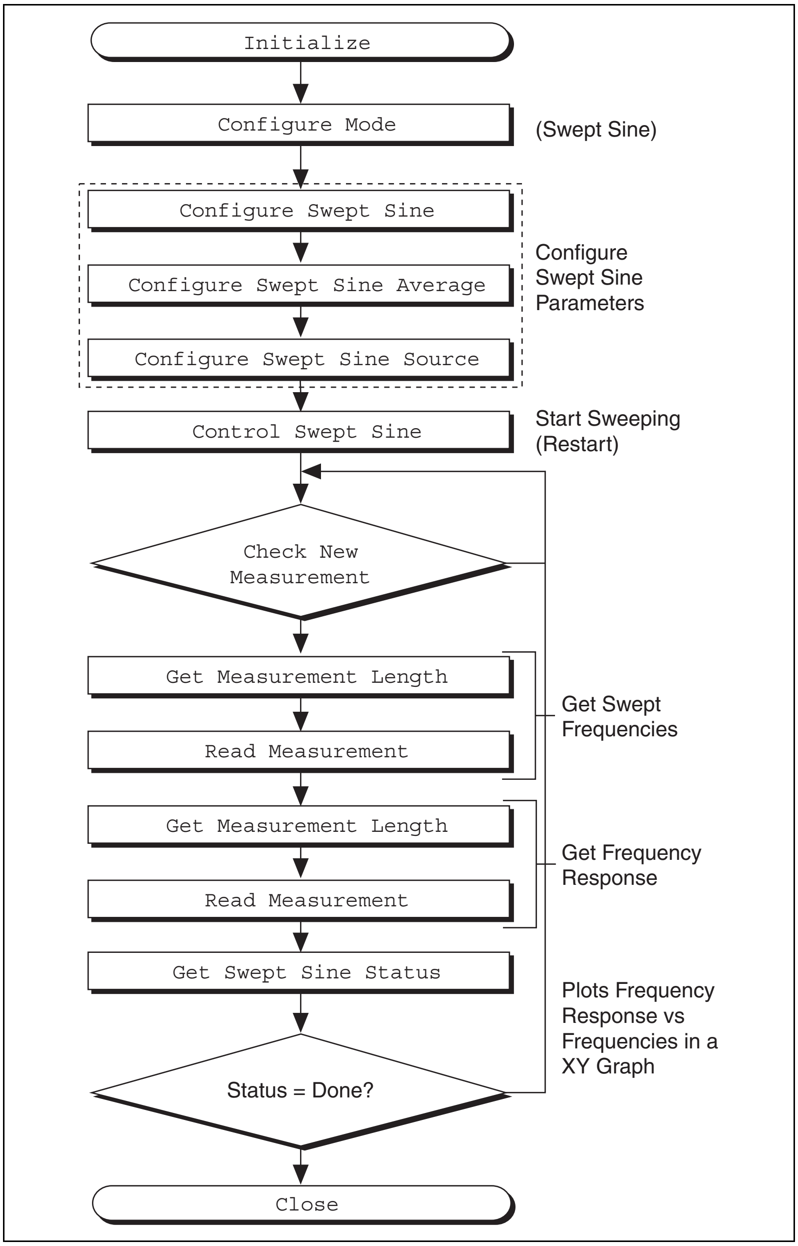

Figure 4-1. Swept-Sine Programming Flowchart

Step 2—Configure the Swept-Sine Analyzer

Configuring Inputs

Configure frequency characteristics of inputs to your *device under test* (DUT) with `Configure Swept Sine`. The primary `Configure Swept Sine` parameters are described in the following list:

- **Sweep mode**—choose from the following:
  - Normal—sweep from Start Frequency to Stop Frequency
  - Auto Resolution—automatically adjust sweep within **Auto Max Step**, **Fast Threshold dB**, and **Slow Threshold dB** parameters
  - Custom frequencies—use `Configure Swept Sine Custom` to set sweep parameters
- **Linlog**—set linear or logarithmic frequency steps
- **Start frequency**—starting sweep frequency
- **Stop frequency**—ending sweep frequency
- **Number of steps**—number of frequency steps in sweep

Set the sine sweep settling time and integration time with `Configure Swept Sine Average`.

Configuring the Source

`Configure Swept Sine Source`—sets the amplitude characteristics of your sweep. Important parameters include:

- **Amplitude**—sets the sine source peak level, in volts, unless **Auto Level** is enabled
- **Auto level enable**—when `TRUE`, the source amplitude is automatically maintained at the level specified by **Ideal Ref Level**. The output of the system under test is acquired on an input channel. When **Auto Level** is enabled, the amplitude of the source is adjusted to maintain a constant input channel level.
- **Ramping enable**—when ramping is enabled (**ramping enable** = `TRUE`), the source amplitude changes at the rate specified by **Ramping Rate**; if `FALSE`, the source amplitude is allowed to change instantaneously.

LabVIEW Example

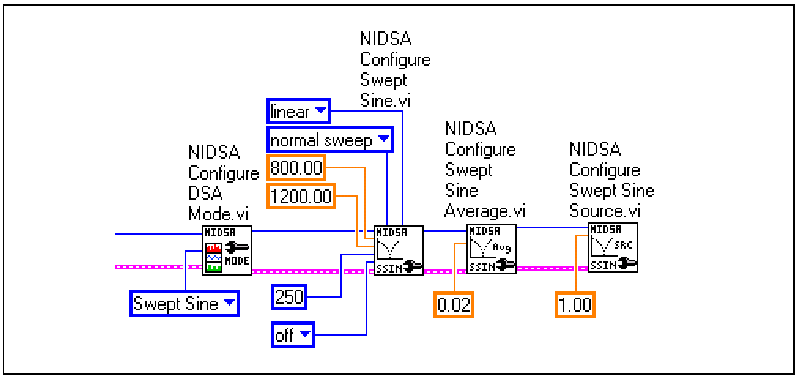
NIDSA Configure Swept Sine.vi
NIDSA Configure DSA Mode.vi
NIDSA Configure Swept Sine Average.vi
NIDSA Configure Swept Sine Source.vi
linear
normal sweep
800.00
1200.00
Swept Sine
250
off
0.02
1.00

Figure 4-2. Configuring the Swept-Sine Analyzer

C Example

```
//Configuring Swept sine analyzer

NIDSA_configure_dsa_mode (gDSASession, SWPSINE_MODE);

NIDSA_configure_swept_sine
(gDSASession,startFreq,stopFreq,numberOfSteps, 0, 0, 0,
512, 1.0, 6.0);

NIDSA_configure_swept_sine_average (gDSASession, 0.01,
1, 0.01, 1);

NIDSA_configure_swept_sine_source (gDSASession,
amplitude, 0, "1",1.0, -3.0, 3.0, 10.0, 0, 1.0);
```

Notes on Examples

Configuring Harmonics Measurements (Optional)

`Configure Swept Sine Harmonics`—configure the harmonics used for swept-sine harmonic measurements.

- **Maximum THD harmonic**—sets the highest harmonic number for THD computations. For example, set maximum THD harmonic to `3` to use only the second and third harmonics to compute THD.
- **Enable harmonic mapping**—sets to `TRUE` to measure harmonics of the source frequency on the response. **Enable Harmonic Mapping** has no effect on your THD measurement.

- **Harmonic mapping array**—if **Enable Harmonic Mapping** is `TRUE`, you must set the five array elements to 32-bit integers corresponding to the five harmonics to be returned as **Swept Sine Harmonic <1..5>**. For example, if you set the elements of the **Harmonic Mapping Array** to [2,3,5,9,20], the measurements returned are as follows:
  - Swept-sine harmonic 1—2nd harmonic
  - Swept-sine harmonic 2—3nd harmonic
  - Swept-sine harmonic 3—5th harmonic
  - Swept-sine harmonic 4—9th harmonic
  - Swept-sine harmonic 5—20th harmonic

If **Enable Harmonic Mapping** is `TRUE` and you do not specify a **Harmonic Mapping Array**, the default array, [2,3,4,5,6], is used.

The maximum value for **Maximum THD Harmonic** and any element in the **Harmonic Mapping Array** is 64.

Configuring Custom Frequencies (Optional)

If **sweep mode** (in `Configure Swept Sine`) is set to custom frequencies, use `Configure Swept Sine Custom` to specify the sweep frequencies. The parameters for this function are:

- **Buffer size**—sets the size of the custom frequencies array, up to 2,048
- **Custom frequencies array**—specifies the frequencies to step through

Step 3—Read Swept-Sine Measurement

Before you can read a measurement, you need to know if it has been completed. `Check New Measurement` returns a `1` if a new measurement is ready. You must call `Check New Measurement` before reading the first measurement and before reading any subsequent measurement set.

Reading a New Measurement

To read a new measurement, use the following functions:

- `Get Measurement Length`—returns the number of data points to read, based on the type and length of the measurement, and the set of frequency lines you choose with **frqLineSelector**.

Values for **frqLineSelector** are:

- Valid lines—returns alias-free lines only
- All lines—returns all FFT lines
- Classical lines—returns alias-free classical lines

- `Read Measurement`—reads the measurement from the specified analyzer channel

The valid measurement **Type** settings for `Get Measurement Length` and `Read Measurement` are limited to the measurements that you specified with `Configure Swept Sine`, `Configure Swept Sine Average`, and `Configure Swept Sine Harmonics`. The valid **type** values for reading swept-sine measurements are as listed:

- Swept spectrum 0—reads complex measurement of stimulus (source) level (fundamental only)
- Swept spectrum 1—reads complex measurement of response level (fundamental only)
- Swept frequency axis—reads the list of frequencies the analyzer has been sweeping
- Swept frequency response—reads frequency response (transfer function)
- Swept cross-power spectrum—reads cross-power spectrum of stimulus and response
- Swept THD—reads the total harmonic distortion measurement, calculated using the second harmonic through the harmonic specified by **Maximum THD Harmonic** in `Configure Swept Sine Harmonics`
- Swept SINAD—reads signal-to-noise + distortion

$$\text{SINAD} = \frac{1}{\text{THD + Noise}}$$

- Swept harmonic 1—reads the frequency response of the source frequency harmonic specified in the first element of the five-element harmonic mapping array created by `Configure Swept Sine Harmonics`
- Swept harmonic 2—same as swept harmonic 1, using the 2nd element in the harmonic mapping array
- Swept harmonic 3—same as swept harmonic 1, using the 3rd element in the harmonic mapping array

- Swept harmonic 4—same as swept harmonic 1, using the 4_{th} element in the harmonic mapping array
- Swept harmonic 5—same as swept harmonic 1, using the 5^{th} element in the harmonic mapping array
- Swept RMS squared—reads the total RMS^2 magnitude of the response
- Swept noise—reads the total noise excluding the fundamental

LabVIEW Example

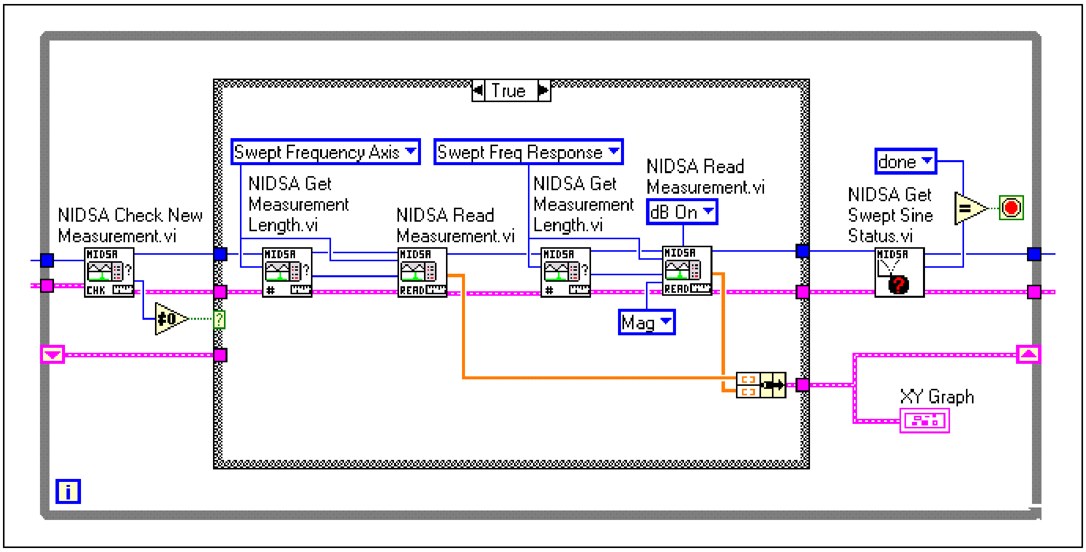
True
Swept Frequency Axis
Swept Freq Response
NIDSA Check New Measurement.vi
NIDSA Get Measurement Length.vi
NIDSA Read Measurement.vi
NIDSA Get Measurement Length.vi
NIDSA Read Measurement.vi
dB On
Mag
done
NIDSA Get Swept Sine Status.vi
XY Graph

Figure 4-3. Reading Swept-Sine Measurement

C Example

```
//Read Swept sine measurements

//Reading Swept Sine measurements is a 2 step process: //Read Frequency axis
informations, then read Frequency //response informations

//These 2 arrays will be displayed in a XY graph //(Frequency response vs
Freq Axis)

int CVICALLBACK TimerCB (int panel, int control, int event,void
*callbackData, int eventData1, int eventData2)

{

ViStatus sweepStatus;

ViStatus NewMeasurement;
```

```
ViInt32 overload;
ViInt32 measError;
ViInt32 sweepState;
                       //Variables used for Frequency axis
                       ViInt32 RealOrComplex;
                       ViInt32 Length;
                       ViReal32 dx;
                       ViReal32 x0;
                       //Variables used for Frequency response
                       ViInt32 respRealOrComplex;
                       ViInt32 respLength;
                       ViReal32 respdx;
                       ViReal32 respx0;
                       switch (event)
                       {
                       case EVENT_TIMER_TICK:
                       NIDSA_get_swept_sine_status (gDSASession,
                       &sweepState, &measError,&overload);
                       //Sweeping in progress
                       if (sweepState == 0)
                       {
                       //Check if a new measurement is ready
                       NIDSA_check_new_measurement (gDSASession,
                       &NewMeasurement);
                       if (!NewMeasurement)
                       return 0;
                       //Read Frequency axis values
                       NIDSA_get_measurement_length (gDSASession,
                       SweptFreqAxis, 1, &Length,&RealOrComplex);
                       NIDSA_read_measurement (gDSASession, SweptFreqAxis,
                       0, Length, 0, 0, 0, 0, &x0,&dx, gFreqAxis);
                       //Read Frequency response
                       NIDSA_get_measurement_length (gDSASession,
                       SweptFreqResp, 1, &respLength,&respRealOrComplex);
```

```
                                NIDSA_read_measurement (gDSASession, SweptFreqResp,
                                0, respLength, 2,1, 0, 0, &respx0, &respdx,
                                gFreqResp);
                                PlotXY (mainpanel, MAINPANEL_GRAPH, gFreqAxis,
                                gFreqResp,Length, VAL_FLOAT, VAL_FLOAT,
                                VAL_THIN_LINE,VAL_EMPTY_SQUARE, VAL_SOLID, 1,
                                VAL_RED);
                                }
                            break;
                            }
return VI_SUCCESS;
}
```

Step 4—Control

Checks the status of the ongoing measurement and error handling.

Status functions return the status of the instrument, such as whether or not an overload has occurred at the input, is the instrument operating in real-time, or the time that has elapsed since averaging was begun. (Example: `Get OLM Status`)

`Check Status`—returns status values for up to seven conditions. Some of the key status indicators are listed here:

- Real time—indicates whether the onboard DSP can keep up with the rate of data acquisition
- Main error—indicates an error on the DSA device
- Input overload status—indicates an overload condition on an input channel
- Wait on trigger—indicates that your DSA instrument is waiting for a trigger condition to be met. Refer to Chapter 3, *Hardware Overview*, of the *NI 4551/4552 User Manual* for more information about triggering.

`Get Swept Sine Status`—returns the status of the swept-sine measurement with the following indicators:

- Sweep state—returns the state the current sweep in swept-sine analyzer mode
  - 0: Sweeping—sweep is in progress
  - 1: Paused—sweep has been paused
  - 2: Done—sweep has completed
- Measurement error—An overload or measurement error has occurred during the present sweep.
- Overload—An overload has occurred during the present sweep

`Control swept sine`—controls the swept-sine analyzer mode

- Sweep control—controls the swept-sine analyzer mode
  - Run—runs/resumes the sweep from the present frequency
  - Pause—pauses the sweep at the present frequency
  - Restart—restarts the sweep at the configured start frequency

LabVIEW Example

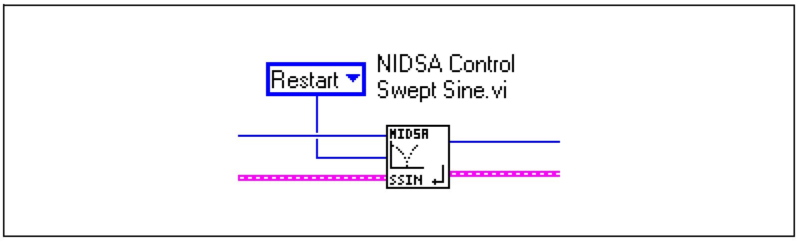

Figure 4-4. Controlling the Swept-Sine Measurement

C Example

```
//Controlling the Swept sine measurement
NIDSA_controlsweptsine (DSASession, SWEEP_RESTART);__
```

Source Mode (NI 4551 Only)

Source mode is useful for generating signals for frequency response measurements, and so on.

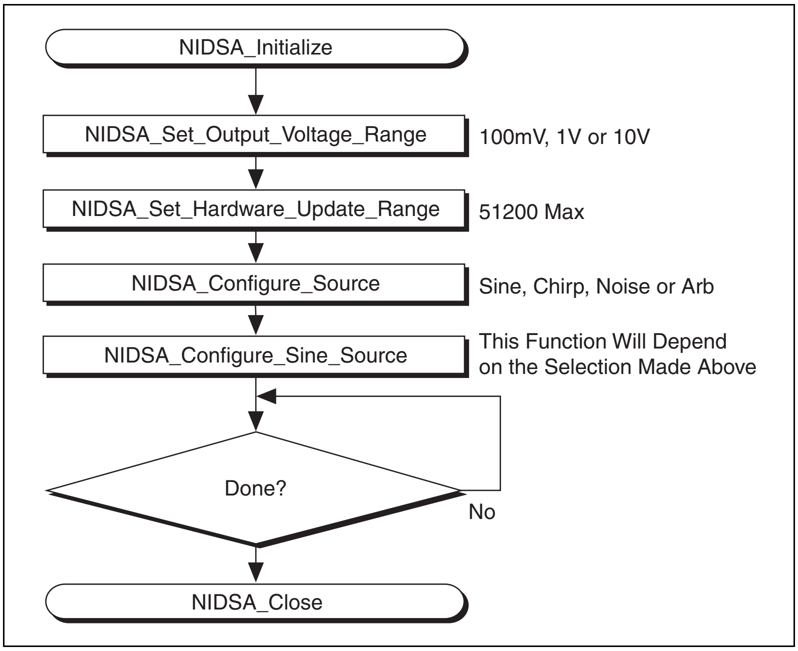
NIDSA_Initialize
NIDSA_Set_Output_Voltage_Range
100mV, 1V or 10V
NIDSA_Set_Hardware_Update_Range
51200 Max
NIDSA_Configure_Source
Sine, Chirp, Noise or Arb
NIDSA_Configure_Sine_Source
This Function Will Depend on the Selection Made Above
Done?
No
NIDSA_Close

Figure 4-5. Source Mode Flowchart

Step 2—Configure Source

- `Set Output Voltage Range`—sets the desired output voltage range for the selected output channel
  - **outputSelect**—analog output channel select to be configured
    - 0: channel 0
    - 1: channel 1
    - –1: configures all channels simultaneously

Note Presently, only one output channel (0) is supported.

  - **voltageRange**—voltage range for the selected output channel
    - 0: ± 10.0V
    - 1: ± 1.00V
    - 2: ± 100mV
    - 3: OFF (not supported)
- `Set Hardware Update Rate`—sets the hardware update rate. Default Value: 51200.0 S/s; Valid Range: 1,250.0 to 51,200.0 S/s.
- `Configure source`—configures the source output, turns the output signal on or off, and sets the desired output signal type. Valid source signal types are Sine (dual tone), Chirp, Noise, and Arbitrary.
  - **sourceOn**—turns the output source signal on or off
    - 0: off (default)
    - 1: on
    - 2: one-shot
  - **type**—sets the type of output signal to generate. The valid values for type, and the function to use to further define the outputs as follows:
    - 0: Default. Sine (dual tone). Use `Configure Sine Source` next.
    - 1: Chirp. Use `Configure Chirp Source` next.
    - 2: Noise. Use `Configure Noise Source` next.
    - 3: Arbitrary. Use `Configure Arbitrary Source` next.
- `Configure Sine source`:
  - **Freq 1**: Frequency of the first tone
  - **Amp 1**: Amplitude of the first tone
  - **Freq 2**: Frequency of the second tone
  - **Amp 2**: Amplitude of the second tone
  - **DC Offset**: DC offset, in volts, of the dual-tone signal output

If the source type is noise, you can also use the **Noise Type** and **Band-Limited Noise** parameters. Refer to the NI-DSA online help for more information.

In order to configure an Arbitrary source, the following operations are requested:

- `Configure Arbitrary Source`
  - **Amp**: Amplitude of the output signal
  - **Amp Mode**: Scale buffer by Amp (buffer needs to be normalized 0 < output value < 1)
- `Configure Arbitrary Buffer`—Enter the values of the arbitrary waveform (max buffer size = 4,096 points)

The update rate of your output is determined by the scan rate set for the inputs. If you change the input rate, the update rate is the same as the scan rate if the scan rate is less or equal than 51.2 kHz; one-half the scan rate if the input scan rate is between 51.2 kHz and 102.4 kHz; and one-fourth of the scan rate if it is between 102.4 kHz and 204.8 kHz.

Both of the outputs on the NI 4551 work as a single output if the device is used in DSA Instrument Mode through NI-DSA. Output channel 0 generates the specified output, and channel 1 generates the same signal with a 180 degree shift. If you need to operate the two output channels of your NI 4551 independently, program the instrument using NI-DAQ.

LabVIEW Examples

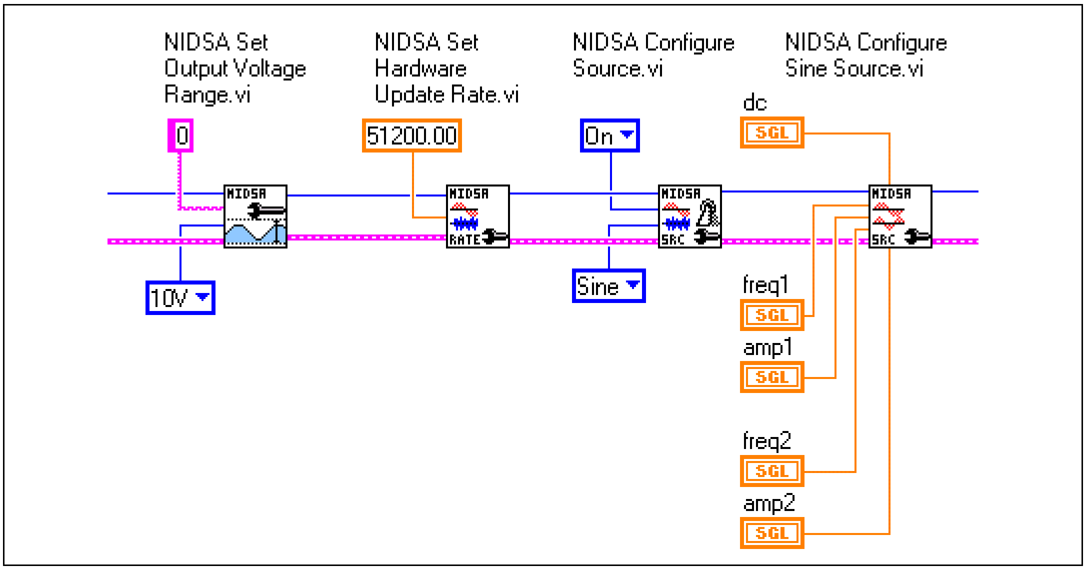
NIDSA Set Output Voltage Range.vi
NIDSA Set Hardware Update Rate.vi
NIDSA Configure Source.vi
NIDSA Configure Sine Source.vi
0
51200.00
On
dc
SGL
NIDSA
RATE
SRC
10V
Sine
freq1
amp1
freq2
amp2

Figure 4-6. Configuring the Source

C Examples

```
//Set Output voltage range to 10 V
NIDSA_set_output_voltage_range (DSASession, "0", OUTVRANGE_10V);

//Set update rate to 51.2 kS/s
NIDSA_set_hw_sampling_rate (DSASession, 51200);

//Turn source ON and select a sine wave output signal
NIDSA_configure_source (DSASession, SRC_ON, SINE_SRC);

//Configure sine source
//Note : This function allows a dual tone generation
//To generate a single tone, set Amplitude2 and Freq2 parameters to 0
NIDSA_configure_sine_source (DSASession, Freq1, Amplitude1, Freq2,
Amplitude2, DCOffset);
```

Step 3—Generate Signal

The preceding step actually starts the generation of the signal. In this step, you control the duration of signal generation. In this example, a while loop is used to continue signal generation for 50 ms.

LabVIEW Example

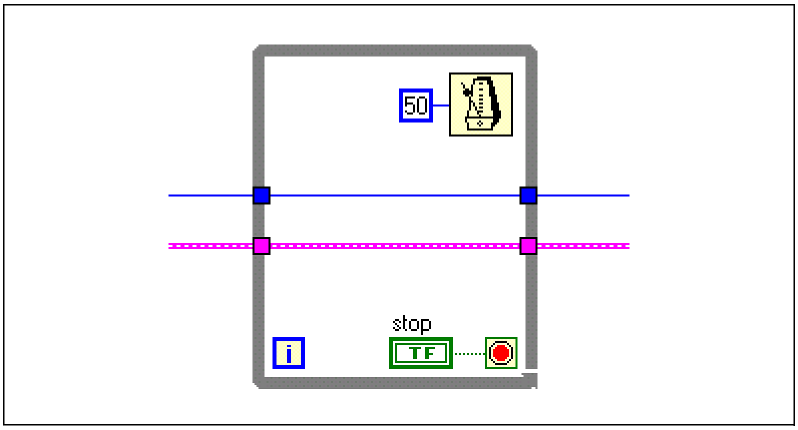

Figure 4-7. Generating the Signal

Octave Analysis (Add-On) Mode Programming

In many cases, when dealing with sound measurement and analysis, the final customer is the human ear, and like most human senses, the ear exhibits a response based on a logarithmic scale for both the level and the frequency. So, to produce results that are somewhere related to this human perception, sound levels are expressed in decibels and frequency content measured with a logarithmic scale. Many research efforts are currently focused on this field of psycho-acoustics, but octave band analysis remains the first choice technique.

Figure 5-1 illustrates the recommended programming flow for octave analysis with NI-DSA.

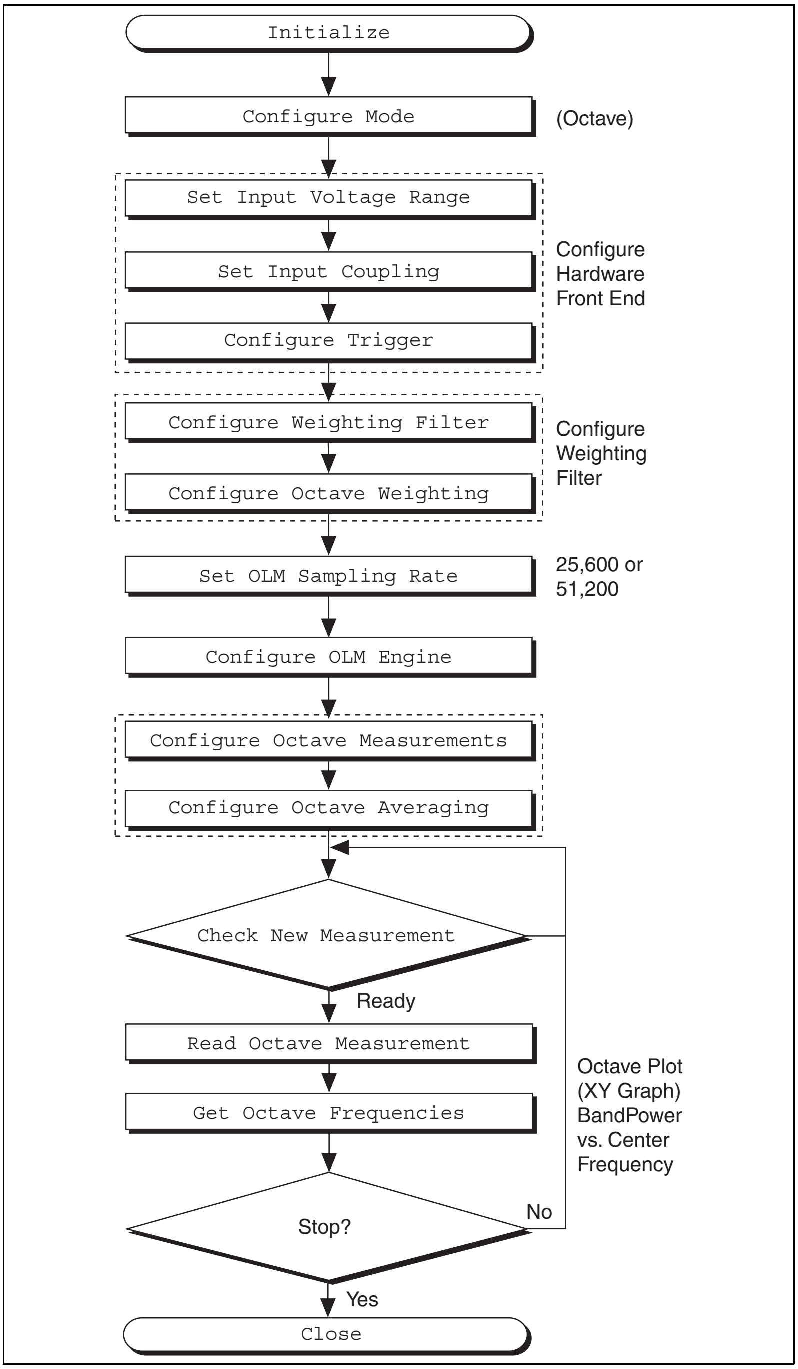

Figure 5-1. Octave Analysis Programming Flowchart for NI-DSA

Step 2—Configure the Octave Analyzer

Configuring Your Octave and Level Measurement

To configure octave and level measurements, use the following functions:

- `Set OLM Sampling rate`—set the hardware **Sample Rate** parameter to 25,600 or 51,200 S/s. The sample rate you select, the number of input channels used, and the number of additional real-time measurements you make affect the maximum center frequencies for octave analysis. Refer to the *Considerations for Octave and Level Measurements* section for more information on how your programming choices affect real-time operation.
- `Configure OLM engine`—enables measurements using the following parameters:
  - **Time**—acquires a time waveform. Time is automatically set to true if any of level, octave, or FFT are true.
  - **Level**—performs level measurement
  - **Octave**—performs octave measurement
  - **FFT**—performs baseband FFT measurement

Note You can set **octave** and **level** to `TRUE` in order to perform octave and level measurements at the same time.

- `Configure Octave weighting`—enables or disables weighting on a per-channel basis
- `Configure weighting filter`—enables A-, B-, or C-weighting
- `Configure Octave Measurements`—configures the type of octave analysis to perform, as well as the low and high center frequencies. Set the following parameters:
  - **Octave type**—1/1, 1/3 or 1/12
  - **Lo center frequency**
  - **Hi center frequency**
  - **Compliance ANSI/IEC**

- `Configure Octave averaging`—sets the averaging mode, time constant, and confidence level
  - **Average Type**
    - 0: None
    - 1: Linear—band power outputs are equally weighted and averaged over the specified integration time
    - 2: Exponential—band power outputs are exponentially averaged, with new filter data weighted more than older data
    - 3: Equal confidence—the time constant for each band power output is individually set so the results have a 68% probability of being within confidence level of the true mean for every band power level
    - 4: Peak hold—band power outputs are set at the peak output from each band filter
  - **Exponential mode**—selects the time constant to use for exponential averaging
    - 0: Fast—125 ms
    - 1: Slow—1,000 ms
    - 2: Impulse—35 ms time constant stage followed by a peak detector with a 1,500 ms time constant decay rate
    - 4: Custom—use **Time Constant** value
  - **Time Constant**—customized time constant value
  - **Confidence Level**

Note To set linear averaging integration time, use `Configure OLM Linear Averaging`.

LabVIEW Examples

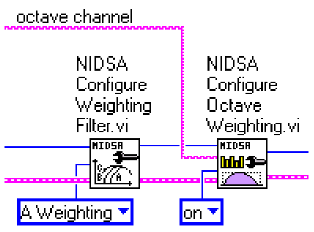
octave channel
NIDSA Configure Weighting Filter.vi
NIDSA Configure Octave Weighting.vi
A Weighting
on

Figure 5-2. Configure Weighting Filter

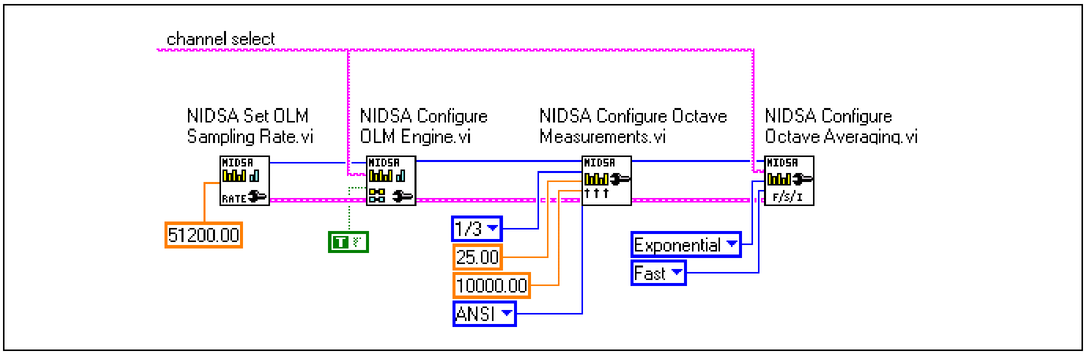
channel select
NIDSA Set OLM Sampling Rate.vi
NIDSA Configure OLM Engine.vi
NIDSA Configure Octave Measurements.vi
NIDSA Configure Octave Averaging.vi
51200.00
1/3
25.00
10000.00
ANSI
Exponential
Fast

Figure 5-3. Configure OLM Measurement

C Example

```
//Set the sampling frequency (25.6 or 51.2 kS/sec)
NIDSA_set_olm_sampling_rate (gDSASession, SampleRate);

//enables time, Octave and level measurements
NIDSA_configure_olm_engine (gDSASession, "0", VI_TRUE,
VI_TRUE,VI_TRUE, VI_FALSE);

NIDSA_configure_octave_measurements (gDSASession,
OctaveType,LoFreq,
HiFreq,OCT_COMPL_ANSI,&NumberOfBands);

NIDSA_configure_octave_averaging (gDSASession, "0",
AvgType, ExpMode,TimeConstant,1.0);
```

Step 3—Read Octave Measurement

Before reading a measurement, check to see if a new measurement is ready. The following operations are typically enclosed in a loop. In this example, a while loop is used.

- `Check new measurement`: returns `1` when a new block of data is ready, a `0` if not.
  - Octave mode: refer the octave description for further information
  - Octave power 0
  - Octave power 1
  - Octave power 2
  - Octave power 3
  - Level measurements 0

- Level measurements 1
- Level measurements 2
- Level measurements 3

When a new measurement is ready, it needs to be read:

- `Read octave measurements`

  Returns:

  - Band power measurements (needed to build the octave graph)
  - Total band-power measurements
  - Get octave frequencies
  - Exact center frequencies
  - Nominal center frequencies. Refer to the *Exact and Preferred Frequencies* section of Chapter 6, *Advanced Concepts*, for more information about these frequencies.
  - The fractional octave graph is an XY graph
  - The X-axis is nominal or exact center frequencies (selectable)
  - The Y-axis is the band-power measurements

LabVIEW Examples

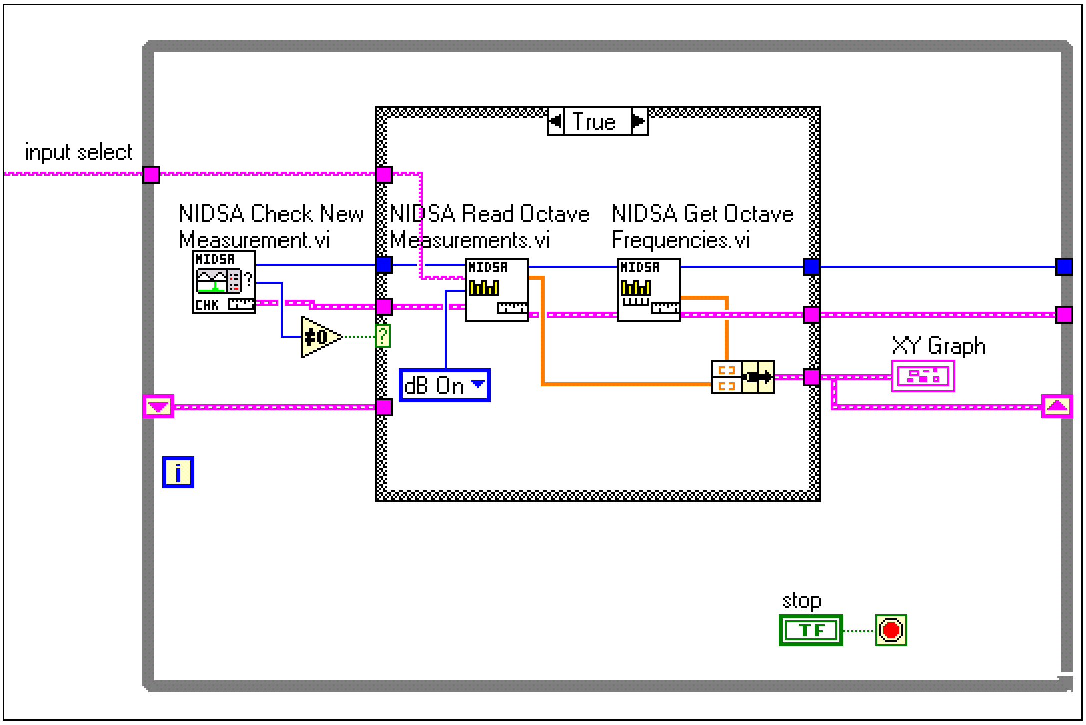
input select
NIDSA Check New Measurement.vi
NIDSA Read Octave Measurements.vi
NIDSA Get Octave Frequencies.vi
True
dB On
XY Graph
stop
TF

Figure 5-4. Reading the Octave Measurement

C Examples

```
int CVICALLBACK TimerCB (int panel, int control, int event,void
*callbackData, int eventData1, int eventData2)
{
                    ViStatus NewMeasurement;
                    //NewMeasurement==1 when a new measurement is ready


ViInt32 NumberOfBands;
ViInt32 FilterSettled;


ViReal32 TotalBandPwr;
```

```
ViReal64 linAvgTime;
ViReal64 ElapsedAvgTime;

ViString UnitLabel;

ViReal32 NominalCenterFreq[120];
ViReal32 ExactCenterFreq[120];
ViReal32 BandPowerMeas[120];

switch (event)
{
case EVENT_TIMER_TICK:

//Check if a new measurement is ready
NIDSA_check_new_measurement (gDSASession, &NewMeasurement);

if (!NewMeasurement)
return 0;

//Read the new measurement
NIDSA_get_octave_number_of_bands (gDSASession, &NumberOfBands);
NIDSA_get_octave_frequencies (gDSASession,
ExactCenterFreq,NominalCenterFreq);
NIDSA_read_octave_measurements (gDSASession, "0", DBON,&FilterSettled,
&ElapsedAvgTime,&linAvgTime, &TotalBandPwr,BandPowerMeas, UnitLabel);

//Displays Octave graph
DeleteGraphPlot (mainpanel, MAINPANEL_GRAPH, -1,VAL_IMMEDIATE_DRAW);
PlotXY (mainpanel, MAINPANEL_GRAPH, NominalCenterFreq,
BandPowerMeas,NumberOfBands, VAL_FLOAT, VAL_FLOAT,
VAL_VERTICAL_BAR,VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_RED);
}
```

```
return VI_SUCCESS;

}
```

Step 4—Control

Use `Get OLM Status`.

6 Advanced Concepts

This chapter contains information useful for understanding the different types of dynamic signal analysis and the algorithms used.

FFT Analyzer

The FFT analyzer mode performs frequency-domain measurements by applying the Fast Fourier Transform (FFT) to acquired time-domain data. The FFT analyzer mode can also be used to provide non-transformed time-domain data to the host computer.

In FFT analyzer mode, NI-DSA and your NI 45*XX* can perform up to two baseband FFTs and two zoom FFTs in real time, or four baseband FFTs. Features of FFT analyzer mode include:

- Baseband FFTs of 50 to 800 lines
- Zoom FFTs of 100 to 1,600 lines
- Classical range up to 80 kHz
- Extended range up to 95 kHz
- Overlap processing
- Phase suppression
- Averaging
- THD, THD+Noise, SINAD

The DSA FFT analyzer engine runs on the onboard processor and can analyze data from up to four input channels in real time, using two dual-channel analyzers: one contains two baseband FFT engines; the other contains two zoom FFT engines.

The dual-channel baseband analyzer allows you to select an independent frequency span from DC to 95 kHz, and up to 800 lines of frequency resolution, for each of two channels.

The dual-channel zoom analyzer lets you use any starting and ending frequencies for the zoom span. The two channels can have different zoom spans with up to 1,600 lines of frequency resolution, depending on the baseband span. The relationship between the zoom span and baseband span is explained in the *Baseband and Zoom Frequency Spans* section.

Any of the four input channels of the NI 4552, or the two input channels of the NI 4551 can be routed to any analyzer channel, or to multiple analyzer channels. For example, you can have one input channel routed to different analyzer channels with different settings in order to view one part of the frequency spectrum at a higher resolution than the other parts.

The functional block diagram for a dual-channel analyzer, either baseband or zoom, is shown below. Any input channel can be routed to either channel A, channel B, or to both channels.

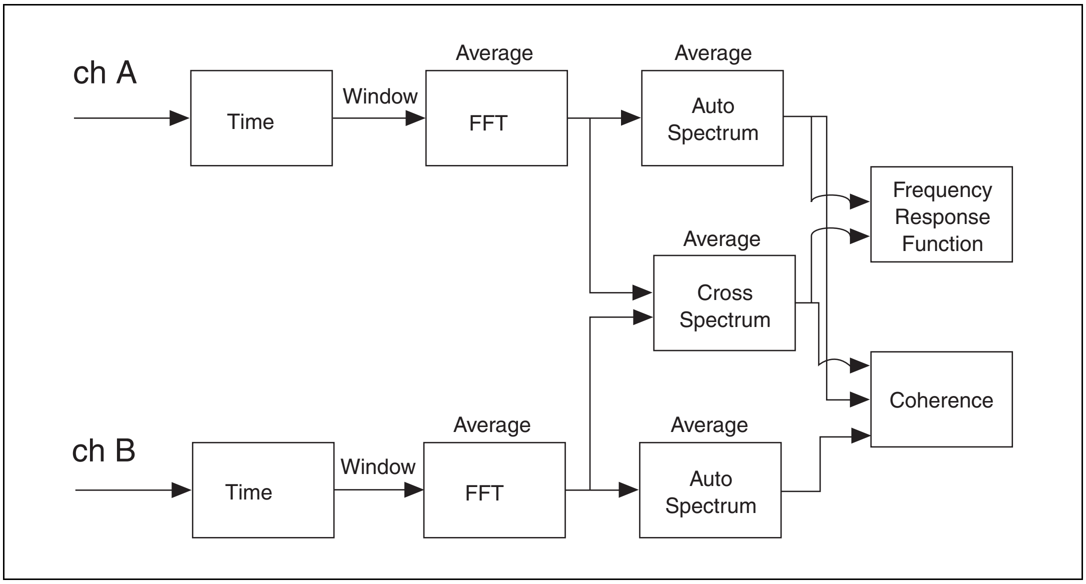

Figure 6-1. Dual-Channel Analyzer Functional Block Diagram

FFT Measurements

An FFT analyzer extracts the frequency spectrum of a time-domain signal. Fourier's theorem states that any waveform in the time domain can be represented by a weighted sum of sines and cosines. The same waveform can then be represented in the frequency domain as a pair of amplitude and phase values at each component frequency.

The FFT transforms digital samples from the time domain into the frequency domain. Each frequency component is the dot product (also known as the *inner product* or *scalar product*) of the time-domain signal with the complex exponential at the component frequency.

The discrete Fourier transform (DFT) describes the relationship between the time-domain signal *x(n)* and the frequency-domain signal *X(k)*:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi nk}{N}}$$

$$= \sum_{n=0}^{N-1} x(n)\left[\cos\left(\frac{2\pi nk}{N}\right) - j\sin\left(\frac{2\pi nk}{N}\right)\right]$$

The term FFT refers to any of several efficient algorithms for computing the DFT. The FFT is significantly faster than the DFT and the performance advantage grows as N increases.

When $k = 0$, the DC component is the dot product of $x(n)$ with $[\cos(0)–j\sin(0)]$, or with 1.0. When $k = 1$, the first bin, or frequency component, is the dot product of $x(n)$ with $\cos(2\pi n/N) – j\sin(2\pi n/N)$. Here, $\cos(2\pi n /N)$ is a single cycle of the cosine wave, and $\sin(2\pi n/N)$ is a single cycle of the sine wave. In general, the real part of FFT bin k, $\mathrm{Re}[X(k)]$, is the dot product of $x(n)$ with k cycles of the cosine wave, and the imaginary part of bin k, $\mathrm{Im}[X(k)]$, is the dot product of $x(n)$ with k cycles of the sine wave.

The FFT output, *X(k)*, is complex, containing real and imaginary components for each index *k*. For real-world applications, the magnitude and phase of the frequency-domain signal is of more interest than the complex output X(k). They can be computed using these formulae, which are essentially a cartesian-to-polar-coordinate conversion:

- Magnitude

$$|X(k)| = \sqrt{\mathrm{Re}[X(k)]^2 \times \mathrm{Im}[X(k)]^2} = X(k) \times X^*(k)$$

- Phase

$$\arg[X(k)] = \tan^{-1}\left[\frac{\mathrm{Re}[X(k)]}{\mathrm{Im}[X(k)]}\right]$$

The power spectrum reflects the energy content:

$$\text{Power Spectrum} = |X(k)|^2$$

The power spectrum, normalized to 1 Hz bandwidth, gives the power spectral density (PSD):

$$\text{PSD} = |X(k)|^2 / (\text{Df})$$

The use of the FFT for frequency analysis implies two important relationships:

- The highest frequency that can be analyzed is related to the sampling rate.
- The frequency resolution is linked to the total acquisition time, which is related to the sampling rate and the block size of the FFT.
- Acquisition time = $1/f_s \times$ N, where f_s = sampling rate and N = number of samples

To illustrate the above relationships, suppose that a digital time record, obtained from discrete samples taken at a selected sampling rate, is used to calculate a corresponding frequency spectrum composed of discrete frequency samples, or bins. For real signals, the frequency spectrum has half as many unique frequency bins as the time domain record has points. If the FFT size used is 1,024, that is, 1,024 samples are acquired, and the sampling rate is 204.8 kS/s, the complete time record acquisition will take 5 ms. This means that the lowest frequency in the spectrum corresponds to the period of the time record itself:

$$\frac{1}{5\text{ ms}} = 200\text{ Hz}$$

The FFT of a 1024-point record has 475 alias-free lines, or 476 including the DC bin. It does not have 512 lines because the width of the anti-alias filter transition band is non-zero. To provide alias protection at the Nyquist frequency ($f_s/2$), the filter also rejects some of the frequencies below the Nyquist frequency.

$$\frac{475\text{ lines}(204{,}800\text{ S/s})}{1024\text{ S}} = 95\text{ kHz}$$

The spectrum obtained represents the frequency range from DC to 95 kHz with 475 bins spaced every 200 Hz, as shown here:

$$\text{lines} = \frac{\text{span}}{F_S}N = \frac{95\text{ kHz}}{204.8\text{ kHz}}1024 = 475$$

$$\Delta f = \frac{\text{span}}{\text{lines}} = \frac{95\text{ kHz}}{475} = 200\text{ Hz}$$

Every frequency component of the FFT has a magnitude and phase. The phase is relative to the start of the time record, or relative to a cosine wave starting at the beginning of the time record. Single-channel phase measurements are stable only if the input signal is triggered. Dual-channel phase measurements compute phase differences between channels so that if the channels are simultaneously sampled, triggering is usually not necessary.

Dual Channel FFT Analysis

A typical application for dual-channel analysis is measuring the frequency response or transfer function of a system.

To characterize the cutoff frequency of an analog low pass filter you could send signals of every frequency within the filter's range and measure how much each signal is attenuated. A better way to do it is to send a signal that has all the frequency components in it, and see how much each of them is attenuated by comparing the response, or output, of the filter with its input. This process of feeding into a DSA both the signal sent to the filter and the response obtained from it, and comparing the amplitude levels of each frequency component is known as a frequency response analysis.

When you analyze two simultaneously sampled channels, the properties of each channel are not particularly important, but the relationship between them are. For example, the exact levels of a filter's input and output signals are not critical; it is the ratio of output to input that determines the filter's response at a given frequency.

A dual channel analyzer computes an instantaneous windowed FFT for each input channel. This instantaneous FFT can be averaged and processed to generate other measurements. The average auto-power spectrum of channel *x* (S_{xx}) is computed by multiplying each FFT component by its complex conjugate:

$$\overline{S_{xx}(f) = S_x{}^*(f)S_x(f)}$$

The average cross-power spectrum (S_{xy}) is a dual-channel measurement computed by multiplying each FFT component of channel *x* by the complex conjugate of the corresponding component from channel *y*:

$$\overline{S_{xy}(f) = S_x{}^*(f)S_y(f)}$$

The complex frequency response function *H(f)* is calculated from measured spectra using the following formula. Here, channel *x* represents a stimulus signal sent to the unit under test, while channel *y* represents the response coming from the test unit.

$$H(f) = \frac{\overline{S_{xy}(f)}}{\overline{S_{xx}(f)}}$$

The diagram below illustrates the frequency response *H(f)* of a system at one particular frequency *f*. A_x and A_y represent the amplitudes of the stimulus and response signals respectively. P_x and P_y are the phase of the stimulus and response signals.

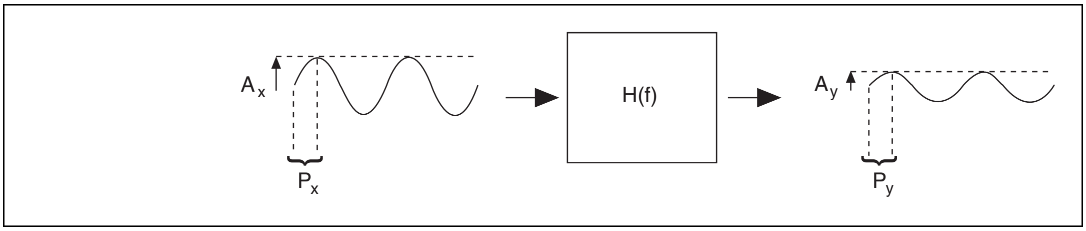

Figure 6-2. Frequency Response Measurement

Most frequency response measurements involve calculating the magnitude and phase of the complex quantity *H(f)*. The magnitude and phase can be related the physical diagram above by the following formulas.

$$\left|H(f)\right| = \sqrt{H^*(f)H(f)} = \frac{A_y(f)}{A_x(f)}$$

$$\angle H(f) = \tan^{-1}\left(\frac{\mathrm{Im}(H(f))}{\mathrm{Re}(H(f))}\right) = P_y(f) - P_x(f)$$

This frequency response function assumes that the unit under test is a perfectly linear, noise-free system. In practice, there is a finite nonlinearity present in all systems. Any noise in the output is generally uncorrelated to the input signal. Thus, the quality of the frequency response measurement can be improved by averaging several spectra to help cancel random noise.

The coherence function γ^2 provides a measure of the correlation between the system response and stimulus. γ^2 is an array quantity with a value at each FFT frequency value f. In a perfectly linear system, γ^2 has a value of 1. Another way of expressing this is that all the response signal was completely caused by the stimulus. A value of 0 indicates no relation between the response and stimulus. The coherence function provides an estimate of the quality of a frequency response measurement. In general, higher γ^2 values indicate better measurements. Coherence is calculated as follows:

$$\gamma^2(f) = \frac{\left|\overline{S_{xy}(f)}\right|^2}{\overline{S_{xx}(f)S_{yy}(f)}}$$

Baseband and Zoom Frequency Spans

As explained in the *FFT Analyzer* section, the NI 45*XX* offers zoom and baseband analyzers. Some of the specifications and uses of these analyzers, and the relationship between them, are explained here.

Alias-Free Bandwidth

The alias-free bandwidth is a specification of the usable frequency range of the instrument at a given sample rate. Signals at or below the alias-free bandwidth is measured at the specified accuracy.

To prevent aliasing, the analog signal applied to a digitizer can not contain frequency components that are above one-half of the sample rate f_s, or $f_s/2$. Such a filter can not be implemented; a realizable filter begins attenuating below $f_s/2$ and does not sufficiently attenuate until some point above $f_s/2$, which can result in aliasing of signals slightly below $f_s/2$. For this reason, the alias-free bandwidth of the instrument will be somewhat less than $f_s/2$.

The oversampling delta-sigma converters used on the NI 4551 and NI 4552 contain integral antialiasing filters that provide an alias-free bandwidth of 46.4% of the sampling frequency. At the maximum sampling rate of 204.8 kS/s, the NI 45*XX* alias-free bandwidth is 95 kS/s (204.8 kS/s $\times$ 46.4% = 95 kS/s)

Classical and Extended FFT Size

Although the superior antialiasing performance of the NI 45*XX* allows a larger alias-free bandwidth and thus more spectral lines, sometimes a user wants to make measurements at bandwidths that have traditionally been used in spectral analysis. Table 6-1 gives the number of frequency lines for standard FFT sizes using the extended and classical FFT spans.

Table 6-1. Number of Frequency Lines for Classical and Extended Spans

FFT Block Size	Extended FFT Span	Classical FFT Span
128 samples	59 lines	50 lines
256 samples	118 lines	100 lines
512 samples	237 lines	200 lines
1,024 samples (1 kS)	475 lines	400 lines
2,048 samples (2 kS)	950 lines	800 lines

Baseband Spans

A baseband is a frequency span for a measurement that starts at DC and extends to the maximum alias-free span of the device, which depends on the sampling rate.

For an FFT block size of 1,024 and a sampling rate of 204.8 kS/s, the final spectrum contains 476 frequency bins, with the first one representing –200Hz/2 to 200Hz/2, the second one 200Hz/2 to 3×200Hz/2, and so on until the 476th, which will be centered around 95 kHz.

Due to the delta-sigma converter's inherent oversampling, filtering, and antialiasing protection, the minimum sampling rate allowed is 5 kS/s, which corresponds to a minimum baseband span of 2,320 Hz (5,000 S/s × (1,024/475) = 2,320Hz) with the extended FFT size and a minimum of 1,953 Hz (5,000 × (1,024/400) = 1,953Hz) with the classical size, and because of the above mentioned reasons the maximum baseband span is 95 kHz.

In NI-DSA, the maximum FFT size is 2 kS (2,048 samples), allowing up to 800 FFT lines classical or 950 lines extended frequency span.

To summarize, the FFT resolution (number of lines) that you specify determines the length of the time-domain record, and number of samples to be acquired. The reverse is also true.

The sampling rate determines the frequency span.

The FFT resolution is the number of lines or bins, and the frequency resolution is the frequency span divided by the number of lines.

Zoom Spans

If you need better frequency resolution than the baseband analyzers can provide, you need to use the narrow span capabilities of the zoom analyzers.

The Real Time Zoom FFT Process

Time record length is directly related to frequency resolution. Increasing the duration of the time record increases the frequency resolution. To increase the duration of the time record, you must either capture more points or lower the sampling rate.

A narrower span allows you to "zoom in" on a center frequency with a selected span, thus improving your frequency resolution. A zoom span has a start frequency, a center frequency, and an end frequency. The zoom span is determined by the difference between the start and end frequencies.

The real-time zoom approach is to digitally filter and down-sample the acquired time record. This technique, along with heterodyning the signal to center the frequency spectrum on a frequency other than DC, provides a zoomed spectrum.

Ideally, you could center the narrow span around any frequency below the baseband span. Using decimating filters alone, all spans must begin at 0 Hz (DC). If you frequency-shift, or *heterodyne*, the input signal before the decimation stages, the resulting span is centered at the modulation frequency, allowing you to center your span in any frequency as long as the narrow span stays within the baseband span.

Heterodyning is used by the analyzer to compute zoomed spans, those starting at frequencies other than DC. The signal processor can heterodyne and filter in real time to provide a time record at all spans and center frequencies. All of the signal processing computations, including modulation, digital filtering and decimation, and computing the FFT are done is less time than it takes to acquire the data, so the NI 45*XX* can process every input sample in measuring the signal spectrum.

In the heterodyning process the input points are multiplied by a complex unit vector (cos(wt) and sin(wt)) to yield a real and imaginary time record.

The original frequency span center is then shifted to DC or heterodyned, this means that the upper half of the span stays positive and the lower half (below the span center) becomes a set of negative frequencies. The result is that a span from 0 to 80kHz becomes a span of –40 to 40 kHz. This data goes through a digital low pass filter which cuts off at ±40 kHz, this operation results in a ±40 kHz usable span centered at 40 kHz. The sampling rate needed is then only 102.4 kS/s instead of the original 204.8 kS/s hardware sampling rate defined by the selected baseband of 80kHz, so only every other point of the original span is used. At this point the original baseband span is represented by a complex time record with half as many points as the original one, half the sampling rate and therefore same duration (both the real and imaginary part of this complex record have half of the original sample rate and half of the original span).

If you keep the hardware sampling rate constant at 204.8 kHz, the NI 45*XX* can digitally halve the effective sampling rate by passing the digital samples through one stage of a decimate-by-2 lowpass filter. Digital filtering and down-sampling are used to narrow the heterodyned data by zooming around the heterodyne frequency (the center of the original span). So while the first filter reduces the sampling rate by half and the number of samples by splitting the time record in real and imaginary parts, the second filter cuts off at ±20 kHz reducing the sample rate by 2 again, but not the number of points. The new record then has twice the original duration and half of the original span, resulting in a a 40kHz span centered around 40kHz. The time record duration is twice the duration of the full span time-record. The sample rate is one-fourth of the baseband rate.

Further iterations of this process can reduce the span and provide better resolution for analysis.

If you pass 2,048 samples (for a 2 KS FFT) at 204.8 kHz through this decimating filter, the output will be 1,024 samples at 102.4 kHz. Both records correspond to a 10 ms duration time record, and so the frequency resolution, or bin width, is 1/10 ms = 100 Hz. Once again, the length of the time record always determines the frequency resolution of the spectrum.

To make sure the decimating filter is fast and memory efficient, the alias-free range for narrow spans is reduced from 475 lines to the classical 400 lines per 1,024-point FFT to allow the use of a faster and less sharp filter, thereby improving performance. As a result of this reduction in range, the maximum alias free frequency span is 80 kHz for zoom instead of the 95 kHz available for baseband. The resulting span is (400/1,024) $\times f_s$, so that our span for one stage of decimation is (400/1,024) $\times f_s/2$ = 40 kHz.

Performing a 1,024 point FFT at the highest span yields a frequency resolution of 200 Hz. 1,024 points of decimated data corresponds to a time duration of twice that of the highest span, so the frequency resolution is improved, to 100 Hz. This process of doubling the time record and halving the span and resolution can be repeated for each additional decimate-by-2 filter stage.

Zoom Span Characteristics

Zoom spans must be a power of 2 of a classical baseband span. If you want a 12,800 Hz zoom span, you can use a 12,800 Hz baseband span in classical mode (100, 200, 400, 800 lines) or 12,800 Hz divided by any power of 2, as long as the resulting span lies within the baseband span. Use the following formula to calculate the classical baseband span required for the zoom span you need:

$$\text{required\_classical\_baseband\_span} = \text{zoom\_span} \times 2^n.$$

A 1,600 Hz zoom span is allowed with a 12,800 Hz classical span, because the following is true:

$$12{,}800 = 1{,}600 \times 2^3$$

A 1,400 Hz zoom span, on the other hand, is not compatible with a 12,800 Hz baseband span, since no power of 2 of 1,400 is equal to 12,800:

If you are using an extended baseband span, an additional calculation is necessary to determine the correct baseband span for a particular zoom span. The formula for this case is:

$$\text{required\_extended\_baseband\_span} = \text{zoom\_span} \times (\text{extended\_lines} / \text{classical\_lines}) \times 2^n$$

If you need a zoom span of 12,800, you need an extended baseband span of 15,200 Hz, 32,400 Hz, or 64,800 Hz, for example. The 475/400 factor converts the classical zoom span to its extended representation in order to select the correct extended baseband span.

It is important to notice that when you select any zoom span, NI-DSA automatically coerces it to the closest possible zoom span available with the current baseband span.

The zoom span limits have to be within the baseband span, and the maximum extended baseband span is 95 kHz. You can select a zoom FFT of 100, 200, 400, 800, or 1,600 lines.

When the selected zoom span is larger than the classical baseband span selected (including the maximum classical baseband span of 80 kHz), the driver will make the zoom analyzer behave as a baseband analyzer and eliminate the modulation and decimation stages. By using the zoom analyzers as baseband analyzers, the two-dual-channel-analyzers architecture lets you perform an FFT analysis in the same frequency span on four channels at the same time.

Windowing

In practical applications, you can obtain only a finite number of samples of a signal. The FFT assumes that this time record repeats. If you have an integral (whole number) number of cycles in your time record, the repetition is "smooth" at the boundaries. However, in practical applications, you usually acquire a nonintegral number of cycles. In such cases, the presumed periodicity of the sampled signal results in discontinuities at the boundaries, as shown in Figure 6-3.

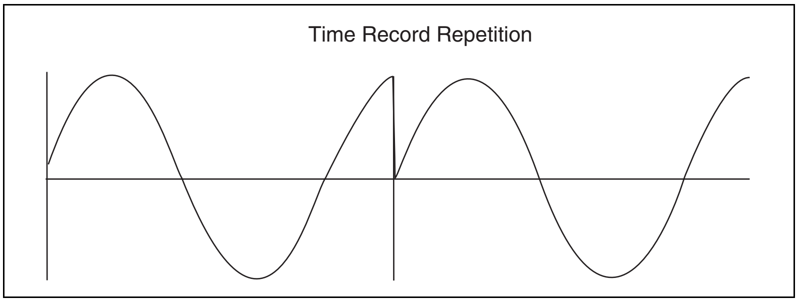

Figure 6-3. Waveform Discontinuity at Record Boundary

These artificial discontinuities were not originally present in your signal and result in a smearing or leakage of energy from your actual frequency to all other frequencies. This phenomenon is known as spectral leakage. The amount of leakage depends on the amplitude of the discontinuity, with a larger discontinuity causing more leakage.

Because the amount of leakage is dependent on the amplitude of the discontinuity at the boundaries, you can use windowing to reduce the size of the discontinuity and, hence, reduce spectral leakage.

Windowing consists of multiplying the time domain signal by another time domain waveform, known as a window, whose amplitude tapers gradually and smoothly toward zero at the edges. The result is a windowed signal with no (or very small) discontinuities, and, thus, reduced spectral leakage. There are many different types of windows. The one you choose depends on your application.

Figure 6-4 illustrates a window applied to a single waveform record. The record contains about 1.25 cycles of the sine wave.

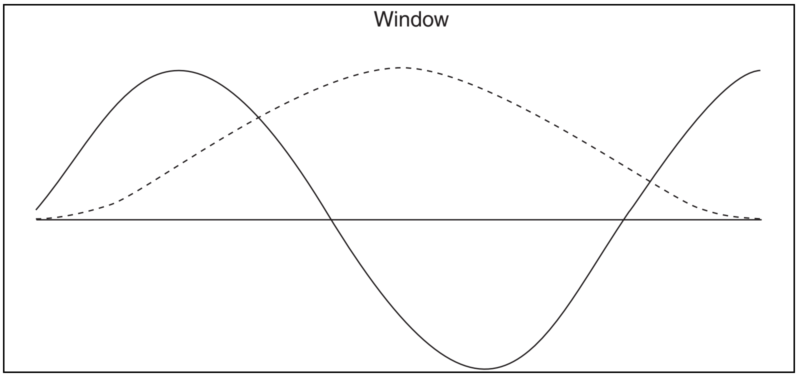

Figure 6-4. Window Applied to Waveform Record

Figure 6-5 illustrates two cycles of the windowed waveform, with the discontinuity between records is no longer present.

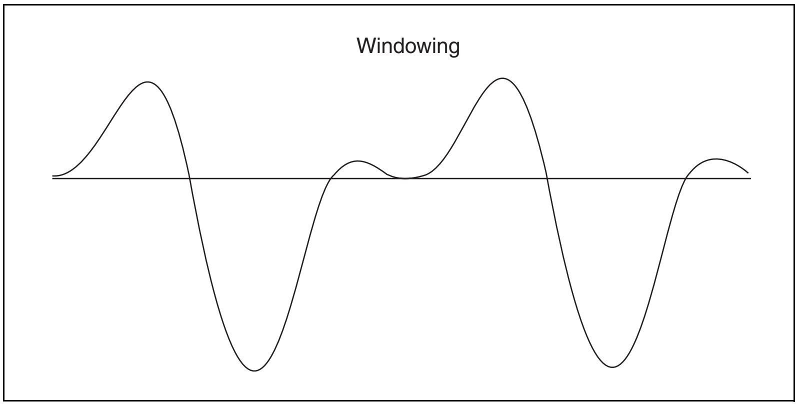

Figure 6-5. Waveform Discontinuity After Windowing

Some of the windows available in NI-DSA are:

- Uniform
- Hanning
- Blackman-Harris (standard, 4-term, and 7-term)

- Flattop
- Kaiser
- Low sidelobe
- Force
- Exponential
- Forced-exponential

For more information about using windowing functions, refer to NI Application Note 041, *The Fundamentals of FFT-Based Signal Analysis and Measurement*.

FFT Averaging

Averaging successive measurements tends to improve measurement accuracy. Averaging is usually performed on measurement results or on individual spectra, but not directly on the time record. Linear and exponential averaging are available in RMS, vector, and peak hold modes.

Linear averaging combines N spectral records with equal weighting. When the number of averages has been completed, the analyzer stops averaging and presents the averaged results.

Exponential averaging provides a continuous weighted average and emphasizes new spectral data more than old.

Both methods of averaging are performed using this formula:

$$\text{New Average} = (\text{New Spectra} \times 1/N) + (\text{Old Average} \times (N-1)/N)$$

where N is the number of averages for linear averaging, or is computed from the time constant for exponential averaging.

RMS averaging computes the weighted mean of the sum of squared values. RMS averaging reduces fluctuations in the data but does not reduce the actual noise floor.

Vector averaging uses the complex FFT spectrum. The real part is averaged separately from the imaginary part. This can reduce the noise floor for random signals, because they are not phase coherent from one time record to the next. For single-channel measurements, vector averaging requires a trigger so that the real and imaginary parts of the signal add in phase instead of canceling randomly.

Peak Hold averaging retains the peak of the spectral magnitudes on a frequency bin-by-bin basis. The peak values are stored in the original complex form so that all complex forms of the data can be displayed.

Each new spectral record for RMS and Vector averaging can be weighted using either Linear or Exponential weighting.

Swept-Sine

Unlike the FFT, which measures all the frequencies simultaneously, the swept-sine analyzer measures one single frequency at a time. Based on a specified sequence of frequency points, the swept-sine analyzer makes the measurements one by one. At each point, the source generates a sine wave and maintains a constant frequency. The inputs measure only at this frequency. After going through the sequence of frequency points, the measurements are complete. Figure 6-6 shows the typical application of swept-sine analysis.

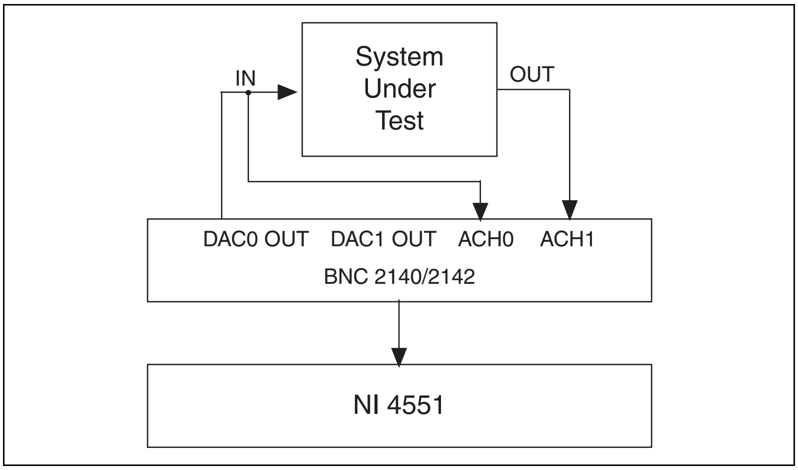

Figure 6-6. Typical Swept-Sine Measurement Application.

Why Swept-Sine Measurement?

The transfer function of a system can be measured by both FFT and swept-sine techniques. Since the swept-sine measurement generates and measures a single frequency at a time, it has several advantages over traditional FFT measurements. Instead of using the same settings on all frequency points as in the FFT case, a swept-sine analyzer can optimize the measurement at each individual frequency with auto-ranging and auto-level features.

Auto-ranging extends the dynamic range and auto-level increases the signal to noise ratio. If the system's transfer function has large variations within the measurement span, a swept-sine analysis can often give you greater dynamic range than an FFT analysis. Because the FFT measures all the frequency components at the same time, the source must contain energy at all of the measured frequencies. Generally each individual component's amplitude is about 30 dB less than the overall source amplitude in the time domain. So each frequency is actually measured at –30 dB relative to full scale. This effectively reduces the dynamic range of the measurement. A swept-sine measurement, on the other hand, can eliminate the 30 dB loss because it measures one frequency at a time, and this frequency can be applied full-scale.

With auto-ranging, swept-sine measurements can be further optimized for each frequency according to the output level of the device under test, so the signal level is as close to full scale as possible. Optimizing the input range at each frequency rather than applying the frequency-rich time-domain data can extend the dynamic range of the measurement to beyond 140 dB. For example, if the transfer function has both gain and attenuation, the auto-level control will adjust the source level to compensate for the output of the system. If the system under test has gain at one certain frequency, the amplitude of the sweep source can be reduced automatically to prevent the output of the system from overloading the DSA input channel. At the points with significant attenuation, the source level can be increased to boost the system output level to provide a better signal to noise ratio. Point-by-point optimization gives swept-sine analysis a great advantage over FFT measurement.

Using auto-resolution mode can greatly speed up your swept-sine measurements. Often, you want to measure a frequency response with a large span and you want to preserve the frequency resolution. With the FFT method, the only option is to increase the size of the FFT. This consumes memory and increases measurement time. In auto-resolution mode, even though a large number of points is still needed to increase the frequency resolution, the analyzer can skip points in the range where the frequency response is flat, and proceed step-by-step where a sharp transition occurs. This results in fewer measurement points without a loss of frequency resolution.

Swept-Sine Controls and Settings

The swept-sine measurements are controlled by the following settings:

- Averaging
- Sweep frequency and auto-resolution
- Input auto-ranging
- Source auto-level and ramping

Averaging

Averaging is controlled by the settling time and integration time.

Settling time is specified by time and cycles. The longer period specified by these two determines how long the instrument will wait before the integration starts after the source changes the frequency. This waiting period allows the device under test to respond to the frequency change so the measurement can be made based on a settled signal.

Like settling time, integration time is specified by time and cycles. Time is converted to the next largest whole number of cycles. The larger of these two will be used as the integration time. The integration time must be an integral number of cycles. This guarantees that the measurement results do not include DC or harmonics of the source frequency.

Swept-sine integration is a process of multiplying the time-domain data with a complex signal $cos(2\pi ft)+j*sin(2\pi ft)$ and averaging the results over the integration time. The frequency f of the complex signal is the same as the source frequency. To put it another way, a swept-sine measurement is just a direct form of Fourier Transform at each frequency point. This is why the Integration Time must correspond to an exact number of cycles. The longer the integration time, the narrower the detection bandwidth at the source frequency. However, the measurement takes longer to perform.

Sweep Frequency and Auto-Resolution

Start and stop frequencies specify the measurement frequency span and can be specified within the range shown here:

$$0 < f < f_s/2$$

Note Sweeping through very low frequencies—1 Hz, for example—will result in excessive integration times.

The number of points results in the minimum frequency resolution of a swept-sine measurement. number of points can be set from 1 to 2,048. The points can be in a linear or logarithmic progression. In some cases, it is desirable to sweep over a wide frequency range while still detecting narrow transition in the frequency response function. An example might be a narrow notch filter. To measure such a system, a large Number of Points are needed to obtain fine frequency resolution. This effectively increases the measurement time. In order to resolve this problem, auto-resolution mode can be used. In auto-resolution mode, fewer measurements will be made in frequency ranges where the response is fairly flat. Auto-resolution mode makes more measurements where response is a strong function of frequency. Auto-resolution can dramatically save measurement time without losing resolution. Since points may be skipped, the number of points actually measured may far less than that specified.

Auto-resolution is specified by these three parameters: **Faster threshold**, **Slower Threshold**, and **Maximum Step Size**.

Auto-resolution mode examines the measurements of successive frequency points. If the change of the most recent measurement relative to the previous one is within the **Fast Threshold** (for both channels), the sweep will take larger steps by skipping frequency points. Each time this threshold is met, the step size will be increased by one until the Maximum Step Size is reached. If the change of the measurements is more than the Slower Threshold (for either channel), then the analyzer discards the newest measurement and measures the point immediately after the last point that was within the thresholds. If measurements differ by more than the Faster Threshold (on either channel), but less than the Slower Threshold (on both channels), the analyzer maintains the present sweep speed by continuing to skip the same number of points.

Input Auto-Ranging

With auto-ranging on, the instrument can automatically adjust the input range based on the measured signal levels in an attempt to measure all inputs at full scale. This has a great impact on the dynamic range of the measurement. Input auto-ranging monitors the measurement of each point. If overloading occurs during the sweep, the analyzer increases the input range to the next higher level. If the signal level drops below the threshold of the next lower range, the analyzer decreases the input range accordingly.

Note Auto-ranging may increase measurement times.

Source Auto-Level and Ramping

There are five parameters associated with source auto-level. They are auto-level channel, ideal reference, upper reference, lower reference, and maximum source level.

Source auto-level adjusts the source amplitude to maintain a constant level, called Ideal Reference, at the channel 0 or channel 1 input. The reason to use Source auto-level is to improve the signal-to-noise ratio of the measured signal. Suppose you want to measure the transform function of a device with an input range set to 1.0 V, and the device has both gain and attenuation within the frequency span, for example from +30dBV to –100dBV. Without source auto-level, the source must maintain a constant level that is less than –30dBV (to prevent overload) across the whole frequency span. When the sweep frequency reaches the point of greatest system attenuation, the output signal level drops to –130dBV. Such a signal level is still measurable, but may not be optimum. With source auto-level, the source tries to maintain an Ideal Reference Level of 1.0V at the output of the device under test. The source level decreases at the point of gain, and the source level increases at the point of attenuation until the Maximum Source Level reached. In this example, the output level of the device under test is 0 dBV to –100 dBV (assuming the Maximum Source Level is reached) instead of 0 dBV to –130 dBV. Varying the source level narrows the range of the output signals, avoiding overloads when there is gain and increasing the output signal-to-noise ratio when there is attenuation.

Source auto-level requires the input auto-ranging to be on, because the inputs must track the changing source level.

Source auto-level should only be used when performing transfer function measurements, because the source level changes are not normalized. Only the ratio of channel 1 to channel 0 is source level independent.

Source Ramping is the rate at which the source level changes. With Source Ramping on, the source level changes gradually at the ramping rate. The settling time starts after the ramp. Enabling Source Ramping increases the sweep time, but avoids instantaneous source-level changes, which could cause the device under test to respond erratically.

Swept-Sine Measurements

Spectrum

The spectrum, the measurement of a single channel over a sweep, can be returned for channel 0, channel 1, or both. The spectrum is complex (meaning it contains real and imaginary components), with an amplitude of the measured signal, and phase relative to source. The phase measurement of a single channel (measuring the system output, but not the input, for example) is not generally meaningful.

Cross-Spectrum

The cross-spectrum of channel 0 and channel 1 is given by this formula:

$$\text{Cross-spectrum} = S_x{}^* \times S_y$$

The cross-spectrum is complex.

Frequency Response

Frequency response is a comparison of the gain and phase of the signal input to a device under test to the gain and phase of the device output. Specifically, channel 0 (source input) over channel 1 (response input). It is calculated as follows:

$$H(f) = \frac{Y(f)(\text{output})}{X(f)(\text{input})}$$

The result is complex.

THD - Total Harmonic Distortion

The calculation of THD is based on the measurement of the 2nd to the 64th harmonics, as shown in this formula:

$$THD_{\%} = 100\% \times \frac{\sqrt{\sum_{n=2}^{64} A_n^2}}{A_1}$$

A_n is the RMS value of the *n*th order harmonic. A_1 is the RMS value of the fundamental frequency. The number of harmonics involved in the THD calculation is bounded by the Nyquist frequency. If the frequency of the *m*th order harmonic exceeds the Nyquist frequency (half of the sampling frequency), *Am* is set to zero.

NI-DSA computes the THD on channel 1, which is connected to the output of the system under test.

With `Read Measurement` parameter **dB units** set to dB on, the THD is returned as a dB scaled value, which is calculated as shown in the following equation:

$$THD_{dB} = 20 \times \log \frac{\sqrt{\sum_{n=2}^{64} A_n^2}}{A_1}$$

SINAD

SINAD stands for Signal in Noise and Distortion. It is the reciprocal of THD+Noise.

In general, SINAD is computed as shown here:

$$SINAD = \sqrt{\frac{A_1^2 + N^2 + D^2}{N^2 + D^2}}$$

Where A_1 = signal amplitude (RMS), N = noise (RMS), and D = distortion (RMS).

NI-DSA calculates SINAD as shown here:

$$SINAD = 100\% \times \sqrt{\frac{E}{E - A_1^2}}$$

Where E = signal amplitude (RMS)^2, A_1 = RMS value of the fundamental frequency.

NI-DSA computes SINAD on channel 1, which is normally connected to the output of the system under test.

With `Read Measurement` parameter **dB units** set to dB on, the SINAD is calculated as shown here:

$$SINAD = 20 \times \log \sqrt{\frac{E}{E - A_1^2}}$$

Swept Harmonics

Up to five individual harmonic distortion measurements can be returned. Each one is calculated as shown in the following formula:

$$HD_m = 100\% \times \frac{A_m}{A_1}$$

Where A_m = RMS value of the *m*th harmonic (*m* is from 2 to 6), and A_1 = RMS value of the fundamental.

NI-DSA computes swept harmonics on channel 1, which is normally connected to the output of the system under test.

With `Read Measurement` parameter **dB units** set to dB on, the distortion of the individual harmonic is calculated as shown here:

$$HD_m = 20 \times \log \frac{A_m}{A_1}$$

RMS Squared

This measurement returns the RMS squared value of the measured signal. The DC component of the signal is excluded from the RMS squared calculation.

NI-DSA computes this measurement for channel 1, which is normally connected the to output of the system under test.

Octave Analyzer (Add-On)

In Octave analyzer mode, your NI 45*XX* can perform fractional-octave analysis and level measurements in real time. The basic analyzer architecture, features, and operation are discussed in this section.

The NI-DSA Real-Time Octave Analysis (RTOA) software add-on for NI-DSA supports:

- Fractional-octave analysis for 1/1, 1/3 and 1/12 octave analysis, with the following averaging modes:
  - Linear
  - Exponential—slow, fast, custom
  - Impulse
  - Peak Hold
  - Equal Confidence
- Level measurements, with the following averaging modes
  - Linear (Leq)
  - Exponential—slow, fast, custom
  - Impulse
  - Impulse-Eq
  - Peak Hold
- A-, B-, and C-weighting

With the RTOA, you can do octave, level, or both octave and level measurements, and have the additional option of specifying the weighting if required.

All octave and level measurements are returned as RMS (root-mean-square) values. The RMS value of a signal is a measure of the energy it contains. In terms of vibration signals, it can be viewed as the destructive ability of the signal. Level measurements return the total amount of energy contained in the signal, and octave measurements return the amount of energy contained in different frequency bands.

Octave and Level Analyzers Software Architecture

Figure 6-7 shows the overall architecture for making octave and level measurements.

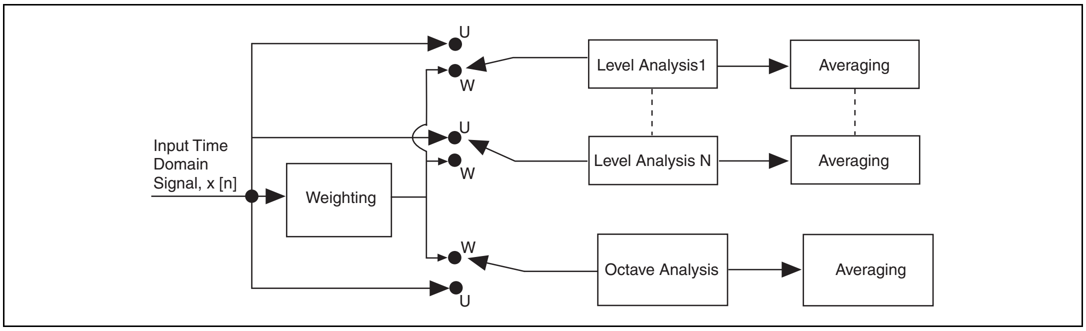

Figure 6-7. Octave and Level Measurement Functional Diagram

You can perform octave and level measurements on one, two, or four channels simultaneously. You can configure and perform up to eight simultaneous level measurements, but only one set of octave measurements. The time-domain input signal is passed through a weighting process if desired, then it is passed to the level analyzer and fractional-octave analyzer, and finally goes through an averaging process.

The input to either the octave or level analyzers can be either the weighted (W) or unweighted (U) time domain signal. For any single channel, it is possible to simultaneously obtain either a weighted or unweighted octave measurement—but not both—plus weighted, unweighted, or both weighted and unweighted level measurements. This is because, for each channel, only one set of octave measurements is returned, but it is possible to have up to eight simultaneous level measurements.

Architecture of the Fractional-Octave Analyzer

Figure 6-8 shows the architecture of the fractional-octave analyzer. The 1/3-octave analyzer is shown; it has three bandpass filters per octave. The full-octave (1/1-octave) analyzer has only one bandpass filter per octave. Similarly, the 1/12-octave analyzer has 12 bandpass filters per octave.

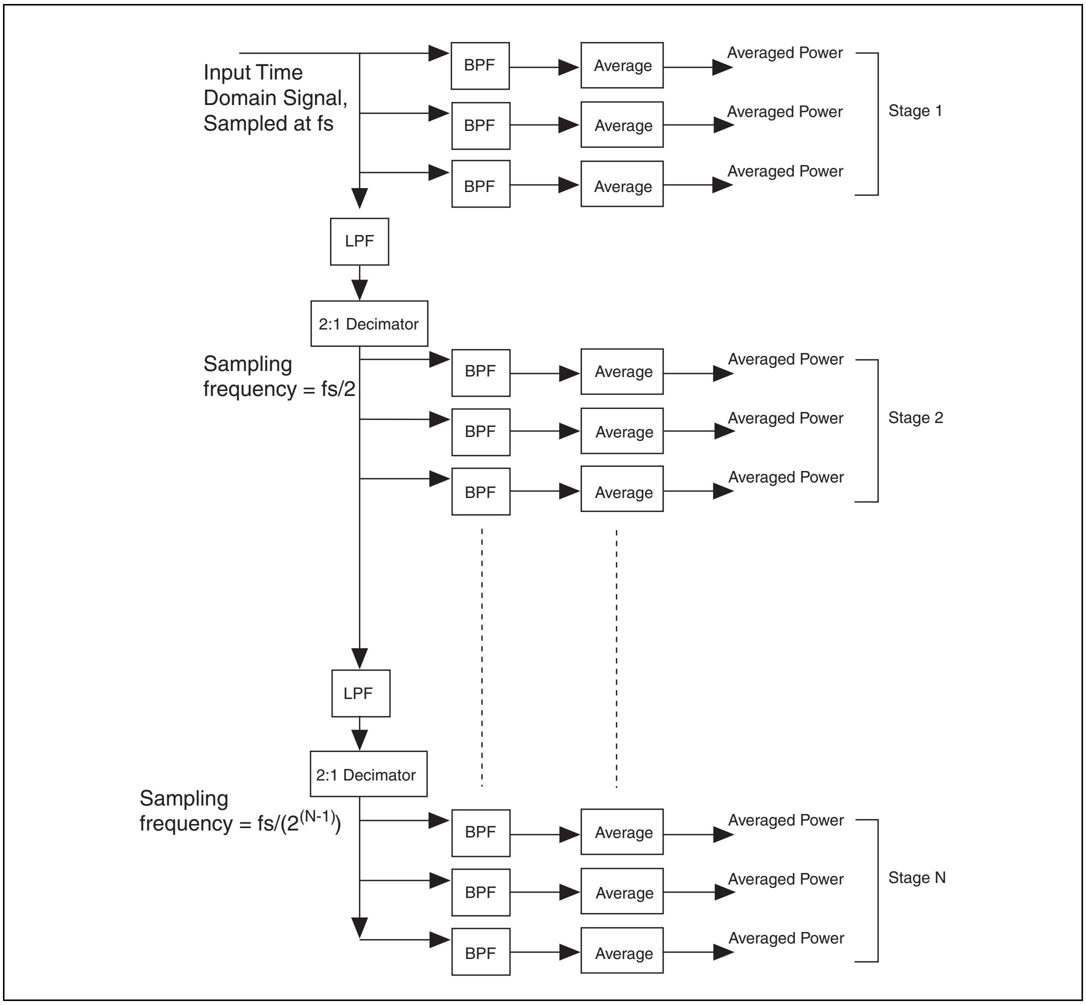

Figure 6-8. Fractional-Octave Analyzer Architecture

The input time-domain signal is sampled at a sampling frequency f_s. The Band Pass Filters (BPFs) in stage 1, corresponding to the highest octave, filter the signal directly at the sampling frequency. The signal is then passed through a digital low pass filter (LPF) to remove higher frequencies, then to a 2:1 decimator, which effectively reduces the sampling frequency to half ($f_s/2$). That is, for every two samples input, one sample is output. The lowpass-filtered and decimated signal is then passed to the BPF(s) corresponding to the next highest octave in stage 2. The process is repeated for a maximum of 10 stages of BPF(s), or a maximum of nine stages of lowpass filtering and decimation.

The time-domain samples at the output of each BPF are squared, then averaged according to the averaging mode selected, and the averaged output of each BPF is returned as the measurement. Third-octave analysis provides a maximum of 30 results (10 octaves × 3 BPFs/octave). 1/1-octave analysis provides a maximum of 10 results (10 octaves × 1 BPF/octave), and 1/12-octave analysis provides a maximum of 120 results (10 octaves × 12 BPFs/octave).

If you analyze a frequency range of less than 10 octaves, you get less than the maximum number of results. If you analyze a frequency range of more than 10 octaves, the analyzer only returns the number of results for 10 octaves.

Exact and Preferred Frequencies

You can choose for the bandpass filters for fractional-octave analysis to conform to either of the following standards:

- *ANSI S1.11-1986: Specification for octave-band and fractional-octave-band analog and digital filters.*
- *IEC 1260 (1995 - 07): Electroacoustics - Octave-band and fractional-octave-band filters.*

Since the NI-DSA fractional-octave analyzer has been designed to operate over the audio frequency range, the reference frequency (f_r) chosen is 1 kHz. The exact midband frequencies, f_m, for each BPF are calculated according to this formula:

$$f_m = f_r \times 2^{(k/b)}$$

where k is an integer, and $b = 1$ for 1/1-octave analysis, $b = 3$ for 1/3-octave analysis, and $b = 12$ for 1/12-octave analysis.

For example, for a 1/3-octave analysis, for $k = 0$, you have $f_m = f_r \times 2^{0/3} =$ 1000 Hz. For $k > 0$, the calculated values of f_m are above 1000 Hz, and for $k < 0$, the calculated values of f_m are below 1000 Hz.

The standards also specify what are known as "preferred" frequencies for 1/1-octave and 1/3-octave analysis. Preferred frequencies are not specified for 1/12-octave analysis. The exact and preferred frequencies for 1/3-octave analysis, for different values of k, are shown in the Table 6-2. The values in bold type are exact and preferred frequencies for 1/1-octave analysis, and the corresponding k values.

Table 6-2. Exact and Preferred Octave Analysis Frequencies

k **(1/3-octave)**	**Exact Frequency (Hz)**	**Preferred Frequency (Hz)**	***k*** **(1/1-octave)**
–19	12.4	12.5	—
–18	**15.62**	**16**	**–6**
–17	19.69	20	—
–16	24.8	25	—
–15	**31.25**	**31.5**	**–5**
–14	39.37	40	—
–13	49.61	50	—
–12	**62.5**	**63**	**–4**
–11	78.75	80	—
–10	99.21	100	—
–9	**125**	**125**	**–3**
–8	157.49	160	—
–7	198.43	200	—
–6	**250**	**250**	**–2**
–5	314.98	315	—
–4	396.85	400	—
–3	**500**	**500**	**–1**
–2	629.96	630	—
–1	793.7	800	—
0	**1,000**	**1,000**	**0**
1	1,259.92	1,250	—
2	1,587.40	1,600	—
3	**2,000**	**2,000**	**1**
4	2,519.84	2,500	—

Table 6-2. Exact and Preferred Octave Analysis Frequencies (Continued)

k (1/3-octave)	Exact Frequency (Hz)	Preferred Frequency (Hz)	*k* (1/1-octave)
5	3,174.80	3,150	—
6	**4,000**	**4,000**	**2**
7	5,039.68	5,000	—
8	6,349.60	6,300	—
9	**8,000**	**8,000**	**3**
10	10,079.37	10,000	—
11	12,699.21	12,500	—
12	**16,000**	**16,000**	**4**
13	20,158.74	20,000	—

Figure 6-9 shows the preferred midband frequencies for some of the BPFs of the 1/3-octave analyzer. These BPFs are constant-Q filters; the bandwidth of the filter increases with the increase in the midband frequency such that the ratio f_m/bandwidth = Q = a constant value. All the bandpass filters have the same value of Q.

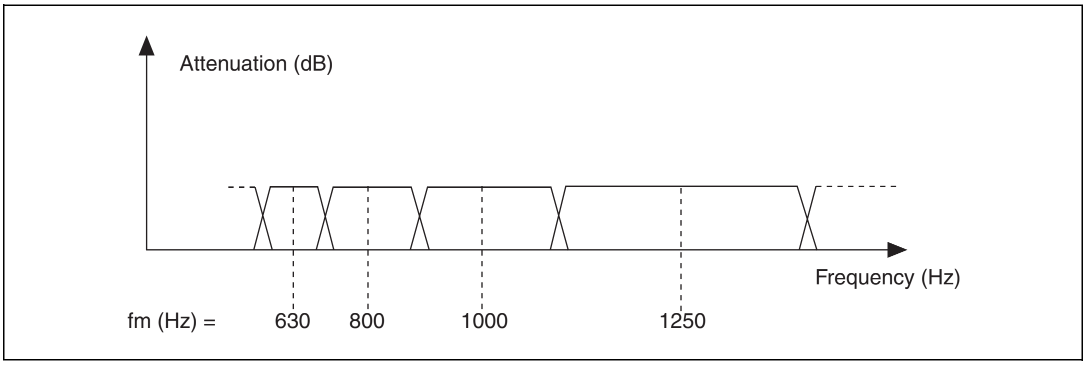

Figure 6-9. Preferred Midband Frequencies for 1/3-Octave BPFs

Summary of NI-DSA Octave Mode Functions

The functions are divided into several categories, depending on their operation. These categories are:

- **Configuration**—functions that configure the instrument for parameters such as the frequency range or the averaging mode. (Example: `Configure Octave Measurements`)
- **Control**—functions that control the operation of the instrument such as restarting averaging for a particular channel. (Example: `Restart OLM Averaging`)
- **Read**—functions that read measurements such as octave band powers, octave preferred or exact frequencies, or level measurements. (Example: `Read Octave Measurements`)
- **Status**—functions that return the status of the instrument, such as whether or not an overload has occurred at the input, is the instrument operating in real-time, or the time that has elapsed since averaging was begun. (Example: `Get OLM Status`)

Figure 6-10 is a flow diagram for level measurements.

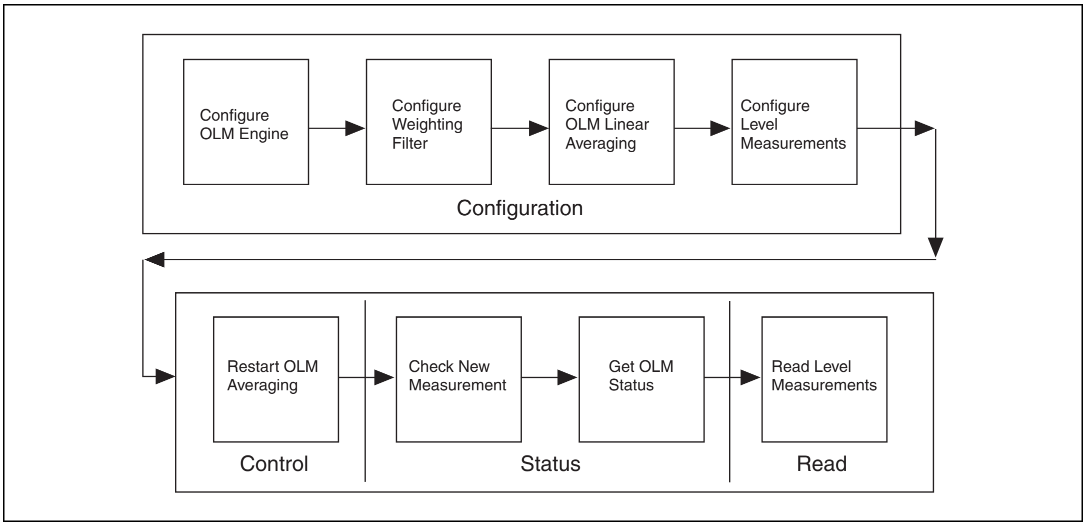

Figure 6-10. Level Measurement Functional Block Diagram

Usually, the configuration functions are called one time before measurements are started, then the status and read functions are repeated in a loop. However, you can program the operation of the instrument to meet the specific needs of your application.

The functions can also be classified according to the types of measurements they are used for. In general, a function used for both octave and level measurements has the abbreviation "OLM" in the function name. A function used only for octave measurements has the word "octave" in its name, and one used only for level measurements has "level" in its name. The categories of functions classified this way are as follows:

- Global Instrument-Wide Functions—apply to the entire device, for all channels, and for both octave and level measurements. Examples of global instrument-wide functions include:
  - `Set OLM Sampling Rate`
  - `Configure Weighting Filter`
- Octave and Level Measurement Functions—apply to both octave and level measurements. The following are octave and level measurement functions:
  - `Configure OLM Engine`
  - `Configure OLM linear averaging`
  - `Restart OLM averaging`
  - `Get OLM status`
- Octave Measurement functions—apply only to octave measurements, and include the following:
  - `Configure Octave Measurements`
  - `Configure Octave Averaging`
  - `Configure Octave Weighting`
  - `Get Octave Frequencies`
  - `Get Octave Number of Bands`
- Level Measurement functions—apply only to level measurements, and include the following functions:
  - `Configure Level Measurements`
  - `Get Level Labels`
  - `Get Level Length`
  - `Read Level Measurements`

Level Measurements

Sound pressure level measurements are defined by two main parameters. The first of these is the type of frequency weighting (A, B, C, or none). The second parameter sets the averaging scheme. To define the level averaging mode, you should call `Configure Level Measurements`.

Seven averaging schemes are available.

- F—Fast Exponential
- S—Slow Exponential
- IQ—Impulse Exponential
- C—Custom Exponential
- EQ—Equivalent Continuous Level
- IEQ—Impulse Equivalent Continuous Level
- P—Peak

The parameters for a level measurement are often designated as L*XY*, where *X* is the weighting filter type and *Y* is the averaging type. For instance, LAEQ indicates an A-weighted, Equivalent Continuous Level measurement.

The four exponential averaging modes weigh recent data more heavily than older data in calculating the level. The time weighting curve is an exponential decay with a definite time constant. For instance, the fast exponential mode has a 125 ms time constant. This means that data from 125 ms in the past is weighted 50% as much as the most recent data in calculating the sound level. The time constant for slow exponential mode is 1 s. Impulse mode features a 35 ms time constant for rising signals and 1500 ms for falling signals. Custom exponential mode allows you to define your own time constant.

The equivalent continuous level mode employs linear averaging over a finite block of time. All time-domain data in this block is weighted equally in computing the level. If you choose to employ the equivalent continuous level mode, make a call to `Configure OLM Linear Averaging` to specify the duration of the averaging block.

Impulse equivalent averaging mode actually combines two types of averaging. The first step is to find a level value using the impulse exponential. This value is then linearly averaged with past impulse exponential results. As with equivalent continuous mode, you specify the length of the linear average time block in `Configure OLM Linear Averaging`.

Peak averaging mode is not truly an averaging operation. Instead, it computes the level by recording the largest instantaneous level measurement since the beginning of the acquisition.

When acquiring level values, you should first call `Get Level Length` to get the number of level measurements currently available. Next, make a call to `Read Level Measurements` to produce an array containing the level values. You can also call `Get Level Labels` to generate a string array containing the corresponding until labels for the level measurements.

The octave analyzer engine includes RMS meters that allow you to perform level measurements on the unweighted or weighted time-domain waveform.

Averaging Modes

You can choose between several averaging modes for both octave and level measurements. Averaging of octave measurements is performed on the filtered outputs of the bandpass filters, whereas averaging for level measurements is performed on the input time domain data. For both octave and level measurements, averaging is done on the power (magnitude squared) value.

An explanation of the various types of averaging modes available are:

- None—In this case, no averaging is done. The result of each measurement is made available without combining it with any other measurement.
- Linear—For linear averaging, a weighted average is performed over the time duration specified in `Configure OLM Averaging`. All measurements made over the time duration are weighted equally.

Linear Averaging

The formula for linear averaging is as follows:

$$y[k] = a1 \times y[k-1] + a2 \times x[k],$$

where $x[k]$ is the new measurement, $y[k]$ is the new average, $y[k-1]$ is the previous average, $a1 = (k-1)/k$, $a2 = 1/k$, and k is an integer equal to the number of measurements averaged so far. Linear averaging returns the arithmetic mean of the measurements.

There are two types of linear averaging available, one-shot and auto-restart. In one-shot mode, linear averaging is performed on all measurements for the specified duration, then the averaging stops. All of the averages calculated before the final average are also available.

In auto-restart mode, as in one-shot mode, linear averaging is done for a specified duration, then the average is reset and averaging in performed on the next set of measurements for the same duration. This process continues indefinitely. In auto-restart mode, the results are available only for the end of each averaging period, that is, each result is the linear average of the measurements for the specified duration. One-shot and auto-restart modes are illustrated in Figure 6-11.

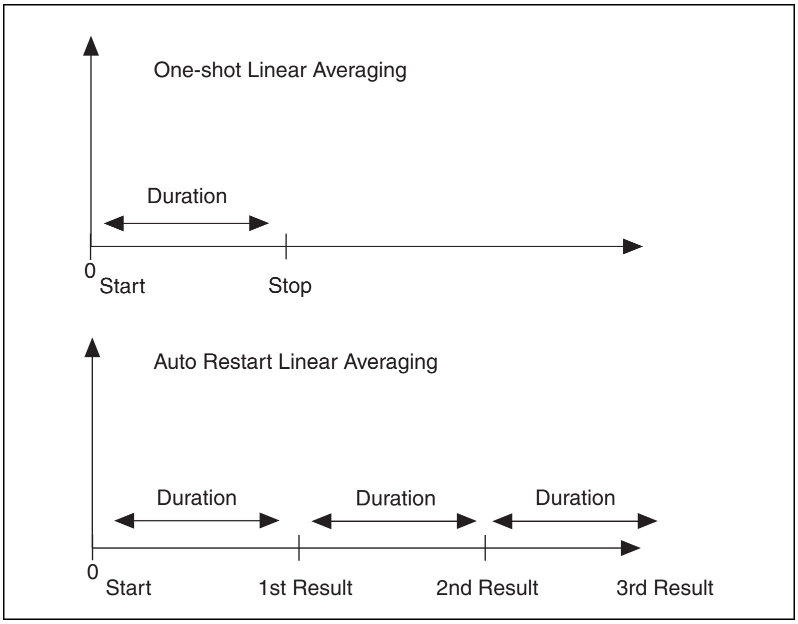

Figure 6-11. One-Shot and Auto-Restart Linear Averaging

While linear averaging is in progress, the elapsed time for the current averaging process is returned as the **linear averaging seconds** parameter of `Read Octave Measurements` and `Read Level Measurements`. There is a one-to-one correspondence between the time returned and the measured values read from these functions. In one-shot mode, the value returned is the actual value in seconds, in auto-restart mode the time returned is a multiple of the **duration** parameter of `Configure OLM Averaging`.

Note In one-shot mode, you can specify any time duration. If the end of the time duration falls between sample intervals, the closest sample is included in the duration. In auto-restart mode, the time duration can only be an integer multiple of 10 milliseconds. If any other duration is specified, it is rounded off to the nearest 10 milliseconds.

For level measurements, the linear average is denoted as Leq (equivalent sound power over the specified time duration).

Exponential Averaging

Exponential averaging is a continuous averaging process that weighs both current as well as past data. The amount of weight given to past data as compared to current data depends on the exponential time constant. The higher the time constant, the more weight given to past data, whereas the lower the time constant, the more weight is placed on the current data. The formula for exponential averaging is as follows:

$$y[k] = a1 \times y[k-1] + a2 \times x[k]$$

Where $x[k]$ is the new measurement, $y[k]$ is the new average, $y[k-1]$ is the previous average, $a1 = \exp(-1.0/time\_constant \times sampling\_rate)$, and $a2 = 1 - a1$.

The value of the *time_constant* depends on the type of exponential averaging selected. The various types of exponential averaging available are:

- Slow—the time constant for slow exponential averaging is 1.0 second. This type of averaging is useful for tracking the energy levels of signals whose energy levels vary slowly.
- Fast—the time constant for fast exponential averaging is 0.125 seconds. This type of averaging is useful for tracking the energy of signals whose energy levels are quickly varying.
- Custom—you can specify the time constant suitable for your particular application.
- Impulse—the time constant for signals with increasing energy levels is 0.035 seconds, whereas for signals with decreasing energy levels is 1.5 seconds. This type of averaging is useful for tracking sudden changes in the signal and holding on to these changes so that they do not disappear too fast.

While exponential averaging is in progress, the time for which the averaging process has elapsed is returned in the **averaging seconds** output of `Read Octave Measurements` and `Read Level Measurements`. There is a one-to-one correspondence between the time returned and the measured values read from these functions.

Impulse + Linear Averaging

Impulse + linear averaging is available only for level measurements. It is a combination of linear averaging and exponential impulse averaging. Impulse averaging is performed over the time duration specified in `Configure OLM Averaging`, and the corresponding measured values are weighted equally to form the average. Like linear averaging, the averaging process stops at the end of the specified time duration.

While impulse + linear averaging is in progress, the time for which the averaging process has elapsed is returned in the **linear averaging seconds** output of `Read Octave Measurements` and `Read Level Measurements`. There is a one-to-one correspondence between the time returned and the measured values read from these functions. In auto-restart mode, the time returned increments in intervals of the **duration** specified in `Configure OLM Averaging`.

Peak Averaging

In peak (or peak-hold) averaging, the largest measured energy value is held. The formula for peak averaging is as follows:

$$y[k] = \max(y[k-1], x[k])$$

Where $x[k]$ is the new measurement, $y[k]$ is the new average, and $y[k-1]$ is the previous average.

As with exponential averaging, the averaging process continues indefinitely.

Strictly speaking, peak hold is not really a form of averaging because successive measurements are not mathematically averaged. However, as with other averaging processes, it combines the results of several measurements into one final measurement, and could thus be considered a kind of averaging.

Equal Confidence Averaging

Equal confidence averaging is available only for octave measurements. It is similar to exponential averaging, the difference being that the time constant is dependent on the bandwidth of the filter. The time constant for each band is such that the results are within +/– the confidence level (in dB, specified in the **confidence level** input of `Configure Octave Averaging`) of the correct mean with a probability of 68%. The probability that the results are within twice the confidence level of the correct mean is 96%.

For example, if you specify a confidence level of 1 dB, and the correct mean is 10 dB, there is a 68% confidence that the measured value is between 9–11 dB, and a 96% confidence that the measured value is between 8–12 dB.

Table 6-3 shows the modes that are available for each of the octave or level measurements.

Table 6-3. Averaging Modes for Octave and Level Measurements

Averaging Mode	Octave	Level
None	Y	—
Linear	Y	Y
Exponential – slow	Y	Y
Exponential – fast	Y	Y
Exponential – custom	Y	Y
Exponential – impulse	Y	Y
Impulse + linear	—	Y
Peak hold	Y	Y
Equal confidence	Y	—

A-, B-, and C-Weighting

The weighting curves correspond to the response of the human ear at different levels of sound pressure. Specifically, A-, B-, and C-weighting correspond to the response at levels of 40 dB, 70 dB, and 100 dB respectively.

Considerations for Octave and Level Measurements

The NI 45*XX* uses an embedded digital signal processor to do most octave and level processing in real time, that is, all samples from the ADC are processed as they arrive at the processor, and none are lost. However, it is possible to exceed the capacity of the processor to handle data in real time. The ability of the processor to keep up with the incoming data depends on the following factors:

- Sampling frequency and number of input channels—the default sampling rate of the NI 45*XX* in octave analyzer mode is 51.2 kHz. You can use `Set OLM Sampling Rate` to set the sampling rate to one of two values: 25.6 kHz or 51.2 kHz. The NI 4551 has two input channels, and the NI 4552 has four input channels. The higher the number of channels used, the more difficult it is for the processor to keep up in real-time. With an NI 4552, the more input channels that you use for fractional-octave analysis, the lower the highest center frequency analyzed must be in order for real-time processing to be possible. The NI 4552 is unable to perform 1/3-octave analysis on all four channels at the default sampling rate, even with the highest center frequency set to 4 kHz. If you use three or more channels, use `Set OLM Sampling Rate` to set the sampling rate to 25.6 kHz, so you can process all four channels with a high center frequency of up to 10 kHz.

- Fractional-octave mode (1/1, 1/3 or 1/12), frequency range, and averaging mode—1/12-octave analysis requires the most processing and 1/1-octave the least. A higher center frequency requires more processing than a lower one. In general, impulse-exponential averaging requires more processing than the other exponential averaging modes. Linear averaging and peak hold averaging are the least computationally intensive of the averaging modes.
- Number of level measurements—The higher the number of level measurements per channel, the more processing time required.
- Weighting (A-, B-, or C-weighting)—A-weighting is most computationally intensive, and C-weighting the least.
- Simultaneous streaming to disk—If your instrument is configured for simultaneous streaming to disk, there is less processor time available for performing octave and level measurements.

Sometimes, you must eliminate one or more of these factors to get real-time performance. For example, you can usually get real-time fractional-octave analysis at higher center frequencies by choosing fewer channels and limiting, or eliminating, simultaneous level measurements or streaming I/O.

The maximum center frequencies for octave analysis, with no averaging or level measurements, are shown in Table 6-4.

Table 6-4. Maximum Center Frequencies for Octave Analysis

Number of Channels	Octave Analysis Type	Maximum Center Frequency per Channel
1	1/1-octave	16 kHz
2	1/1-octave	16 kHz
4	1/1-octave	8 kHz
1	1/3-octave	20 kHz
2	1/3-octave	20 kHz
4	1/3-octave	10 kHz

Configuration Limits

- NI 4551

Table 6-5 shows different configurations for which real-time processing on the NI 4551 is possible, and illustrates trade-offs that you can make. The table is not exhaustive, but a guide; other configurations are possible. For this table, the following assumptions are made: the sampling rate is the default rate, 51.2 kHz, and both input channels are processed.

Table 6-5. Real-Time Octave Analysis Configuration Guide for NI 4551

Fractional -Octave	Fractional -Octave Averaging Mode	Number of Level Measurements	Level Measurements Averaging Mode	Weighting Mode	Highest Fractional-Octave Center Frequency
1/1	Impulse	8	Impulse-Eq	A	16 kHz
1/3	Impulse	8	Leq	A	12.5 kHz
1/3	Impulse	8	Leq	A	16 kHz
1/3	Impulse	2	Leq	A	20 kHz
1/3	Fast	8	Leq	A	20 kHz
1/12	Impulse	0	—	None	5,339 Hz (ANSI) 5,496 Hz (IEC)
1/12	Fast	0	—	A	5,339 Hz (ANSI) 5,496 Hz (IEC)
1/12	Fast	8	Leq	A	4,490 Hz (ANSI) 4,621 Hz (IEC)

As Table 6-5 shows, the NI 4551 can perform a 1/1-octave analysis in real time while applying the most computationally-intensive averaging and weighting (impulse and A, respectively). In addition, it can simultaneously perform up to eight level measurements with impulse-eq averaging.

For 1/3-octave analysis with impulse averaging, you can gain a higher center frequency by sacrificing the number of level measurements, but you can keep both a high center frequency and up to eight level measurements by using exponential-fast averaging instead of impulse averaging.

For 1/12-octave analysis using impulse averaging, the highest center frequency possible is 5,339 Hz (ANSI) or 5,496 Hz (IEC), without weighting and without level measurements. You can add A-weighting, but to keep the highest center frequency the same, you must change to a less processor-intensive averaging mode. If you add eight level measurements per channel, the highest possible center frequency drops to 4,490 Hz (ANSI) or 4,621 Hz (IEC).

Note To run the 1/12-octave analyzer to even higher center frequencies, use `Set OLM Sampling Rate` to reduce the sampling frequency to 25.6 kHz.

♦ NI 4552

The NI 4552 has four input channels, so the number of channels configured for octave measurements can be a more important factor in processor performance than it is with a NI 4551.

If you configure one or two channels for octave analysis, configurations that allow real-time operation of the NI 4551 generally will for the NI 4552 as well. However, if you configure three or four channels for octave analysis, the additional processing demands reduce the frequency range over which the processor can operate in real-time. To get a higher frequency range, make the following changes to your configuration:

- Use `Set OLM Sampling Rate` to set the sampling rate to 25.6 kHz
- Use `Configure OLM Engine` to turn off level measurements

Using the fast averaging mode, these changes enable you to set the highest center frequency at 8 kHz for 1/1-octave analysis, 10 kHz for 1/3-octave analysis, and 2,997 Hz (ANSI) or 3,084 Hz (IEC) for 1/12-octave analysis.

Table 6-6 shows different configurations for which real-time processing on the NI 4552 is possible, and illustrates trade-offs that you can make. The table is not exhaustive, but a guide; other configurations are possible. For this table, the following assumptions are made: the sampling rate is set at 25.6 kHz, all four input channels are used, and no level measurements are made.

Table 6-6. Real-Time Octave Analysis Configuration Guide for NI 4552

Fractional-Octave	Fractional-Octave Averaging Mode	Weighting	Highest Fractional-Octave Center Frequency
1/1	Impulse	A	8 kHz
1/3	Impulse	A	10 kHz
1/12	Impulse	A	2,520 Hz (ANSI) or 2,594 Hz (IEC)
1/12	Impulse	None	2,828 Hz (ANSI) or 2,911 Hz (IEC)
1/12	Fast	None	2,997 Hz (ANSI) or 3,084 Hz (IEC)

With computationally intensive impulse averaging and A-weighting, both 1/1-octave and 1/3-octave analysis are possible on all four channels simultaneously, to a maximum center frequency of 8 kHz and 10 kHz respectively.

For 1/12-octave analysis using impulse averaging and A-weighting, the maximum center frequency can be as high as 2,378 Hz (ANSI) or 2,448 Hz (IEC) for real-time processing. If you sacrifice the weighting, change to exponential fast averaging mode, or both, the maximum center frequency can be higher.

Frequency Ranges

For each fractional-octave mode, Table 6-7 summarizes the lowest and highest center frequencies for a single channel at different sampling rates. For both 1/1-octave and 1/3-octave, the frequencies are specified as preferred frequencies from the corresponding ANSI and IEC standards. For 1/12-octave, they are specified in terms of the exact center frequencies.

Table 6-7. Octave Analyzer Frequency Ranges for a Single Channel

Analysis Type	Center Frequencies at f_s = 25.6 kHz	Center Frequencies at f_s = 51.2 kHz
1/1-octave	High: 8 kHz Low: 16 Hz	High: 16 kHz Low: 31.5 Hz
1/3-octave	High: 10 kHz Low: 12.5 Hz	High: 20 kHz Low: 25 Hz
1/12-octave	High: 10,678.72 Hz (ANSI) Low: 11.05 Hz (ANSI) High: 10,991.63 Hz (IEC) Low: 11.37 Hz (IEC)	High: 5,339.36 Hz (ANSI) Low: 5.52 Hz (ANSI) High: 5,495.81 Hz (IEC) Low: 5.69 Hz (IEC)

Settling Time

The fractional-octave filters require a certain amount of time before any valid measurement is available at the outputs of the filter. This time is the *settling time* and is inversely proportional to the bandwidth of the filter according to the formula *settling time* = 1/(5 × *bandwidth*), where the settling time is in seconds and the bandwidth is in hertz (Hz).

The bandwidth for each band can be calculated by the formula *bandwidth* = *exact center frequency* × ($2^{1/2n} – 2^{–1/2n}$), where n = 1 for full-octave, 3 for 1/3-octave, and 12 for 1/12-octave, and the exact center frequency for each band is returned in the **exact center frequencies** returned from `Get Octave Frequencies`.

The lower frequency bands require more settling time because of their smaller bandwidth. You can find out if the octave filters have settled by checking the **settled** parameter returned by `Read Octave Measurements`.

If both octave and level measurements are configured in `Configure OLM Engine`, the averaging process is started only after all the octave filters have settled. If only level measurements are configured, then the averaging process is started immediately.

Overload Detection

Two types of overload detection flags, *sticky* or *non-sticky*, are used by the NI-DSA software. A sticky overload flag is set when an overload is detected on a particular channel, remains set after the overload condition disappears, and is not reset until you reconfigure, or restart averaging on, that channel. `Get OLM Status` returns a sticky overload status.

A non-sticky overload flag is set when an overload is detected on any input channel, and is reset automatically when the overload condition passes. `Check status` returns non-sticky overload status. The overload information returned by `Check Status` is bit-encoded as shown in Table 6-8.

Table 6-8. Overload Detection Bit-Encoding

Value Returned by `Check Status`	Bit Pattern (LSBs)	Channels with Overload Condition
0	0000	None
1	0001	1
2	0010	2
3	0011	1 and 2
4	0100	3

Table 6-8. Overload Detection Bit-Encoding (Continued)

Value Returned by `Check Status`	**Bit Pattern (LSBs)**	**Channels with Overload Condition**
5	0101	1 and 3
6	0110	2 and 3
7	0111	1, 2, and 3
8	1000	4
9	1001	1 and 4
A	1010	2 and 4
B	1011	1, 2, and 4
C	1100	3 and 4
D	1101	1, 3, and 4
E	1110	2, 3, and 4
F	1111	1, 2, 3, and 4

A Common Questions

FFT Mode

FFT Analysis Questions

What is the maximum number of bins (lines), for a baseband FFT measurement using NI-DSA?

The maximum number of lines is 800 for a classical span and 950 for an extended span.

What is the difference between a classical baseband span and an extended baseband span?

A *classical span* is one with a ratio of one frequency line for every 2.56 data samples. In an ideal analyzer with perfect brick-wall antialiasing, the number of useful frequency lines would be exactly half the number of samples—a 2,048-point FFT would yield 1,024 frequency lines. Most analyzers have filters that significantly attenuate frequencies near integer multiples, so a 2,048-point FFT actually produces 800 useful frequency lines. With NI-DSA, you can choose FFTs with classical spans of 50, 100, 200, 400, or 800 lines.

The NI 455*X* devices feature anti-alias filters with a very sharp roll-off. Because of this feature, the devices can push closer to ideal behavior and provide more useful lines without requiring additional samples. This feature is called the extended span. Extended-span FFTs provide 59, 118, 237, 475, or 950 frequency lines without the need for additional data points, higher sampling rates, or a longer acquisition time.The number of frequency bins in an FFT depends on the number of time-domain points you analyze. To increase the frequency resolution of an FFT, you must sample over a longer time period.

Why does a 400-line baseband FFT return an array with 401 data points?

By definition, a 400-line FFT contains information on the spectral content of a signal at 400 nonzero frequencies. The analyzer also returns the DC content of the signal, providing a total of 401 amplitude values. An x-line FFT always returns an array of length x+1, and the first element of this array is the amplitude of the DC signal component.

What is real-time operation for a 455*X* DSA device?

When referring to a NI 455*X* DSA device, real-time operation simply means that all time-domain data points are used in the frequency-domain calculations you request. Another way to express this is that the analyzers on the DSA device can keep up with the incoming time-domain data points. If real-time operation is not maintained, the analyzers skip some time-domain data points because they cannot perform their computations fast enough to include them all. Failure to maintain real-time operation does not generate an error or software failure.

How do I verify that I am maintaining real-time operation?

Make a call to `Check Status` after performing a read operation to verify that you are running in real-time. This function returns an array of status values; the first element in this array is the real-time flag. A value of indicates that real-time has been maintained, and a value of zero indicates that it has not.

If your software does not make frequent calls to `Read Measurement`, you may not transfer all of your frequency domain measurements from the device analyzer to your host PC. This condition does not constitute a failure to maintain real-time, and does not assert the real-time warning flag in `Check Status`.

Why does my time-domain data sometimes look incorrect in FFT analyzer mode?

When you use `NI-DSA Read Measurement` to read in time-domain data, it important to set the **View** parameter on the function to Real. Viewing the time-domain data as magnitude values produces incorrect results. Also, you should not use the dB On variable for time-domain data.

On the other hand, most frequency-domain applications call for the magnitude or phase view options. dB on is often helpful for visualizing the magnitude response.

What are the benefits of capture mode?

Capture mode allows you to stream time-domain data points directly to a buffer in host RAM. Even if you do not read the time-domain data from the driver in real time, you do not lose any of the time-domain data. The most common application of capture mode is to stream the whole time-domain record to disk.

Capture mode is very similar to standard circular-buffered NI-DAQ functions in LabVIEW. You should use `Configure Capture Buffer` to reserve this memory space. Like standard DAQ functions, Capture mode generates an error if this buffer overflows. Read from the buffer periodically to prevent an overflow from occurring.

What do the terms *overlap* and *time increment* mean?

Overlap is a percentage ranging from 0–100%. It indicates the fraction of time domain data that is shared between two adjacent FFTs. For instance, consider two FFTs, each of which is 0.1 second in length. Assume the first FFT covers the block of time from T = 5.00 seconds to T = 5.10 seconds. The second FFT covers the block of time from T = 5.02 to T = 5.12 seconds. The share 80% of their points (those between T = 5.02 and T = 5.10). Thus, the FFT overlap is 80% in this case. Increasing, the overlap actually generates more FFT curves in any given time period.

Time increment is simply defined as 100% minus the overlap, so in the above example, the time increment is 20%. In software, you control overlap/time increment using `Configure Base FFT Settings` and `Configure Zoom FFT Settings`. Time increment is one of the input parameters for these functions—you should enter a number between 0–100. Enter a number greater than 100 to skip data points.

Zoom FFT Analysis Questions

What is the major benefit of using a zoom FFT?

The zoom FFT technique is very useful for improving the frequency resolution of an FFT measurement if you are interested in a limited range of frequencies. For instance, assume you want to analyze the frequencies from 5,000 Hz to 6,000 Hz. If you use an extended-range baseband span with 950 lines, you can look at all frequencies from 0–6,000 Hz. In this case, your frequency resolution is 6,000 Hz/950 lines = 6.32 Hz.

A zoom FFT allows you to examine a frequency starting at a value greater than zero. Using the zoom FFT, you can obtain up to 1,600 frequency resolution from 5,000 Hz to 6,000 Hz. The frequency resolution improves dramatically: 1000 Hz/1600 lines = 0.625 Hz.

Does the zoom FFT reduce acquisition time?

For any given frequency resolution, you must acquire data for the same time interval whether you are using baseband or zoom analysis. If you use either a zoom FFT or a baseband FFT, the time block of data required to perform a single FFT is 1/Df, where Df is the frequency resolution. In the example above, you require 1/0.625 Hz = 1.6 seconds of data for an FFT.

When using zoom FFTs, should I set the baseband span as well as the zoom span?

Yes. Set the classical baseband span to an appropriate value when performing a zoom analysis. The reason for this setting is that the baseband span settings control the sampling rate on the DSA hardware. For any particular zoom span, choose the *smallest* classical baseband span that meets the following three criteria:

1. The classical baseband span is equal to, or greater than, the maximum frequency of interest in the zoom span.
2. The classical baseband span is equal to, or greater than, the device minimum of 1,953.7 Hz. This minimum span corresponds to a sampling rate of 5 kS/s.
3. The ratio between the baseband span width and zoom span width is a power of 2.

For example, if you set a zoom span to examine frequencies between 600 Hz and 800 Hz, then the zoom span width is 200 Hz because you need a baseband span that produces a power of 2 when divided by 200 Hz. Possible values include 200 Hz, 400 Hz, 800 Hz, and so on, yet 800 Hz is the lowest possible span that is greater than, or equal to, the top of the zoom span. However, 800 Hz violates the device minimum span, so the smallest acceptable value also that fulfills the power of 2 requirement is 3,200 Hz. Thus, you call `Set Classical Baseband Span` with a span value of 3,200 Hz.

The NI-DSA shipping examples for performing zoom FFTs in LabVIEW and LabWINDOWS/CVI demonstrate a quick log-based algorithm to calculate the appropriate baseband span for any given zoom span.

Can I use a zoom FFT and set the lower frequency span limit at 0 Hz?

Yes. This operation is essentially a high-resolution baseband analysis although you still program it in zoom FFT mode. The minimum span for a baseband analyzer on the NI 455*X* DSA devices is just under 1,953.7 Hz, meeting the minimum sampling rate of 5 kHz. However, this minimum span width limitation does not apply to zoom mode. Using zoom allows you to increase the frequency resolution of your FFT both by increasing the number of lines to 1,600 and by using a smaller upper frequency limit.

If you use the zoom FFT analyzer with a lower limit of 0 Hz, the zoom algorithm skips the heterodyning step to shift frequencies but still performs decimation and filtering. This feature provides a slight improvement in the efficiency of the algorithm for case of a 0 Hz lower limit.

What are the *zoom time* and *zoom win time* measurements?

The zoom time measurement provides the most recent time-domain data after it has been heterodyned, filtered, and decimated by the zoom analyzer. The zoom win time shows the same calculation including the effect of the time-domain smoothing window. These are very specialized measurements that you should use only if you need to perform additional processing on the heterodyned data in software. For most applications requiring time domain data access, you should use either the time measurement or employ capture mode.

Source Mode

What types of analog output signals can I generate using NI-DSA?

NI-DSA includes functions to generate standard white noise, pink noise, band-limited noise, periodic white noise (PRN), single or dual-tone sine, and chirp signals.

Furthermore, you can use NI-DSA to define and generate an arbitrary analog output signal. It is important to notice that the data for this signal is stored entirely within the 4,096-point FIFO memory on the board. Because of this constraint, the number of point in your arbitrary data array must be a power of two equal to or less than 4,096. If you use the NI 455*X* in NI-DAQ compatibility mode, you can store a much longer analog output data array in the RAM of the host computer.

What is the relationship between the input and output sampling rates on the NI 455*X* device?

If you are performing simultaneous analog input and output with your NI 455*X* device, the ratio between the input and output sampling rates must always be a power of 2. This requirement is because the input and output converters are driven by the same DDS circuitry on the device. The hardware synchronization between input and output is convenient for performing stimulus-response tests. If you make subsequent calls to `Get`

`HW Sampling Rate` and `Get HW Update Rate`, you will see that the ratio between these two parameters is a power of 2.

What is the maximum sine output frequency I can obtain using NI-DSA?

The maximum output frequency you can generate with NI-DSA is 23 kHz. This limit is governed by the maximum output sampling rate of 51.2 kS/s. Because the digital-to-analog converters on the NI 455*X* DSA devices feature sharp roll-off, anti-imaging filters, you can generate a spectrally pure sine tone up to 23 kHz with minimal quantization noise, so when using the NI 455*X* devices, you do not need to worry about the output signal becoming choppy or noisy when approaching the Nyquist frequency.

Why does NI-DSA sometimes produce a slightly different output frequency than the one I request?

The sine signal generated by NI-DSA always consists of 4,096 data points. This buffer must contain an integer number of cycles of the sine wave to avoid adding unwanted noise to the signal. Also, the physical sample rate on the device is tied to the analog input rate. Under some conditions, these two constraints prevent NI-DSA from generating the precise analog frequency you request. If you are unsure about the frequency that is being generated at any time, you can query the software for the actual analog frequency by calling `Get Sine Source Settings`. In many cases, the DSA device generates the requested frequency exactly, and if it does not, the discrepancy is rarely exceeds a few hertz.

There are also a few cases in which NI-DSA is unable to coerce the output frequency to a nearby value and returns an error. For example, the baseband span is 12,000 Hz, corresponding to a hardware sampling rate of 30,720 S/s. You request an output sine frequency at 16,000 Hz, which requires an analog output sampling rate greater than 32,000 S/s because of the Nyquist frequency 30,720 S/s (the input rate) is not fast enough. Doubling this output sampling rate to a value greater than 64,000 S/s is also not feasible because the hardware supports a maximum output rate of 51,200 S/s. In this example, the best solution is to increase the input baseband span to 16,000 Hz or greater, speeding up the input sampling rate and allowing you to obtain higher output sampling rates.

Swept-Sine Mode

What is the advantage to using swept-sine analysis rather than a broadband FFT-based frequency response?

Swept-sine analysis provides two main advantages over a broadband frequency response measurement in which all frequencies are excited and measured simultaneously. The first is a wider dynamic range. NI-DSA can automatically adjust the input gain and output attenuation to maximize the dynamic range of your measurements. If your system's response at a particular frequency is very weak, the driver uses a higher gain setting on input. Likewise, NI-DSA can attenuate the output signal and reduce the input gain in when the response is strong to eliminate clipping.

The second advantage that swept-sine provides is flexible measurements of signal distortion and nonlinearity; quantifying nonlinear behavior is often important in audio applications. Swept-sine analysis allows you to measure distortion because it provides excitation at only a single discrete frequency at any given time. For example, assume you are exciting an amplifier with a 1 kHz tone. The basic response of the amplifier can be measured at 1 kHz, and the total harmonic distortion (THD) can be found by summing the response amplitudes at 2 kHz, 3 kHz, 4 kHz, and so on. This is not feasible with broadband excitation because there is no way to determine if a 2 kHz response signal is a harmonic from 1 kHz or if it is simply the linear response to excitation at 2 kHz.

When using swept-sine analysis, why does it take longer to test low frequencies than higher ones?

In order to measure the response at a particular frequency, NI-DSA must generate at least one complete sinusoidal cycle at that frequency. This cycle takes longer at low frequencies—1 second at 1 Hz, for example.

Is there a minimum frequency I can successfully test using NI-DSA swept-sine analysis?

The minimum test frequency for swept-sine analysis is 0.1 Hz.

Can I measure the level of individual harmonics (rather than THD) using swept-sine analysis?

Yes. You can obtain up to five harmonics from the swept-sine measurement. To set up the swept-sine harmonic analysis, call `Configure Swept Sine Harmonics`. Pass a value of `1` to **Enable Harmonic Mapping**. The

Harmonic Mapping parameter is an array of five integers defining the harmonics of interest. For example, if the elements of this array are [2,3,5,9,20], the swept-sine harmonic measurements returned are:

- Swept-sine harmonic 1—2nd harmonic
- Swept-sine harmonic 2—3rd harmonic
- Swept-sine harmonic 3—5th harmonic
- Swept-sine harmonic 4—9th harmonic
- Swept-sine harmonic 5—20th harmonic

The default value for the mapping is [2,3,4,5,6], giving you the first five harmonics after the fundamental frequency.

In some cases, you may not obtain a valid reading for the higher harmonics at all test frequencies. This error occur if the frequency of the harmonic exceeds 23,000 Hz. For instance, you cannot see the fourth or fifth harmonic when the excitation frequency exceeds 4,600s Hz. NI-DSA returns a harmonic level of –200 dB if the harmonic lies beyond the acquisition bandwidth.

How do the averaging settings work with swept-sine analysis?

You can use `Configure Swept Sine Average` to control how much time goes into a swept-sine measurement. Up to a point, longer test times can produce more accurate and repeatable frequency response measurements. `Configure Swept Sine Average` has four input arguments:

- Settle time—this value gives how many seconds should pass before each new frequency from the swept-sine list is generated. Since the units for this input are in seconds (absolute time), the length of this delay does not depend on the frequency being generated.
- Settle cycles—like settle time, this value also defines the length of the settling period. However, it is given in units of signal cycles rather than absolute time. This means that the corresponding time changes as the test frequency increases—smaller frequencies produce longer settling times.

Settle time and settle cycles both control the same physical parameter—the length of the settling period between measuring individual frequencies. At any given frequency, the software pauses for either settle time or (Settle Cycles * Frequency), which ever is greater. For example, assume settle time is 0.01 s, and settle cycles is 10 cycles. In this case, the physical settling time is (10 * Frequency) for all frequencies below 1 kHz, and 0.01 s for all greater frequencies.

- Integration time—this value, given in units of seconds, defines how long each frequency measurement should actually take. This is the time during which the DSA device generates an excitation, reads the response, and calculates the transfer function.
- Integration cycles—this is controls how many excitation signal cycles should go into a particular frequency response calculation.

As with the actual settling period, the calculation time at any frequency can be either settle time or (Settle Cycles * Frequency), which ever is greater. The total time for an entire swept-sine measurement is the sum of the settling time and integration time used at every frequency under test.

There is no quick formula for determining the best values for the swept-sine averaging parameters. For the most part these value should be determined empirically—start at large values for all parameters and reduce the values slowly until you see the final results of your frequency response test begin to change. In general, a longer settling period helps for measurements on systems whose response dies off slowly when excitation is ceased. Examples include high-Q resonant circuits or sound measurements in a fairly reverberant environment. A longer integration period helps if the unit under test contains some element of random noise—this should average out over the course of several cycles of measurement.

Octave and Level Analyzer Mode (Add-On)

How do I configure the duration for linear averaging?

To set the time interval for linear averaging for both octave and level measurements, use the **Duration** parameter of `Configure OLM Linear Averaging`. The same value is used for the duration of linear averaging for both octave and level measurements, and for all channels. You cannot set the duration for linear averaging of octave measurements independently of that for level measurements, nor can you set the duration of linear averaging independently for different channels.

How do I make level measurements?

To configure level measurements, use `Configure Level Measurements`. To read the level measurements, first call `Get Level Length` to determine the number of measurements to read, then call `Read Level Measurements` to read the measurements.

How do I configure weighting?

- Use `Configure Weighting Filter` to select the type of weighting filter (A-, B- or C-weighting). `Configure Weighting Filter` selects the weighting filter for the entire device, that is, for all channels, for both octave and level measurements.
- For octave weighting use `Configure Octave Weighting` to turn weighting on or off for a particular channel.
- For level weighting use `Configure Level Measurements` to turn weighting on or off for a particular channel and a particular level measurement.

Why can't I change the sampling frequency of the DSA device to arbitrary values when in octave mode?

To ensure compliance with existing ANSI and IEC standards, the sampling frequency of the device can be set to either 51,200 Hz or 25,600 Hz when in octave mode by using `Set OLM Sampling Rate`. It is not possible to change the sampling frequency to another value using `Set Hardware Sampling Rate`. You can read the current value of the sampling frequency by using `Get Hardware Sampling Rate`.

How long does the signal take to propagate through the input section of the NI 4551/4552?

The signal delay—the time it takes for a signal entering an analog input channel to become available as digital data at the output of the delta-sigma ADC—is 42 sample periods, regardless of the sample rate, for example—when using the octave mode and a sampling rate of 51.2 kS/s. In octave analyzer mode, the sampling frequency is fixed at 25,600 Hz or 51,200 Hz. The signal delay with a sample rate of 51,200 Hz is approximately 0.82 ms (42 S/51,200 S/s = 0.82 ms). Thus, it takes about 0.82 milliseconds for any change in the analog input signal to be available for processing by the fractional octave analyzer.

What are the center frequencies for 1/12-octave analysis?

The exact center frequencies depend on which standard, ANSI or IEC, you choose. The ANSI- and IEC-standard center frequencies for 1/12-octave analysis are as shown in Table A-1. For reference purposes, the table shows the center frequencies up to over 20 kHz. However, the center frequencies are limited to 10,678.72 Hz (ANSI) or 10,991.63 Hz (IEC) when the sampling rate is 25,600 Hz, and to 53,39.36 Hz (ANSI) or 5,495.81 Hz (IEC) when the sampling rate is 51,200 Hz.

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
–90	5.52	5.69
–89	5.85	6.02
–88	6.20	6.38
–87	6.57	6.76
–86	6.96	7.16
–85	7.37	7.59
–84	7.81	8.04
–83	8.28	8.52
–82	8.77	9.03
–81	9.29	9.56
–80	9.84	10.13
–79	10.43	10.73
–78	11.05	11.37
–77	11.71	12.05
–76	12.40	12.76
–75	13.14	13.52
–74	13.92	14.33
–73	14.75	15.18
–72	15.625	16.083
–71	16.554	17.039
–70	17.538	18.052
–69	18.581	19.126
–68	19.686	20.263
–67	20.857	21.468

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis (Continued)

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
–66	22.097	22.745
–65	23.411	24.097
–64	24.803	25.530
–63	26.278	27.048
–62	27.841	28.656
–61	29.496	30.360
–60	31.250	32.166
–59	33.108	34.078
–58	35.077	36.105
–57	37.163	38.252
–56	39.373	40.526
–55	41.714	42.936
–54	44.194	45.489
–53	46.822	48.194
–52	49.606	51.060
–51	52.556	54.096
–50	55.681	57.313
–49	58.992	60.721
–48	62.500	64.331
–47	66.216	68.157
–46	70.154	72.210
–45	74.325	76.503
–44	78.745	81.052
–43	83.427	85.872

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis (Continued)

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
–42	88.388	90.978
–41	93.644	96.388
–40	99.213	102.120
–39	105.112	108.192
–38	111.362	114.626
–37	117.984	121.441
–36	125.000	128.663
–35	132.433	136.313
–34	140.308	144.419
–33	148.651	153.007
–32	157.490	162.105
–31	166.855	171.744
–30	176.78	181.96
–29	187.29	192.78
–28	198.43	204.24
–27	210.22	216.38
–26	222.72	229.25
–25	235.97	242.88
–24	250.00	257.33
–23	264.87	272.63
–22	280.62	288.84
–21	297.30	306.01
–20	314.98	324.21
–19	333.71	343.49

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis (Continued)

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
–18	353.55	363.91
–17	374.58	385.55
–16	396.85	408.48
–15	420.45	432.77
–14	445.45	458.50
–13	471.94	485.77
–12	500	514.65
–11	529.73	545.25
–10	561.23	577.68
–9	594.60	612.03
–8	629.96	648.42
–7	667.42	686.98
–6	707.11	727.83
–5	749.15	771.11
–4	793.70	816.96
–3	840.9	865.54
–2	890.90	917.00
–1	943.87	971.53
0	1,000	1,029.30
1	1,059.46	1,090.51
2	1,122.46	1,155.35
3	1,189.21	1,224.05
4	1,259.92	1,296.84
5	1,334.84	1,373.95

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis (Continued)

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
6	1,414.21	1,455.65
7	1498.31	1,542.21
8	1,587.40	1,633.92
9	1,681.79	1,731.07
10	1,781.80	1,834.01
11	1,887.75	1,943.06
12	2,000	2,058.60
13	2,118.93	2,181.02
14	2,244.92	2,310.71
15	2,378.41	2,448.11
16	2,519.84	2,593.68
17	2,669.68	2,747.91
18	2,828.43	2,911.31
19	2,996.61	3,084.42
20	3,174.80	3,267.83
21	3,363.59	3,462.15
22	3,563.59	3,668.02
23	3,775.5	3,886.13
24	4,000	4,117.21
25	4,237.85	4,362.03
26	4,489.85	4,621.41
27	4,756.83	4,896.21
28	5,039.68	5,187.36
29	5,339.36	5,495.81

Table A-1. Exact Center Frequencies for 1/12 Octave Analysis (Continued)

k (1/12 octave)	ANSI-Standard Exact Center Frequencies (Hz)	IEC-Standard Exact Center Frequencies (Hz)
30	5,656.85	5,822.61
31	5,993.23	6,168.84
32	6,349.60	6,535.66
33	6,727.17	6,924.29
34	7,172.19	7,336.03
35	7,550.99	7,772.26
36	8,000	8,234.42
37	8,475.70	8,724.06
38	8,979.70	9,242.82
39	9,513.66	9,792.43
40	10,079.37	10,374.72
41	10,678.72	10,991.63
42	11,313.71	11,645.23
43	11,986.46	12,337.69
44	12,699.21	13,071.32
45	13,454.34	13,848.58
46	14,254.38	14,67206
47	15,101.99	15,545.51
48	16,000	16,468.84
49	16,951.41	17,448.12
50	17,957.39	18,485.64
51	19,027.31	19,584.86
52	20,158.74	20,749.43
53	21,357.44	21,983.26

B

Technical Support Resources

Web Support

National Instruments Web support is your first stop for help in solving installation, configuration, and application problems and questions. Online problem-solving and diagnostic resources include frequently asked questions, knowledge bases, product-specific troubleshooting wizards, manuals, drivers, software updates, and more. Web support is available through the Technical Support section of `ni.com`.

NI Developer Zone

The NI Developer Zone at `ni.com/zone` is the essential resource for building measurement and automation systems. At the NI Developer Zone, you can easily access the latest example programs, system configurators, tutorials, technical news, as well as a community of developers ready to share their own techniques.

Customer Education

National Instruments provides a number of alternatives to satisfy your training needs, from self-paced tutorials, videos, and interactive CDs to instructor-led hands-on courses at locations around the world. Visit the Customer Education section of `ni.com` for online course schedules, syllabi, training centers, and class registration.

System Integration

If you have time constraints, limited in-house technical resources, or other dilemmas, you may prefer to employ consulting or system integration services. You can rely on the expertise available through our worldwide network of Alliance Program members. To find out more about our Alliance system integration solutions, visit the System Integration section of `ni.com`.

Worldwide Support

National Instruments has offices located around the world to help address your support needs. You can access our branch office Web sites from the Worldwide Offices section of `ni.com`. Branch office Web sites provide up-to-date contact information, support phone numbers, e-mail addresses, and current events.

If you have searched the technical support resources on our Web site and still cannot find the answers you need, contact your local office or National Instruments corporate. Phone numbers for our worldwide offices are listed at the front of this manual.

Glossary

Prefix	Meaning	Value
p-	pico-	10^{-12}
n-	nano-	10^{-9}
μ-	micro-	10^{-6}
m-	milli-	10^{-3}
k-	kilo-	10^{3}
M-	mega-	10^{6}
G-	giga-	10^{9}
t-	tera-	10^{12}

Numbers/Symbols

°	degree
–	negative of, or minus
Ω	ohm
/	per
%	percent
+	positive of, or plus
+5 V	+5 VDC source signal

A

A	amperes
AC	alternating current
AC coupled	allowing the transmission of AC signals while blocking DC signals

ACH	analog input channel signal
A/D	analog-to-digital
ADC	analog-to-digital converter—an electronic device, often an integrated circuit, that converts an analog voltage to a digital number
ADC resolution	the size of the discrete steps in the ADC's input-to-output transfer function; therefore, the smallest voltage difference an ADC can discriminate with a single measurement
AIGND	analog input ground signal
alias	a false lower frequency component that appears in sampled data acquired at too low a sampling rate
amplification	a type of signal conditioning that improves accuracy in the resulting digitized signal and reduces noise
amplitude flatness	a measure of how close to constant the gain of a circuit remains over a range of frequencies
antialiasing filter	a lowpass filter preceding an ADC (usually a brickwall filter) that rejects signal energy above the Nyquist frequency (1/2 the sample rate) of the ADC so that the ADC does not mistake out-of-band signals for in-band signals.
anti-imaging filter	a lowpass filter after a DAC (usually a brickwall filter) that rejects signal energy above the Nyquist frequency (1/2 the sample rate) of the DAC in order to suppress out-of-band images of the in-band signal created by the D/A conversion process.
AOGND	analog output ground signal
ASIC	Application-Specific Integrated Circuit—a proprietary semiconductor component designed and manufactured to perform a set of specific functions for a specific customer
asynchronous	(1) hardware—a property of an event that occurs at an arbitrary time, without synchronization to a reference clock (2) software—a property of a function that begins an operation and returns prior to the completion or termination of the operation

attenuate	to decrease the amplitude of a signal
attenuation ratio	the factor by which a signal's amplitude is decreased

B

b	bit—one binary digit, either 0 or 1
B	byte—eight related bits of data, an eight-bit binary number. Also used to denote the amount of memory required to store one byte of data
bandwidth	the range of frequencies present in a signal, or the range of frequencies to which a measuring device can respond
base address	a memory address that serves as the starting address for programmable registers. All other addresses are located by adding to the base address.
binary	a number system with a base of 2
bipolar	a signal range that includes both positive and negative values (for example, –5 V to +5 V)
BNC	a type of coaxial signal connector
brickwall filter	a lowpass filter having a very flat passband, a very sudden, sharp transition region, and high rejection in the stopband.
buffer	temporary storage for acquired or generated data (software)
burst-mode	a high-speed data transfer in which the address of the data is sent followed by back-to-back data words while a physical signal is asserted
bus	the group of conductors that interconnect individual circuitry in a computer. Typically, a bus is the expansion vehicle to which I/O or other devices are connected. Examples of PC buses are the ISA and PCI bus.
bus master	a type of a plug-in board or controller with the ability to read and write devices on the computer bus

C

C	Celsius
CalDAC	calibration DAC
channel	pin or wire lead to which you apply or from which you read the analog or digital signal. Analog signals can be single-ended or differential. For digital signals, you group channels to form ports. Ports usually consist of either four or eight digital channels.
circuit trigger	a condition for starting or stopping clocks
clip	clipping occurs when an input signal exceeds the input range of the amplifier
clock	hardware component that controls timing for reading from or writing to groups
CMOS	complementary metal-oxide semiconductor
CMRR	common-mode rejection ratio—a measure of an instrument's ability to reject interference from a common-mode signal, usually expressed in decibels
code width	the smallest detectable change in an input voltage of a DAQ device
common-mode range	the input range over which a circuit can handle a common-mode signal
common-mode signal	the mathematical average voltage, relative to the computer's ground, of the signals from a differential input
common-mode voltage	any voltage present at the instrumentation amplifier inputs with respect to amplifier ground
compensation range	the range of a parameter for which compensating adjustment can be made
conditional retrieval	a method of triggering in which you simulate an analog trigger using software. Also called software triggering.
conversion device	device that transforms a signal from one form to another. For example, analog-to-digital converters (ADCs) for analog input, digital-to-analog converters (DACs) for analog output, digital input or output ports, and counter/timers are conversion devices.

conversion time	the time required, in an analog input or output system, from the moment a channel is interrogated (such as with a read instruction) to the moment that accurate data is available
counter/timer	a circuit that counts external pulses or clock pulses (timing)
coupling	the manner in which a signal is connected from one location to another
crosstalk	an unwanted signal on one channel due to an input on a different channel
current drive capability	the amount of current a digital or analog output channel is capable of sourcing or sinking while still operating within voltage range specifications
current sinking	the ability of a DAQ board to dissipate current for analog or digital output signals
current sourcing	the ability of a DAQ board to supply current for analog or digital output signals

D

D/A	digital-to-analog
DAC	digital-to-analog converter—an electronic device, often an integrated circuit, that converts a digital number into a corresponding analog voltage or current
DAC*x*OUT	analog channel *x* output signal
daisy-chain	a method of propagating signals along a bus, in which the devices are prioritized on the basis of their position on the bus
DAQ	data acquisition—(1) collecting and measuring electrical signals from sensors, transducers, and test probes or fixtures and inputting them to a computer for processing; (2) collecting and measuring the same kinds of electrical signals with A/D and/or DIO boards plugged into a computer, and possibly generating control signals with D/A and/or DIO boards in the same computer
dB	decibel—the unit for expressing a logarithmic measure of the ratio of two signal levels: $dB=20\log_{10} (V_1/V_2)$, for signals in volts
DC	direct current

DC coupled	allowing the transmission of both AC and DC signals
DDS	direct digital synthesis
default setting	a default parameter value recorded in the driver. In many cases, the default input of a control is a certain value (often 0) that means *use the current default setting*. For example, the default input for a parameter may be *do not change current setting*, and the default setting may be *no AMUX-64T boards*. If you do change the value of such a parameter, the new value becomes the new setting. You can set default settings for some parameters in the configuration utility or manually using switches located on the device.
delta-sigma modulating ADC	a high-accuracy circuit that samples at a higher rate and lower resolution than is needed and (by means of feedback loops) pushes the quantization noise above the frequency range of interest. This out-of-band noise is typically removed by digital filters.
device	a plug-in data acquisition board, card, or pad that can contain multiple channels and conversion devices. Plug-in boards, PCMCIA cards, and devices such as the DAQPad-1200, which connects to your computer parallel port, are all examples of DAQ devices. SCXI modules are distinct from devices, with the exception of the SCXI-1200, which is a hybrid.
DGND	digital ground signal
DIFF	differential mode
differential input	an analog input consisting of two terminals, both of which are isolated from computer ground, whose difference is measured
differential measurement system	a way you can configure your device to read signals, in which you do not need to connect either input to a fixed reference, such as the earth or a building ground
digital port	*See* port.
digital trigger	a TTL level signal having two discrete levels—a high and a low level
DIO	digital input/output
DMA	direct memory access—a method by which data can be transferred to/from computer memory from/to a device or memory on the bus while the processor does something else. DMA is the fastest method of transferring data to/from computer memory.

DNL	differential nonlinearity—a measure in least significant bit of the worst-case deviation of code widths from their ideal value of 1 LSB
down counter	performing frequency division on an internal signal
drivers	software that controls a specific hardware device such as a DAQ board or a GPIB interface board
dynamic range	the ratio of the largest signal level a circuit can handle to the smallest signal level it can handle (usually taken to be the noise level), normally expressed in decibels

E

encoder	a device that converts linear or rotary displacement into digital or pulse signals. The most popular type of encoder is the optical encoder, which uses a rotating disk with alternating opaque areas, a light source, and a photodetector.
event	the condition or state of an analog or digital signal
expansion ROM	an onboard EEPROM that may contain device-specific initialization and system boot functionality
EXT_TRIG	external digital trigger
external trigger	a voltage pulse from an external source that triggers an event such as A/D conversion

F

false triggering	triggering that occurs at an unintended time
FIFO	first-in first-out memory buffer—the first data stored is the first data sent to the acceptor. FIFOs are often used on DAQ devices to temporarily store incoming or outgoing data until that data can be retrieved or output. For example, an analog input FIFO stores the results of A/D conversions until the data can be retrieved into system memory, a process that requires the servicing of interrupts and often the programming of the DMA controller. This process can take several milliseconds in some cases. During this time, data accumulates in the FIFO for future retrieval. With a larger FIFO, longer latencies can be tolerated. In the case of analog output, a FIFO permits faster update rates, because the waveform data can be stored on the FIFO ahead of time. This again reduces the effect of latencies associated with getting the data from system memory to the DAQ device.
filtering	a type of signal conditioning that allows you to attenuate unwanted portions of the signal you are trying to measure
FIR	finite impulse response—a non recursive digital filter with linear phase
flash ADC	an ADC whose output code is determined in a single step by a bank of comparators and encoding logic
floating signal sources	signal sources with voltage signals that are not connected to an absolute reference or system ground. Also called nonreferenced signal sources. Some common example of floating signal sources are batteries, transformers, or thermocouples.
f_s	sample rate
ft	feet

G

gain	the factor by which a signal is amplified, sometimes expressed in decibels
gain accuracy	a measure of deviation of the gain of an amplifier from the ideal gain

GND	ground
grounded measurement system	*See* SE.

H

h	hour
half-power bandwidth	the frequency range over which a circuit maintains a level of at least –3 dB with respect to the nominal level
handshaked digital I/O	a type of digital acquisition/generation where a device or module accepts or transfers data after a digital pulse has been received. Also called latched digital I/O.
hardware	the physical components of a computer system, such as the circuit boards, plug-in boards, chassis, enclosures, peripherals, and cables
hardware triggering	a form of triggering where you set the start time of an acquisition and gather data at a known position in time relative to a trigger signal
hysteresis	the lag between making a change and the effect of the change
Hz	hertz—cycles per second. Specifically refers to the repetition frequency of a waveform.

I

IC	integrated circuit
IMD	intermodulation distortion—the ratio, in decibels, of the total rms signal level of harmonic sum and difference distortion products, to the overall rms signal level. The test signal is two sine waves added together according to the following standards: SMPTE—A 60 Hz sine wave and a 7 kHz sine wave added in a 4:1 amplitude ratio. DIN—A 250 Hz sine wave and an 8 kHz sine wave added in a 4:1 amplitude ratio. CCIF—A 14 kHz sine wave and a 15 kHz sine wave added in a 1:1 amplitude ratio.
in.	inches

INL	integral nonlinearity—a measure in LSB of the worst-case deviation from the ideal A/D or D/A transfer characteristic of the analog I/O circuitry
input bias current	the current that flows into the inputs of a circuit
input impedance	the measured resistance and reactance between the input terminals of a circuit
input offset current	the difference in the input bias currents of the two inputs of an instrumentation amplifier
instrument driver	a set of high-level software functions that controls a specific GPIB, VXI, or RS-232 programmable instrument or a specific plug-in DAQ board. Instrument drivers are available in several forms, ranging from a function callable language to a virtual instrument (VI) in LabVIEW.
instrumentation amplifier	a circuit whose output voltage with respect to ground is proportional to the difference between the voltages at its two inputs
integrating ADC	an ADC whose output code represents the average value of the input voltage over a given time interval
interrupt	a computer signal indicating that the CPU should suspend its current task to service a designated activity
interrupt level	the relative priority at which a device can interrupt
I/O	input/output—the transfer of data to/from a computer system involving communications channels, operator interface devices, and/or data acquisition and control interfaces
I_{OH}	current, output high
I_{OL}	current, output low
IRQ	interrupt request
isolation	a type of signal conditioning in which you isolate the transducer signals from the computer for safety purposes. This protects you and your computer from large voltage spikes and makes sure the measurements from the DAQ device are not affected by differences in ground potentials.
isolation voltage	the voltage that an isolated circuit can normally withstand, usually specified from input to input and/or from any input to the amplifier output, or to the computer bus

K

k	kilo—the standard metric prefix for 1,000, or 10^3, used with units of measure such as volts, hertz, and meters
K	kilo—the prefix for 1,024, or 2^{10}, used with B in quantifying data or computer memory
kbytes/s	a unit for data transfer that means 1,024 bytes/s
kS	1,000 samples
Kword	1,024 words of memory

L

LabVIEW	laboratory virtual instrument engineering workbench
latched digital I/O	a type of digital acquisition/generation where a device or module accepts or transfers data after a digital pulse has been received. Also called handshaked digital I/O.
library	a file containing compiled object modules, each comprised of one of more functions, that can be linked to other object modules that make use of these functions. NIDAQMSC.LIB is a library that contains NI-DAQ functions. The NI-DAQ function set is broken down into object modules so that only the object modules that are relevant to your application are linked in, while those object modules that are not relevant are not linked.
linearity	the adherence of device response to the equation R = KS, where R = response, S = stimulus, and K = a constant
linearization	a type of signal conditioning in which software linearizes the voltage levels from transducers, so the voltages can be scaled to measure physical phenomena
low frequency corner	in an AC-coupled circuit, the frequency below which signals are attenuated by at least 3 dB
LSB	least significant bit

M

m	meters
M	(1) Mega, the standard metric prefix for 1 million or 10^6, when used with units of measure such as volts and hertz; (2) mega, the prefix for 1,048,576, or 2^{20}, when used with B to quantify data or computer memory
Mbytes/s	a unit for data transfer that means 1,048,576 bytes/s
memory buffer	*See* buffer.
MITE	MXI Interface to Everything—a custom ASIC designed by National Instruments that implements the PCI bus interface. The MITE supports bus mastering for high-speed data transfers over the PCI bus.
MS	million samples
MSB	most significant bit
MTBF	mean time between failure
MTTR	mean time to repair—predicts downtime and how long it takes to fix a product

N

NC	normally closed, or not connected
NI-DAQ	National Instruments driver software for DAQ hardware
NIST	National Institute of Standards and Technology
noise	an undesirable electrical signal—Noise comes from external sources such as the AC power line, motors, generators, transformers, fluorescent lights, soldering irons, CRT displays, computers, electrical storms, welders, radio transmitters, and internal sources such as semiconductors, resistors, and capacitors. Noise corrupts signals you are trying to send or receive.
nonlatched digital I/O	a type of digital acquisition/generation where LabVIEW updates the digital lines or port states immediately or returns the digital value of an input line. Also called immediate digital I/O or non-handshaking.

nonreferenced signal sources	signal sources with voltage signals that are not connected to an absolute reference or system ground. Also called floating signal sources. Some common example of nonreferenced signal sources are batteries, transformers, or thermocouples.
NRSE	nonreferenced single-ended mode—all measurements are made with respect to a common (NRSE) measurement system reference, but the voltage at this reference can vary with respect to the measurement system ground
Nyquist Frequency	one-half of the sampling rate, f_s

O

onboard channels	channels provided by the plug-in data acquisition board
operating system	base-level software that controls a computer, runs programs, interacts with users, and communicates with installed hardware or peripheral devices
optical isolation	the technique of using an optoelectric transmitter and receiver to transfer data without electrical continuity, to eliminate high-potential differences and transients
output settling time	the amount of time required for the analog output voltage to reach its final value within specified limits
output slew rate	the maximum rate of change of analog output voltage from one level to another

P

passband	the range of frequencies which a device can properly propagate or measure
pattern generation	a type of handshaked (latched) digital I/O in which internal counters generate the handshaked signal, which in turn initiates a digital transfer. Because counters output digital pulses at a constant rate, this means you can generate and retrieve patterns at a constant rate because the handshaked signal is produced at a constant rate.

PCI	Peripheral Component Interconnect—a high-performance expansion bus architecture originally developed by Intel to replace ISA and EISA. It is achieving widespread acceptance as a standard for PCs and work-stations; it offers a theoretical maximum transfer rate of 132 Mbytes/s.
peak to peak	a measure of signal amplitude; the difference between the highest and lowest excursions of the signal
PFI	programmable function input
phase suppression	a feature in FFT mode that lets you set a threshold to effectively filter out noise from signal phase information
Plug and Play devices	devices that do not require DIP switches or jumpers to configure resources on the devices—also called switchless devices
port	(1) a communications connection on a computer or a remote controller (2) a digital port, consisting of four or eight lines of digital input and/or output
posttriggering	the technique used on a DAQ board to acquire a programmed number of samples after trigger conditions are met
potentiometer	an electrical device the resistance of which can be manually adjusted; used for manual adjustment of electrical circuits and as a transducer for linear or rotary position
ppm	parts per million
pretriggering	the technique used on a DAQ board to keep a continuous buffer filled with data, so that when the trigger conditions are met, the sample includes the data leading up to the trigger condition
propagation	the transmission of a signal through a computer system
propagation delay	the amount of time required for a signal to pass through a circuit
pts	points
pulse trains	multiple pulses
pulsed output	a form of counter signal generation by which a pulse is outputted when a counter reaches a certain value

Q

quantization error	the inherent uncertainty in digitizing an analog value due to the finite resolution of the conversion process
quantizer	a device that maps a variable from a continuous distribution to a discrete distribution

R

real time	a property of an event or system in which data is processed as it is acquired instead of being accumulated and processed at a later time
relative accuracy	a measure in LSB of the linearity of an ADC. It includes all non-linearity and quantization errors. It does not include offset and gain errors of the circuitry feeding the ADC.
resolution	the smallest signal increment that can be detected by a measurement system. Resolution can be expressed in bits, in proportions, or in percent of full scale. For example, a system has 12-bit resolution, one part in 4,096 resolution, and 0.0244% of full scale.
resource locking	a technique whereby a device is signaled not to use its local memory while the memory is in use from the bus
retry	an acknowledge by a destination that signifies that the cycle did not complete and should be repeated
ribbon cable	a flat cable in which the conductors are side by side
rise time	the difference in time between the 10% and 90% points of a system's step response
rms	root mean square—the square root of the average value of the square of the instantaneous signal amplitude; a measure of signal amplitude

S

s	seconds
S	samples

sensor	a device that responds to a physical stimulus (heat, light, sound, pressure, motion, flow, and so on), and produces a corresponding electrical signal
settling time	the amount of time required for a voltage to reach its final value within specified limits
signal conditioning	the manipulation of signals to prepare them for digitizing
SNR	signal-to-noise ratio—the ratio of the overall rms signal level to the rms noise level, expressed in decibels
software trigger	a programmed event that triggers an event such as data acquisition
software triggering	a method of triggering in which you simulate an analog trigger using software. Also called conditional retrieval.
source impedance	a parameter of signal sources that reflects current-driving ability of voltage sources (lower is better) and the voltage-driving ability of current sources (higher is better)
SS	simultaneous sampling—a property of a system in which each input or output channel is digitized or updated at the same instant
S/s	samples per second—used to express the rate at which a DAQ board samples an analog signal
synchronous	(1) hardware—a property of an event that is synchronized to a reference clock (2) software—a property of a function that begins an operation and returns only when the operation is complete
system noise	a measure of the amount of noise seen by an analog circuit or an ADC when the analog inputs are grounded
system RAM	RAM installed on a personal computer and used by the operating system, as contrasted with onboard RAM

T

TC	terminal count—the highest value of a counter
T/H	track-and-hold—a circuit that tracks an analog voltage and holds the value on command

THD	total harmonic distortion—the ratio of the total rms signal due to harmonic distortion to the overall rms signal, in decibel or a percentage
THD+N	signal-to-THD plus noise—the ratio in decibels of the overall rms signal to the rms signal of harmonic distortion plus noise introduced
throughput rate	the data, measured in bytes/s, for a given continuous operation, calculated to include software overhead.
transducer	*See* sensor.
transducer excitation	a type of signal conditioning that uses external voltages and currents to excite the circuitry of a signal conditioning system into measuring physical phenomena
transfer rate	the rate, measured in bytes/s, at which data is moved from source to destination after software initialization and set up operations; the maximum rate at which the hardware can operate
trigger	any event that causes or starts some form of data capture
TTL	transistor-transistor logic

U

unipolar	a signal range that is always positive (for example, 0 to +10 V)
update	the output equivalent of a scan. One or more analog or digital output samples. Typically, the number of output samples in an update is equal to the number of channels in the output group. For example, one pulse from the update clock produces one update which sends one new sample to every analog output channel in the group.
update rate	the number of output updates per second

V

V	volts
VI	virtual instrument—(1) a combination of hardware and/or software elements, typically used with a PC, that has the functionality of a classic stand-alone instrument (2) a LabVIEW software module (VI), which consists of a front panel user interface and a block diagram program

W

waveform	multiple voltage readings taken at a specific sampling rate
word	the standard number of bits that a processor or memory manipulates at one time. Microprocessors typically use 8-, 16-, or 32-bit words.
working voltage	the highest voltage that should be applied to a product in normal use, normally well under the breakdown voltage for safety margin.

Z

zero-overhead looping	the ability of a high-performance processor to repeat instructions without requiring time to branch to the beginning of the instructions
zero-wait-state memory	memory fast enough that the processor does not have to wait during any reads and writes to the memory