WHITE PAPER

Engineer's Guide to the Digitization of Analog Signals

Your go-to technical resource for understanding and improving the quality of your analog measurements.

CONTENTS

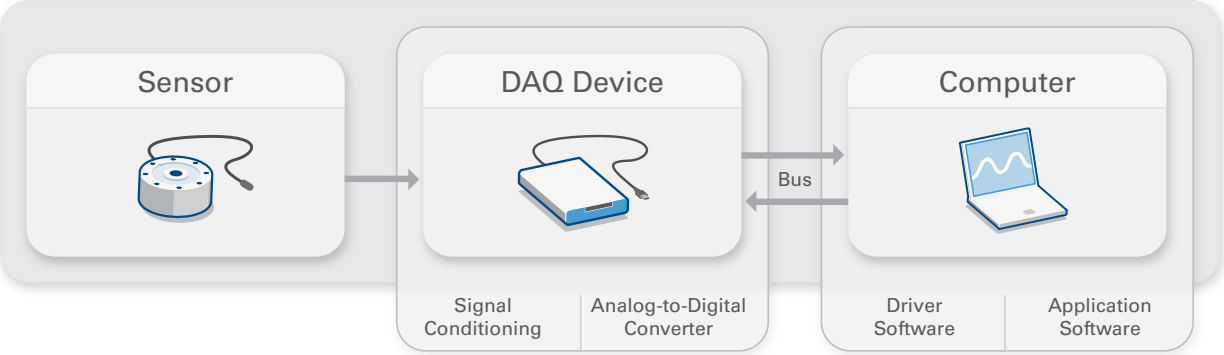

Figure 1. These are the primary components of any data acquisition system.

What Is Digitization?

Digitization is the process of converting an electrical signal, often from a sensor, into a digital signal that a computer can read. It is the core function of a data acquisition (DAQ) system. A DAQ system consists of sensors, DAQ measurement hardware, and a computer with programmable software. Compared with traditional measurement systems, PC-based DAQ systems exploit the processing power, productivity, display, and connectivity capabilities of industry-standard computers to provide a more powerful, flexible, and cost-effective measurement solution.

Sensor

The measurement of a physical phenomenon, such as the temperature of a room, the intensity of a light source, or the force applied to an object, begins with a sensor. A sensor, also called a transducer, converts a physical phenomenon into a measurable electrical signal. Depending on the type of sensor, its electrical output can be a voltage, a current, resistance, or another electrical attribute that varies over time. Some sensors may require additional components and circuitry to properly produce a signal that can accurately and safely be read by a DAQ device.

For more information on sensor fundamentals, download the Engineer's Guide to Accurate Sensor Measurements.

DAQ Device

DAQ hardware acts as the interface between a computer and signals from the outside world. It primarily functions as a device that translates incoming analog signals so that a computer can interpret them. The three key components of a DAQ device used for measuring a signal are:

1. The signal conditioning circuitry
2. The analog-to-digital converter (ADC)—**this guide focuses on this topic**
3. The computer bus

Many DAQ devices include other functions for automating measurement systems and processes. For example, digital-to-analog converters (DACs) output analog signals, digital I/O lines input and output digital signals, and counter/timers measure and generate digital pulses.

For more information regarding signal conditioning, download the Engineer's Guide to Signal Conditioning.

Computer

A computer with configurable software controls the operation of the DAQ device and is used for processing, visualizing, and storing measurement data. Different types of computers are used in different types of applications. A desktop may be used in a lab for its processing power, a laptop may be used in the field for its portability, or an industrial computer may be used in a manufacturing plant for its ruggedness.

To understand processing and address considerations for all parts of a DAQ system, download the Complete Guide to Building a Measurement System.

Selecting Digitization Hardware

This section examines the best technologies for a particular measurement. The following sections explore in greater technical detail the foundation of these technologies that you can combine to create digitization devices. Hyperlinked text takes you to related sections for more information.

At the highest level, the goal when using digitization hardware is to place a measurement of a given signal in both time and amplitude. The degree to which you can achieve this depends on a multitude of factors, and every measurement scenario is somewhat unique in its requirements.

Though you have many factors to think about, this document focuses on a few of the primary considerations.

Understanding the Nature of the Signal You Are Measuring

When selecting a digitization device, you need to understand the analog signal you are digitizing. Consider the following questions about this signal:

Is the signal continuous or do you expect discontinuities (for example, nonrepeating level changes, large frequency variations, spikes, and so on)?

Continuous signals at moderate frequencies (audio spectrum and below) where noise may be a concern need a delta-sigma-based DAQ device that implements oversampling with anti-aliasing filters.	Noncontinuous or rapidly changing signals may require the higher instantaneous bandwidth and faster sample rate of a SAR-based or pipeline-based DAQ device for lower latency.

Does the amplitude change rapidly with respect to time, or is it slow changing or even DC?

Fast-changing (high-frequency) signals must be sampled at a sufficient rate to avoid aliasing per the Nyquist Sampling Theorem.	Slow-changing (low-frequency) signals may need to be sampled slowly to avoid an unmanageably large data set.

Is the signal or the environment noisy?

Noisy signals can benefit from onboard filtering and averaging to mitigate—or remove entirely—the impact on the measured signal.	A better understanding of the sources of noise can also help you select a device to minimize the impact.

Are the signals of interest triggered by another event?

Understanding the type of event, if any, that may trigger the digitization—a digital pulse, an analog level, and so on—is key.	Latency—the time between the trigger event and the first digitized sample—should be understood and managed accordingly.

What range of voltages are expected?

Selecting a device with appropriate input ranges helps maximize the resolution of the signal.	Some devices offer a larger dynamic range for capturing signals that may vary significantly over the course of a measurement period.

Understanding the Nature of the Desired Insights

Analog signals are digitized to provide some understanding of a real-world phenomenon. Consider the following questions to ensure a digitization device can provide the correct data for gathering the required engineering insights:

Is it more important to characterize the state of the signal in the time domain or to understand the spectral content of the signal in the frequency domain?

A higher sample rate and higher bandwidth can ensure time-domain data and possibly capture noise in its entirety.	Accurately capturing spectral content in the frequency domain requires appropriate filtering and alias rejection.

What volume of measurements generates a meaningful data set?

Multichannel applications can quickly become costly if the tight synchronization and low latency of simultaneous measurements are critical.	Multiplexed architectures, on the other hand, help to increase channel density without increasing cost, but their trade-off is interchannel delays and possibly ghosting at faster sample rates.

Determining the Required Level of Accuracy

A perfect one-to-one translation of a signal from analog to digital is impossible in practical terms. Understanding the impact of digitizer accuracy on measurement results ensures that you use the appropriate hardware to achieve your desired level of confidence:

How close to actual does the measured amplitude need to be to provide meaningful insights?

Resolution and input range dictate the sensitivity and code width (smallest detectable change) of a given measurement device. Absolute accuracy (the degree to which the measured signal matches the actual signal) accounts for many other factors.

For periodic signals, what level of accuracy is required for time-domain frequency measurements?

The difference between the measured frequency of the signal and the frequency of the actual signal is a function of the DAQ device. Understanding the capability of a DAQ device's timebase helps characterize frequency measurement performance.

Are the signals of interest DC or AC?

AC measurement accuracy does not directly correlate to DC accuracy specifications, and appropriate DAQ devices are specified accordingly.

Selecting a Digitization Device Based on Requirements: Case Studies

Though possibly challenging, selecting a device based on your application requirements instead of the device's capabilities can ultimately lead to a more appropriate measurement. Since every application is different, consider the following two case studies to help you mentally prepare to select the appropriate digitization device:

Case 1: Tracking Temperature Change During the Ramp-and-Soak Process

This application involves a large tank of fluid that needs to be heated and cooled in phases, with some stable temperature time in between. In this case, the signal of interest is the temperature of the tank in many different locations to monitor and control the process. The environment is a factory floor, so machine and power line noise is a concern. The temperature changes are relatively slow, and accuracy is critical to only plus or minus a degree or so.

The first option to consider is a device based on successive-approximation register (SAR), which would keep costs low and measurement accuracy sufficient. You could also oversample and average the signals to help smooth some of the noise. A SAR device is not likely to have the necessary conditioning built in for thermocouple measurements, but with enough samples per data point and a selectable input range, the resolution should be sufficient to make up for shortcomings.

The second, and more ideal, option is a DSA-based device with built-in signal conditioning. The resolution is better because of the inherent oversampling in DSA devices. And anti-alias filtering is already accounted for in the hardware, making post-processing less necessary. This is especially pertinent because you use these temperature measurements to control the process. By spending slightly more per channel, you can ensure the integrity of the signal is better than that of the SAR-based device without additional engineering effort on the software or the signal conditioning sides of the problem.

Case 2: Measuring Transients in Battery Discharge Test

In this case, you are monitoring a multicell battery during charge and discharge to characterize performance. One key measurement is the number of transient voltage spikes during the process because these can be difficult to observe and potentially hazardous to the rest of the system's components. The transients occur very quickly, probably on the order of microseconds. Faster spikes are possible but unlikely. This battery comprises about 30 individual cells, all of which you ideally need to monitor independently.

One option for this case is a SAR-based device. You likely need the built-in or external isolation this device provides to accommodate higher common-mode voltages, but the true driving factor is sample rate capability. Since the transients are spikes that last only microseconds, you must sample the voltage lines quickly to avoid missing a spike. A SAR-based device provides good channel density with fast enough sample rates for most of the transient signals expected in this system. Though you could multiplex, a simultaneous device is likely the better choice to keep interchannel phase alignment as tight as possible while minimizing the opportunity for ghosting and keeping the price per channel reasonable.

Since transients can occur even faster than expected, another option is a pipeline-based device that provides exceptionally high sample rates along with isolated inputs. The trade-off here is that pipeline devices cost more and typically offer only one or two channels per device, so you need lots of them to measure all the channels. Though these devices guarantee more complete test coverage, they may be a bit overkill for the application and too expensive for your solution.

Hardware Architectures for Digitization

Though you can choose from many architectures to digitize analog signals to digital codes, a few are by far the most common: SAR, delta sigma (ΔΣ), and pipeline. Each has specific advantages and disadvantages, but all perform fundamentally the same task: characterizing an analog voltage with a digital representation.

The ADC architecture you select impacts your ability to trigger a measurement based on an event. Each ADC's latency is a measure of the delay between a trigger event and the next valid measurement.

Successive-Approximation Register (SAR)

Cost	Sample Rate	Resolution	Noise	Latency
Moderate	Moderate (<10 MS/s)	Moderate	Moderate	Low

Table 1. SAR ADCs are known for being exceptional at low-latency digitization.

SAR ADCs feature one of the more common architectures used for data acquisition. Because of their low cost and relatively simple implementation, you can incorporate them in cost-effective solutions that deliver exceptional performance. Most notably, SAR-based DAQ devices can take measurements with minimal latency, meaning they can precisely place measurements in time.

You digitize analog signals with SAR-based devices by using a method like a binary search: you capture and compare the input voltage with successively smaller references until a precise measurement results. Visually, this method looks like Figure 2, which shows a 10-bit (steps along the x-axis) SAR ADC measuring an input voltage of 3.5 V with an internal reference voltage of 5 V:

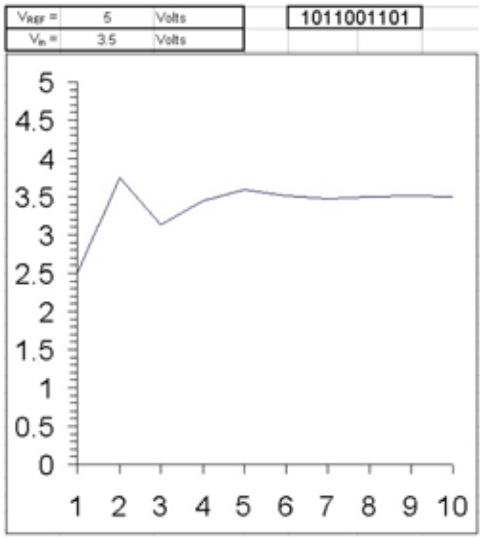

Figure 2. Successive Approximation Conversion by Russ Puskarcik—Own Work, CC BY 3.0, *https://commons.wikimedia.org/w/index.php?curid=39713802*

Additionally, the SAR architecture lends itself well to the inclusion of multiple inputs in the same ADC circuitry at the expense of sample rate per channel. This process, called multiplexing, is discussed in greater detail later in this document. When you need to tightly synchronize adjacent channels, you can use multiple SAR ADCs on a single DAQ device to make simultaneous measurements

Delta Sigma (ΔΣ)

Cost	Sample Rate	Resolution	Noise	Latency
Moderate	Lower (<1 MS/s)	High	Low	High

Table 2. Delta-sigma ADCs are known for their exceptional dynamic signal performance and resolution.

Delta-sigma-based DAQ devices have the primary advantages of noise and alias rejection with exceptional resolution, but they provide a lower effective sample rate, lower bandwidth, and increased latency, or pinpointing measurements with respect to absolute time.

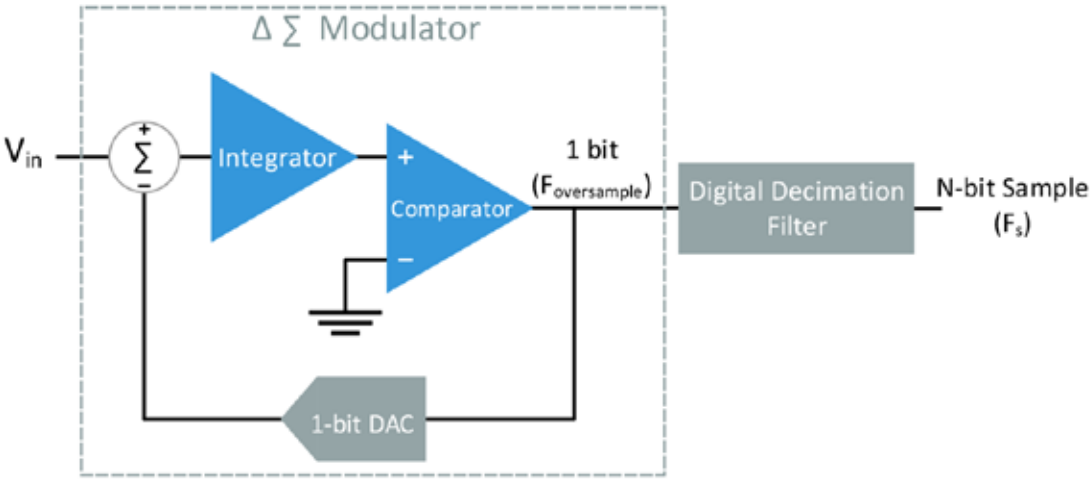

Figure 3. Delta-sigma modulation uses a 1-bit DAC, an integrator, and a comparator to characterize dynamic signals by oversampling.

Conceptually, like pulse-width modulation, the ΔΣmodulator (Figure 3) turns the input voltage you intend to measure into a stream of pulses between zero and some reference voltage. This binary bit stream is clocked at a much higher rate (called oversampling) than your desired rate of the measurement, so you can filter and decimate blocks of the bit stream into a higher resolution representation of the original input signal.

Though the technical implementation of this digitization technique is beyond the scope of this document, the implications are relevant. The oversampling approach means the Nyquist frequency is well above the frequency of the input signal, which in turn means the aliases of that signal are even higher still.

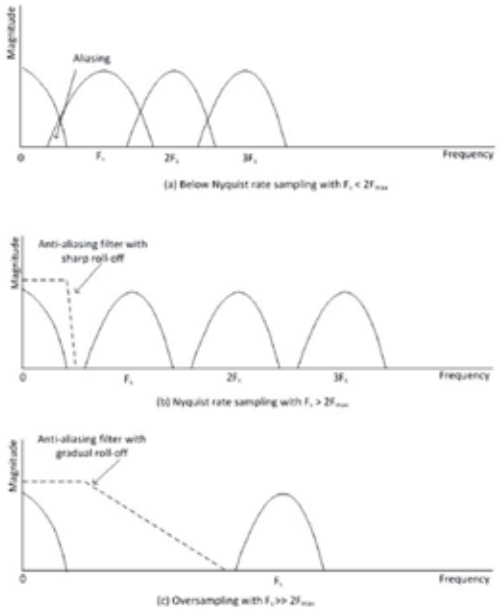

Figure 4. Anti-aliasing filters are simpler to implement with space between the signal of interest and the Nyquist frequency.

Since oversampling provides a generous amount of space between the signal of interest and the Nyquist frequency, filtering becomes a simpler task. By applying an anti-aliasing filter

with a cutoff before the Nyquist frequency, you can reject any higher frequency content in the measured signal, which reduces the amount of noise and distortion around the signal of interest.

Additionally, oversampling helps reduce noise and increase the dynamic range of a measurement by spreading the quantization noise across the wider frequency spectrum of the oversampled data. Inherent in the analog-to-digital conversion process, quantization is a factor in all forms of digitization. During sampling, quantization noise is spread uniformly across the Nyquist frequency band of interest, as shown in the first graph in Figure 5:

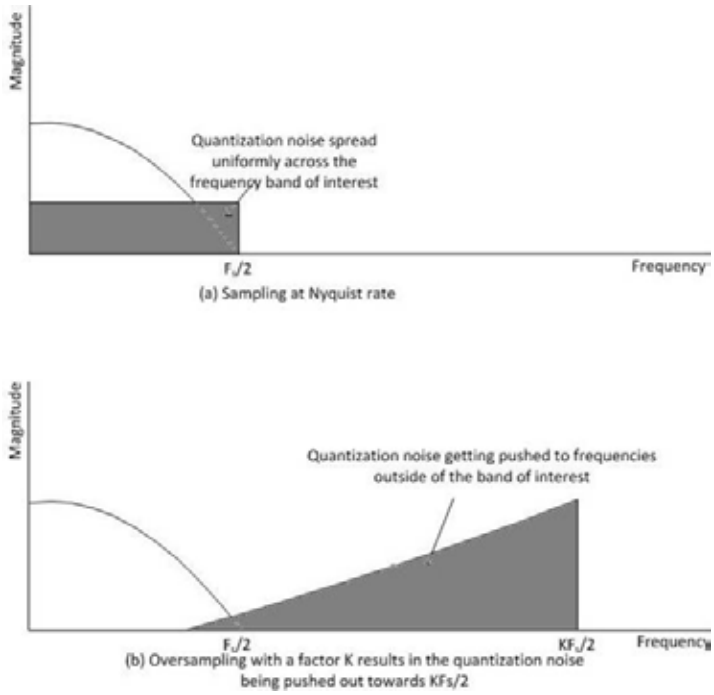

Figure 5. Oversampling pushes quantization noise beyond the frequency bands containing the signal of interest.

In the second graph of Figure 5, the digitization rate has been increased by an oversample factor (K), which pushes the quantization noise out to higher frequencies. Since the frequency band of interest is below the Nyquist frequency, the quantization noise is greatly reduced relative to the measured signal. Because of this, delta-sigma ADCs can provide superior signal-to-noise ratios with exceptional dynamic range.

Delta-sigma ADCs are commonly used in applications with continuous signals for which low noise and high dynamic range are requirements and often coupled with additional signal conditioning. Also, this architecture is common to digital multimeter (DMM) devices, which provide exceptional resolution at the expense of a low absolute acquisition rate.

Pipeline

Cost	Sample Rate	Resolution	Noise	Latency
High	High (1 GS/s)	Moderate	Moderate	Moderate

Table 3. Pipeline ADCs are known for producing exceptionally fast sample rates.

Pipeline-based ADCs use a collection of low-resolution ADCs in sequential stages to achieve exceptionally fast acquisition rates at a moderate resolution. Generally, faster sample rates correspond with lower resolution and vice versa.

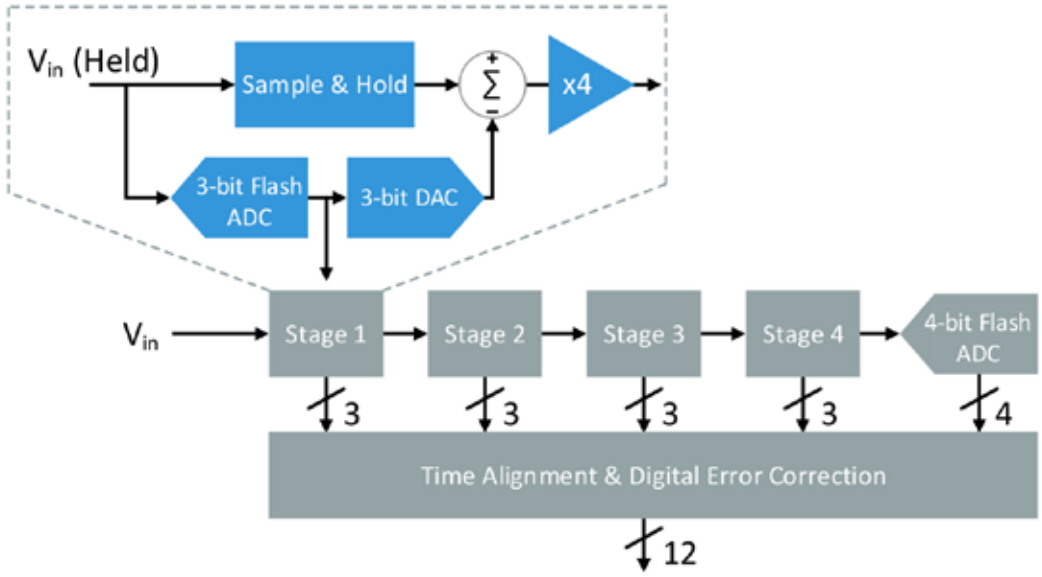

Figure 6. Pipelined ADCs use a series of stages to parallelize digitization and increase sample rate.

In Figure 6, the input voltage is digitized at each stage process with a 3-bit flash ADC. These three bits are recorded and then converted back to analog with a 3-bit DAC. The resulting signal is then subtracted from the original input voltage. This residual signal is amplified by a gain factor (in this case four) and passed along to the next stage. In the final stage, a 4-bit flash ADC determines the last four bits.

When each stage ends, additional circuitry combines the bits generated at each stage and time-aligns them to produce the final digital representation of the original input voltage. This may seem like a long process, but since stages can begin operation on the next input voltage as soon as their residual voltage is passed along—hence the "pipeline" name—fast sample rates can be attained.

Pipeline-based ADCs are most commonly used in high-speed digitizers like oscilloscopes and wireless base stations and in applications that require high-speed spectrum analysis.

Architecture Comparison

	Cost	Sample Rate	Resolution	Noise	Latency
SAR	Low to Moderate	Moderate (<10 MS/s)	Moderate	Moderate	Low
ΔΣ	Moderate	Lower (<1 MS/s)	High	Low	High
Pipelined	High	High (1 GS/s)	Moderate	Moderate	Moderate

Table 4. Carefully selecting an ADC type can make a significant difference in overall system performance and cost.

On the frequency versus resolution curve, these architectures largely occupy unique areas; however, you need to consider where they overlap in capability (Figure 7).

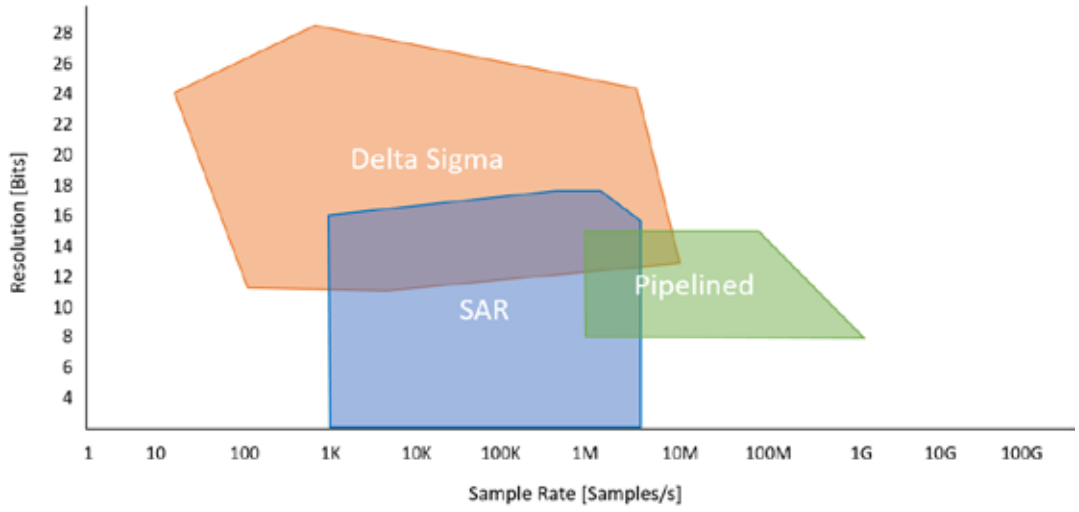

Figure 7. Each ADC type covers an area of the frequency versus resolution spectrum.

Digitization Timing

Modern DAQ hardware provides a variety of digitization architectures, all of which require some type of electrical connection to a physical pin or terminal to receive the signal you are measuring. The ideal option varies depending on your DAQ application needs, including channel density, acquisition rate per channel, and phase alignment between adjacent channels on a device. Your choice of architecture to pair ADCs with input terminals affects sample conversion timing.

Digitization Timing Architectures

The digitization of analog signals is orchestrated by a timing infrastructure within a DAQ device. One or multiple time sources may specify when samples are converted from analog to digital and at what rate. The two most common architectures are simultaneous and multiplexed DAQ devices.

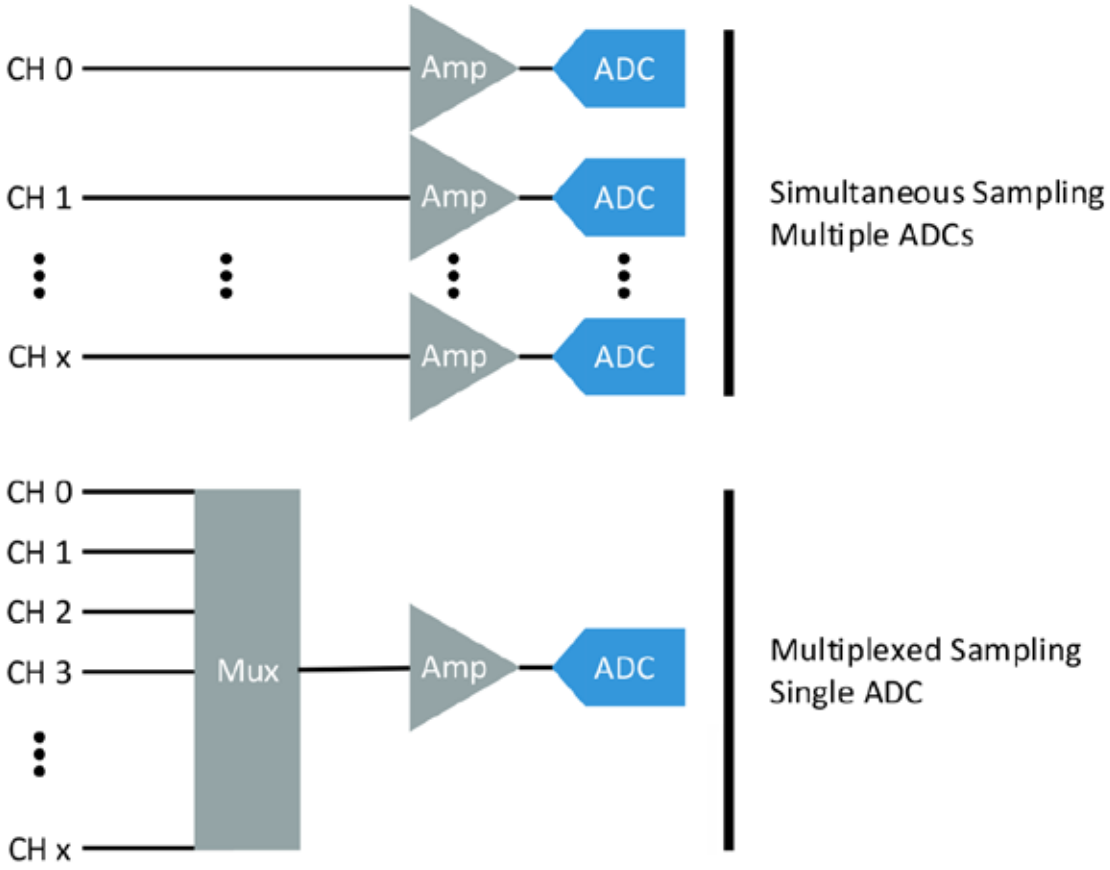

Figure 8. The number of ADCs is the primary differentiator between architectures.

Simultaneous architectures use one ADC per input channel. This has several implications:

1. You can achieve the full sample rate on all channels regardless of how many channels you are using
2. You can acquire samples in parallel at precisely the same moment in time
3. Space limitations curb the number of channels that can fit on a DAQ device
4. Device cost tends to be greater due to the extra ADC and analog front-end components

Multiplexed architectures use one ADC and analog front end for all channels and a multiplexer —or switch—to scan the input channels. This also has several implications:

1. The full sample rate is shared among all channels, meaning the maximum rate per channel decreases as more channels are added to the acquisition
2. Samples are acquired in series with a precise delay between channels
3. You can achieve greater channel densities on a single DAQ device
4. Cost per channel tends to be less

Both these architectures use a sample clock. The sample clock controls the rate at which samples are acquired and generated. Specifically, the sample clock sets the time interval between samples.

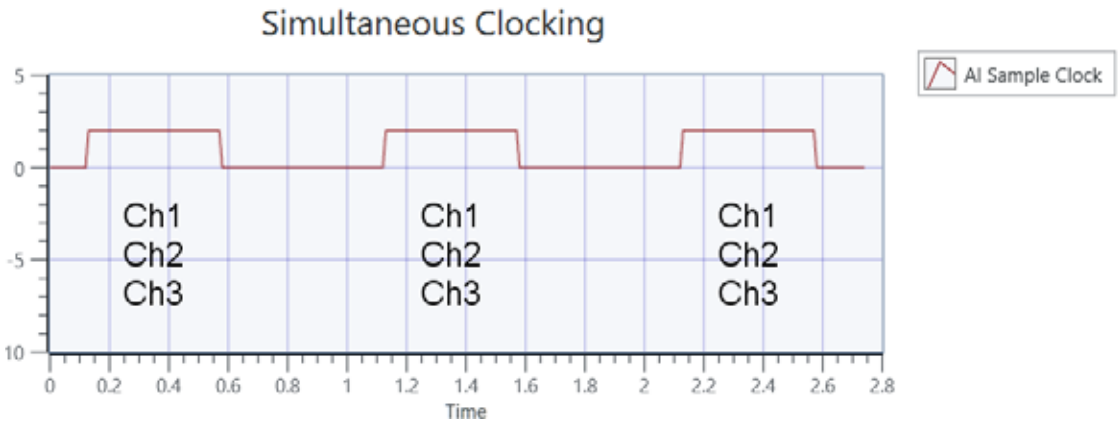

Figure 9. Each tick of the sample clock triggers acquisition on all active channels.

This acquisition is sampling three channels simultaneously. These channels should be able to guarantee phase alignment between acquisition channels to within a fraction of a degree of separation.

Multiplexed architectures also require a convert clock. The convert clock directly causes ADC conversions. The relationship between these signals is illustrated in Figure 10.

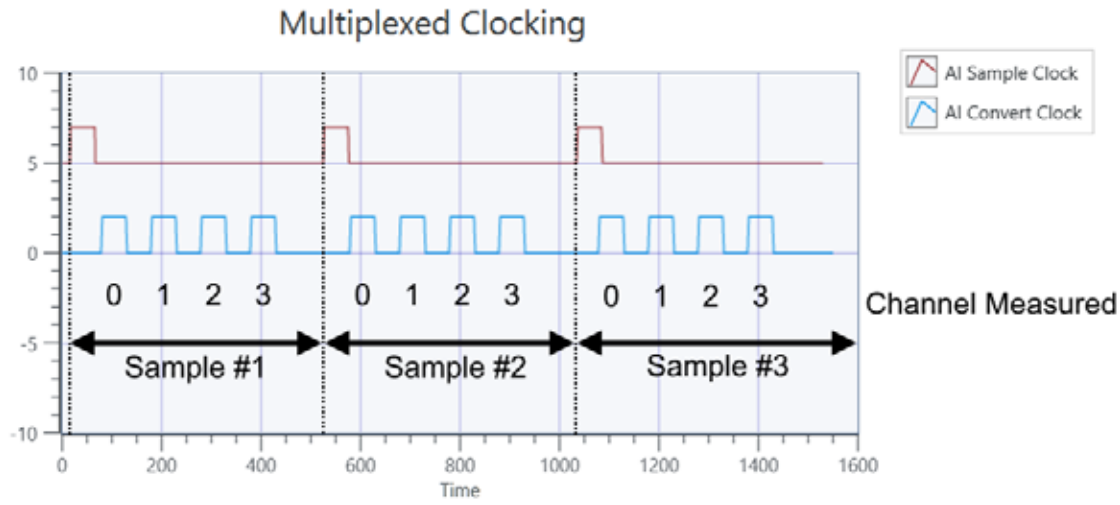

Figure 10. Each tick of the sample clock initiates a series of acquisitions on all active channels, as specified by a convert clock.

The Figure 10 multiplexed acquisition is sampling four input channels at a frequency dictated by the AI Sample Clock. Note that the AI Convert Clock must run at a rate at least four times greater than the sample clock to ensure that all four channels are digitized before the next sample clock tick occurs.

Correct Phase Alignment in Software

Certain applications require the precision timing and increased sample rates of simultaneously sampled hardware. These applications include sound and vibration, ultrasonic, ballistic, transient analysis, high-speed control, and other applications with fast, synchronized dynamic signals. For applications with less strict requirements, you can use software to process multiplexed data and realign channels as long as you time the acquisition with hardware during acquisition and you know the clocking frequencies of the DAQ device.

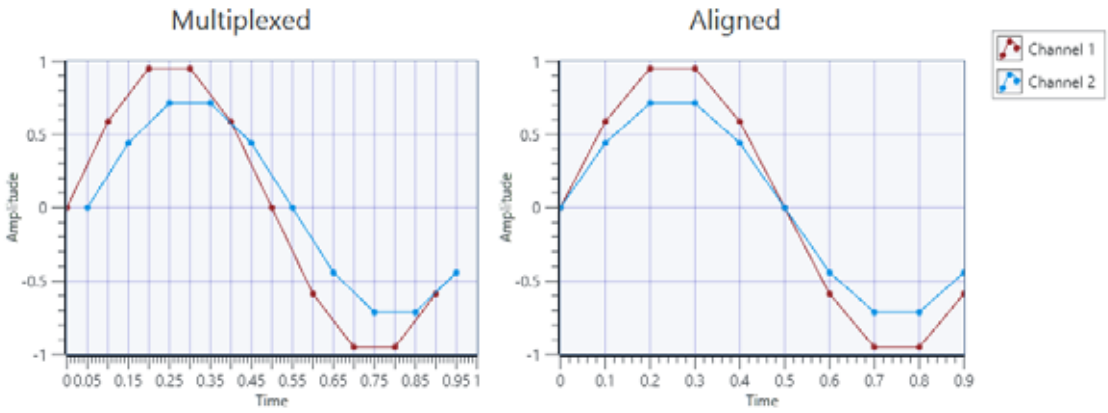

Figure 11. Multiplexing causes an inherent phase delay between sampled channels.

For example, the multiplexed acquisition in Figure 11 was a 2-channel, hardware-timed acquisition with a known convert clock frequency. By subtracting the time between convert clock ticks from the time component of the channel 2 waveform data, you can realign the samples to be effectively simultaneous.

This method is recommended only for low- to medium-sample-rate applications that measure generally repetitive signals. Signals with known aliased data or transient data should not use realignment algorithms.

Ghosting

As the sample rate increases with a multiplexed device and time between sample clock ticks becomes shorter and shorter, you must consider the settling time of each input channel. Settling time is the time required for a signal to reach a specified accuracy range, shown as t_s in Figure 12:

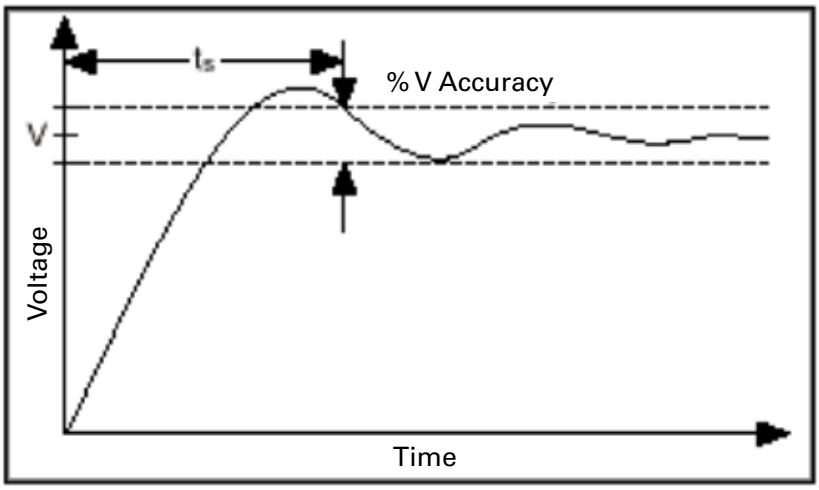

Figure 12. Settling time is the time it takes for a switched voltage to settle to within acceptable limits.

Ghosting is a phenomenon that occurs when the time between convert clock ticks is less than the settling time of a DAQ device. In this case, signals on adjacent channels may begin to interfere with one another, which leads to inaccurate measurements.

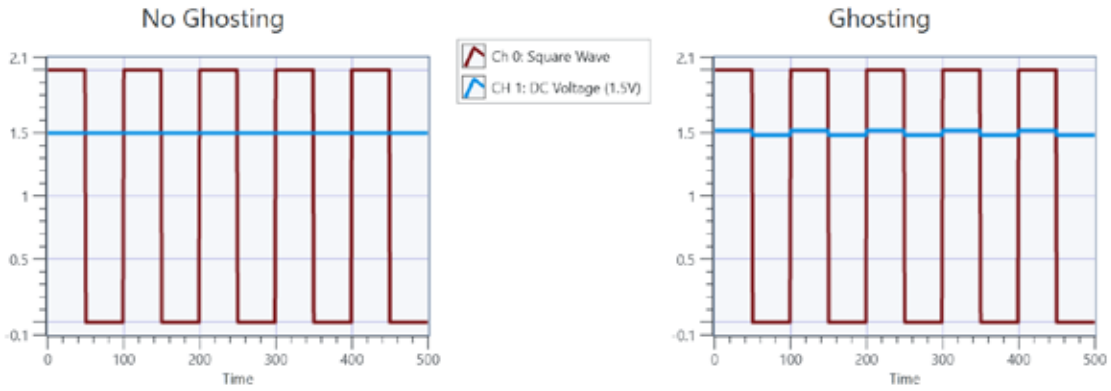

Figure 13. Ghosting can occur as high sample rates on multiplexed devices.

In the graph on the left in Figure 13, the sample rate is low enough to allow for proper settling between measurements on channels 0 and 1. In the graph on the right, the sample rate has been increased to a point where adequate sampling time is not available, and you can observe the impact of the square wave on the DC voltage measurement.

You can avoid ghosting by taking the following precautions:

1. Sample at a rate low enough to avoid violating the prescribed settling time for a device, or reduce the number of channels acquired per device.
2. Be conscious of which signals are positioned next to each other in the channel scan list. Avoid large jumps in voltage on adjacent channels to reduce the settling time required.

3. Measure "dummy" channels between "real" channels. This is especially useful when changing input ranges from one channel to the next. As an example, you may measure channel 0 at ±5 V, then short channel 1 to ground and measure at ±2 V, and finally measure channel 2 at ±2 V. You can discard the channel 1 measurement, but give the amplification circuitry enough time to settle before the "real" measurement on channel 2.
4. Provide a lower source impedance. If the impedance of the signal source is greater than 1 kΩ, it can cause the acquisition circuitry to take longer to settle, which can lead to ghosting at higher sample rates. A unity gain buffer is one method to reduce source impedance.

Triggering an Acquisition

Though allowing a digitizer to run freely for an undefined amount of time is one option to ensure events of interest are captured, it is not the most efficient use of processing power or disk space. Data files could quickly become unmanageably large, especially for acquisitions at faster sample rates. Consider the following types of triggering when evaluating digitization hardware because their capabilities often vary:

Digital Edge Triggers

These triggers generally use transistor-transistor logic (TTL) levels and require a relatively clean digital edge at specific voltage levels (below 5 V) to trigger properly. TTL defines the thresholds above and below which a trigger event can be registered and the rise and fall times required in the transition from logic high to logic low. Digital triggers also require a separate digital input or trigger input line that monitors for digital edge triggers. Figure 14 shows these specifications for digital output and digital input TTL signals:

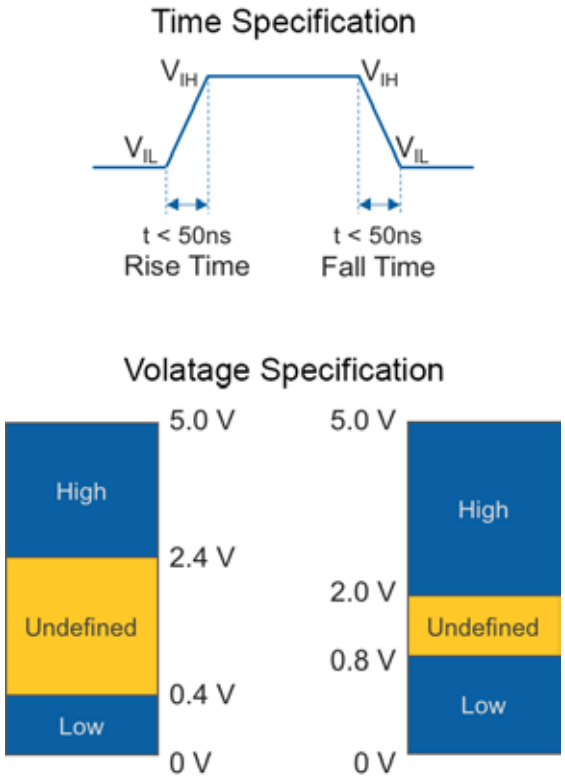

Figure 14. Digital triggering is specified by the TTL standard, which dictates both the time and level of the digital signals.

Analog Triggers

Analog triggering typically monitors the same analog input line that is digitized after a defined trigger event is registered on that line. It is more flexible than digital triggering, but it requires careful consideration to ensure the configured trigger captures the intended event instead of erroneous activity. To achieve this, DAQ devices may support one of the following analog trigger types.

Analog level triggers are the most straightforward, and they define a trigger event by some input voltage level. When and if the input signal passes that level, a trigger event is registered.

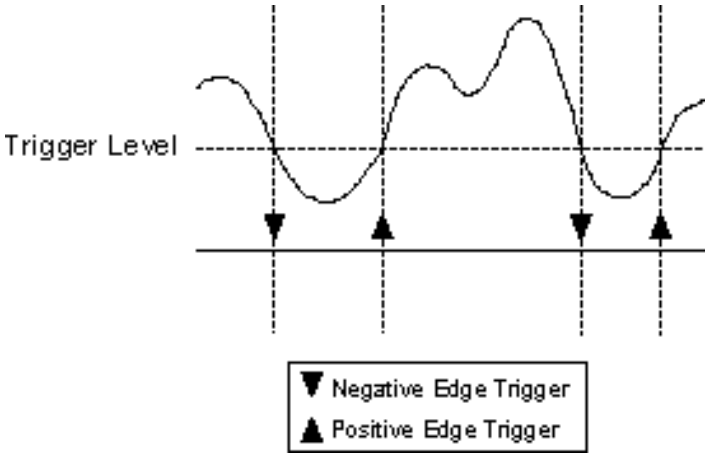

Figure 15. A trigger is registered whenever the analog signal passes the specified level.

Analog window triggers specify both a high and low level to create a "window." When the input signal enters or leaves the window, a trigger event is registered.

Figure 16 shows an entering window trigger:

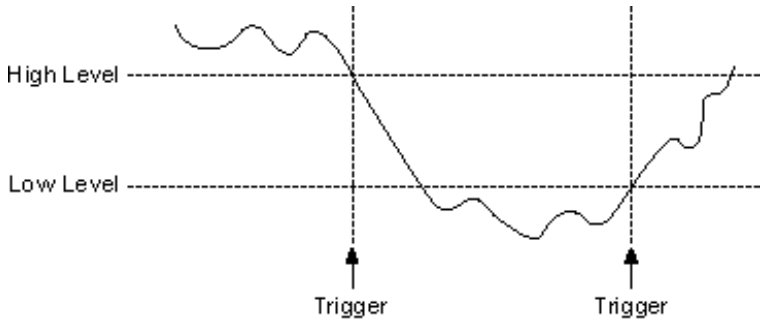

Figure 16. A trigger is registered whenever the signal enters the specified window.

Figure 17 shows a leaving window trigger:

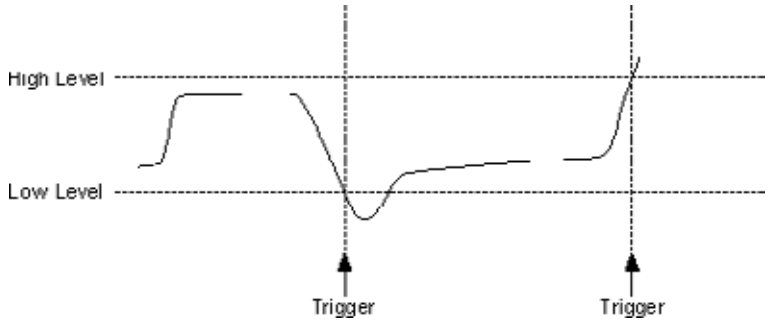

Figure 17. A trigger is registered whenever the signal leaves the specified window.

Analog hysteresis triggers specify a trigger level and a hysteresis value that can be either above or below the trigger level. These triggers help eliminate incorrect triggers caused by noisy signals.

A positive slope hysteresis trigger is generated when the signal crosses below the voltage specified by the trigger level parameter minus the hysteresis value, and then crosses the trigger level.

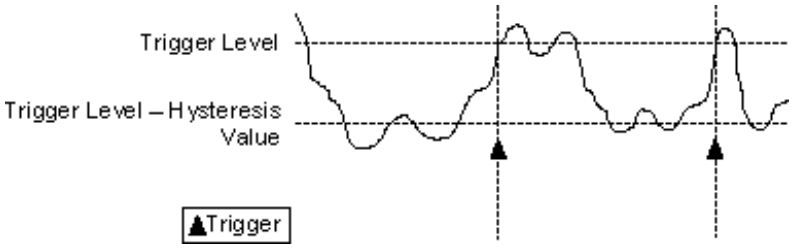

Figure 18. A trigger is registered whenever the signal crosses both the hysteresis and trigger levels.

A negative slope hysteresis trigger is generated when a signal crosses above the voltage specified by the trigger level parameter plus the hysteresis value, and then crosses the trigger level.

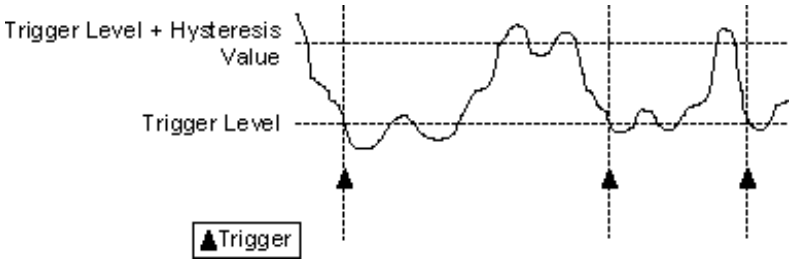

Figure 19. A trigger is registered whenever the signal crosses both the hysteresis and trigger levels.

Other Timing Methods

Software Timing

With software timing, the software and OS determine the generation and acquisition rates of samples. Keep in mind that a hardware clock can run much faster and offer more accuracy than a software loop. Thus, software timing may be the best option for slow-speed applications, noncritical measurements, and devices that do not support hardware timing.

Hardware-Timed Single-Point Timing

Hardware-timed single-point timing acquires or generates samples continuously using hardware timing and no buffer. The primary use case for this timing method is a high-speed control application for which precise loop times must be known. Because there is no buffer, reads and writes must execute fast enough to keep up with the hardware timing of the measurement to avoid overflowing or underflowing the measurement buffer. Because of this, reserve this method for applications with a real-time OS running the acquisition or generation application to maintain a specific loop rate.

Sampling Theory

Sample Rate

The sample rate is the rate at which the ADC converts the analog input waveform to digital data. It is frequently a key specification in the selection of a digitizer because it inversely correlates measurement timing precision to sample resolution. That is, sample rate generally increases at the expense of the number of bits of resolution. This is a logical trade-off because slower sampling provides more time for a high-resolution ADC to perform its digitization as well as any signal conditioning that may occur prior to that point.

Nyquist Sampling Theorem and Aliasing

The Nyquist Sampling Theorem determines the minimum sampling speeds you need in an application to ensure a faithful reproduction of the original signal. It states that as long as the sample rate, f_S, in any system exceeds twice the maximum signal frequency, the original signal can be reconstructed in the receiver with minimal distortion. This frequency is often referred to as the Nyquist frequency, f_N.

$$f_s > 2 * f_N$$

If frequency components in the system are higher than the Nyquist frequency, false lower-frequency components may appear in the sampled data. This phenomenon is referred to as aliasing. Figure 20 shows aliasing in signals A and B, for which the signal is undersampled. Signal C is adequately sampled to preserve the frequency of the original signal.

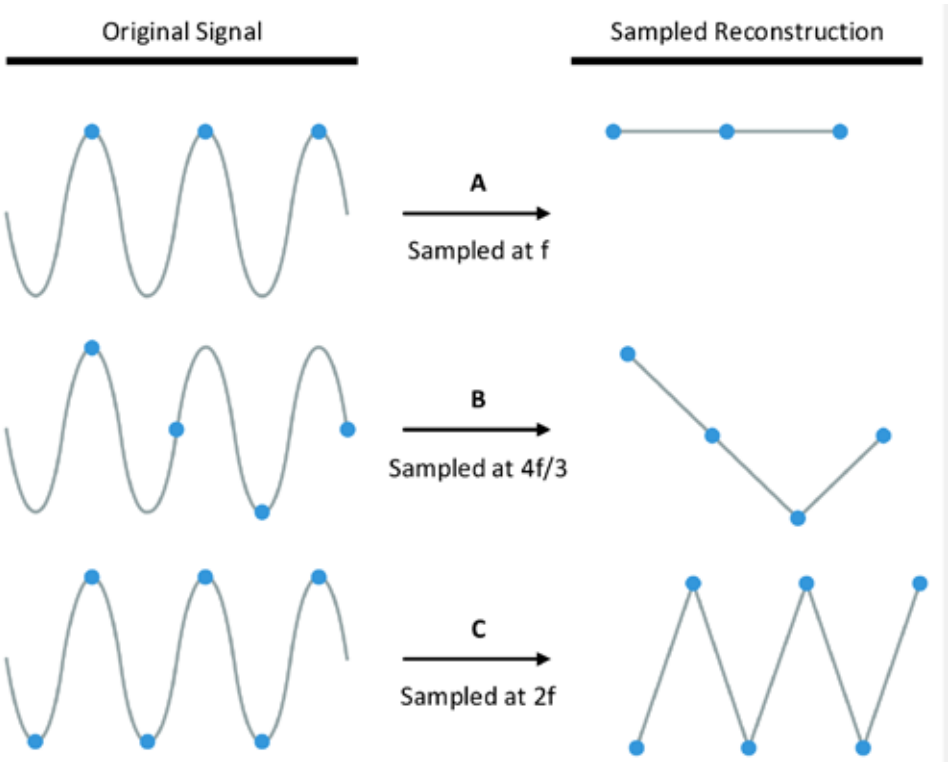

Figure 20. Sampling below the Nyquist frequency can lead to aliased versions of the original signal.

Consider a signal with a sample frequency of 100 Hz, and the input signal contains the following frequencies: 25 Hz, 70 Hz, 160 Hz, and 510 Hz. Frequencies below the Nyquist frequency of 50 Hz are sampled correctly; those over 50 Hz appear as alias.

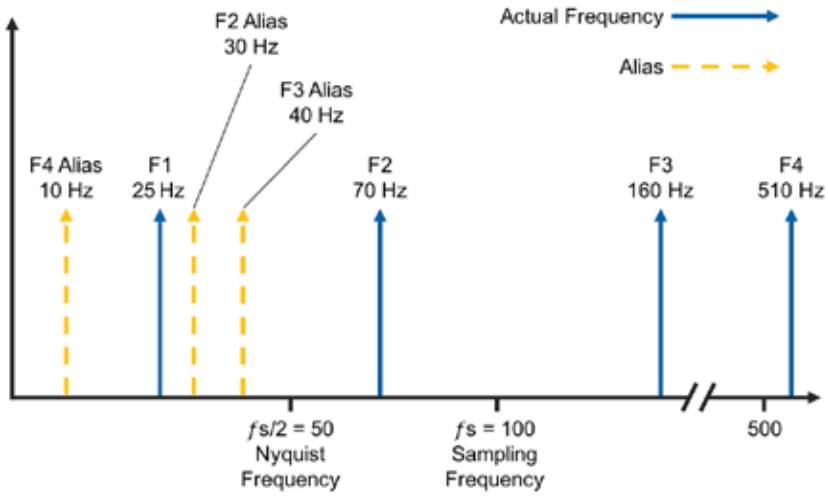

Figure 21. Aliases are reflections of actual frequency components that are being undersampled by an acquisition operation.

Oversampling

Oversampling, or sampling at a rate beyond twice the Nyquist frequency, is recommended to help prevent aliasing and preserve the shape of the original signal. Because real-world signals are not perfectly filtered, they often contain frequency components greater than twice the critical frequency of interest. You can use oversampling to increase the Nyquist frequency (one half the sample rate) and reduce the possibility of aliasing in these higher frequency components. Oversampling is also necessary when you want to capture fast edges, transients, and one-time events.

Choosing a Sample Rate

The Nyquist Sampling Theorem states that you can accurately reconstruct a signal by sampling two times the highest frequency component of interest. However, in practice, you should sample at least 10 times the maximum frequency to represent the shape of your signal. Choosing a DAQ device with a sample rate at least 10 times the frequency of your signal ensures that you measure or generate a more accurate representation of your signal.

For example, suppose you want to measure a sine wave that has a frequency of 1 kHz. The Nyquist Sampling Theorem says you must sample at 2 kHz minimum, but it is highly recommended to sample at 10 kHz to measure or generate a more accurate representation of your signal. Figure 22 compares a 1 kHz sine wave measured at 2 kHz and 10 kHz.

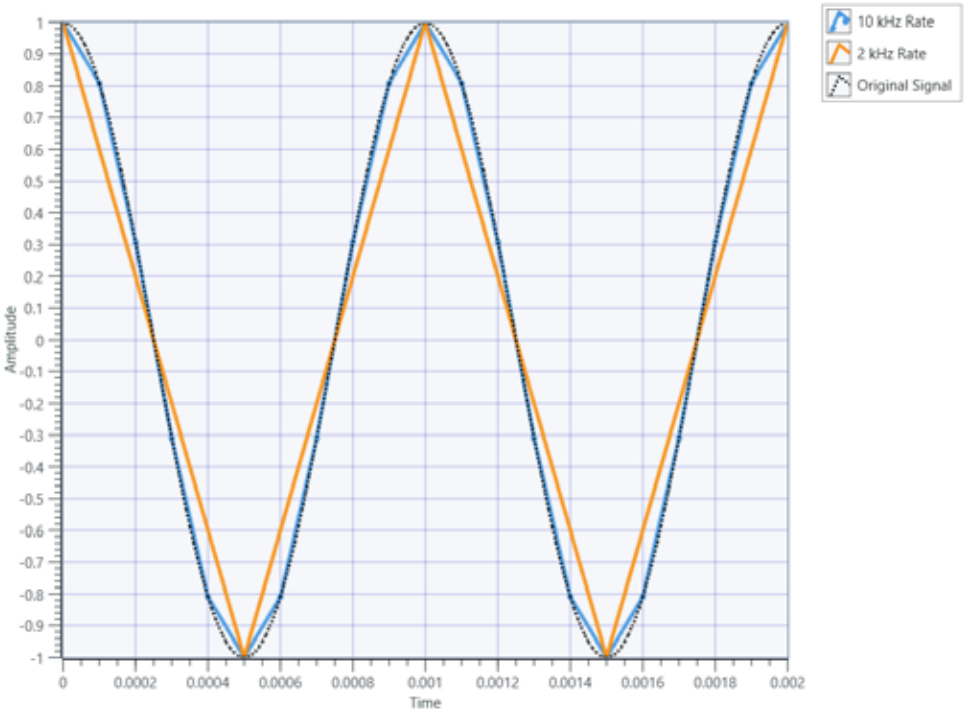

Figure 22. Increasing the sample rate creates a more accurate representation of the shape of a signal.

One downside of oversampling is that higher sample rates consume more processor resources. Every DAQ device has both a maximum and minimum sample rate, and selecting a device with the appropriate capability ensures that you adequately characterize signals while being data efficient.

Accuracy of Acquisition Timing Engines

As discussed in the Digitization Timing Architectures section, sample rate is driven by a digital pulse sequence called a sample clock, which is typically part of a timing engine on the measurement device. The sample clock is often derived from a faster time source that can be divided down to achieve the requested sample rate for a given measurement. This time source should generally be stable, but, as with analog digitization circuitry, you always need to consider resolution and accuracy.

The resolution of the time source dictates the actual sample rate. For a DAQ device with a timing resolution of 50 ns, for example, you can request a sample rate of 1 MHz (1000 ns sample period). A sample rate of 900 kHz (~1111 ns period), however, would be coerced up to ~909.09 kHz (1100 ns period).

Another way to think of it is that the sample clock is a divide down from a 20 MHz timebase, so you can choose only rates that are integer divisors of 20 MHz. For example, 20 MHz/22 = ~909.09 kHz. Devices with lower timing resolution have a faster timebase from which to derive the clock signals.

Timing accuracy, on the other hand, is a measure of how close the actual sample rate is to the requested sample rate and is generally specified in parts per million (ppm). For example, a timing accuracy of 50 ppm means that the sample rate error can be up to:

$$\frac{50}{1{,}000{,}000} \cdot DesiredSampleRate$$

So, for a sample rate of 1 kHz, the same rate error is 0.05 Hz. This means the actual sample rate is 1000 ± 0.05 Hz.

Bus Choice and Buffering

All PC buses have a limit to the amount of data that can be transferred in a certain period of time. Known as the bus bandwidth, this limit is often specified in megabytes per second (MB/s). If dynamic waveform measurements are important in your application, be sure to consider a bus with enough bandwidth.

Depending on the bus you choose, the total bandwidth can be shared among several devices or dedicated to certain devices. The PCI bus, for example, has a theoretical bandwidth of 132 MB/s that is shared among all PCI boards in the computer. Gigabit Ethernet offers 125 MB/s shared across devices on a subnet or network. Buses that offer dedicated bandwidth, such as PCI Express and PXI Express, provide the maximum data throughput per device.

When taking waveform measurements, you need to achieve a certain sample rate and resolution based on how fast your signal is changing. You can calculate the minimum required bandwidth by multiplying the number of bytes per sample (rounded up to the next byte) by the sampling speed and then by the number of channels.

For example, a 16-bit device (2 bytes) sampling at 4 MS/s on four channels would be:

$$\frac{2 \text{ bytes}}{S} * \frac{4 \text{ MS}}{\text{sec}} * 4 \text{ channels} = 32 \text{ MB/s}$$

Your bus bandwidth needs to support the speed at which you are acquiring data. Note that the actual system bandwidth is lower than the theoretical bus limits. Actual observed bandwidth depends on the number of devices in a system and any additional bus traffic from overhead. If you need to stream a lot of data on many channels, bandwidth may be the most important consideration when choosing your DAQ bus.

Table 5 shows several different buses and different factors to consider when choosing a bus.

● = Best | ◐ = Better | ○ = Good

Bus	Waveform Streaming	Latency	Portability	Distributed Measurements
PCI	132 MB/s (shared)	●	○	○
PCI Express	250 MB/s (per lane)	●	○	○
PXI	132 MB/s (shared)	●	◐	◐
PXI Express	250 MB/s (per lane)	●	◐	◐
USB	60 MB/s	◐	●	◐
Ethernet	125 MB/s (shared)	○	●	●
Wireless	6.75 MB/s (per 802.11g channel)	○	●	●

Table 5. Different bus types provide different benefits for data transfer and usability.

The sample rate for an application also has implications on the buffer size. A buffer is temporary storage in a computer's memory for acquired or to-be-generated samples. Typically, this storage, also called the task buffer, is allocated from the computer's memory. For input operations, a data transfer mechanism transfers samples from the device into the buffer, where they wait for the application to pull them out of the buffer. For output operations, the application copies samples into the buffer, where they wait for the data transfer mechanism to transfer them to the device.

The four main data transfer mechanisms are:

1. Direct Memory Access (DMA)—DMA transfers data between the device and computer memory without the involvement of the CPU. Because of this, DMA is the fastest data transfer mechanism. NI uses DMA hardware and software technology to achieve high-throughput rates and increase system utilization. It is the default data transfer method for DAQ devices that support it. However, each device typically has a limited number of DMA channels.
2. Interrupt Request (IRQ)—IRQ transfers rely on the CPU to service data transfer requests. The device notifies the CPU when it is ready to transfer data. The data transfer speed depends on the rate at which the CPU can service the interrupt requests. If you are using interrupts to acquire data at a rate faster than the CPU can service the interrupts, system performance may be reduced.
3. Programmed I/O—Programmed I/O does not use a buffer; instead, the computer reads and writes directly to the device. Software-timed (on-demand) operations typically use programmed I/O.

4. USB Bulk—USB Bulk is a buffered, message-based streaming mechanism for data transfer. This high-speed method is the default transfer mechanism for USB devices.

The larger the sample rate, the more samples are acquired in the same amount of time and the faster a buffer fills up. Once a buffer fills up, one of several different errors may occur. An overwrite error is the most common error when performing a circular buffered acquisition. The error indicates that information is lost and occurs when the application does not read data from the PC buffer quickly enough. Samples written to the circular PC buffer are overwritten before they are read into application development environment (ADE) memory. Avoiding an overwrite error requires a larger buffer size, more frequent reads, or slower writes.

An overflow error is also common when performing a circular buffered acquisition. Overflow errors are more serious than overwrite errors because they indicate that information has been lost earlier in the data acquisition process. Overflow errors show that the first-in-first-out (FIFO) memory buffer on board your DAQ device has reached its maximum capacity for storing acquired samples and can no longer accept new samples. An overflow error is symptomatic of a bus transfer rate that falls short of the requested data input rate. Using a DMA transfer, decreasing the requested data input rate, or reducing the number of devices sharing the PCI bus can help avoid overflows.

Increasing Effective Sample Rate

You can functionally increase sample rates through a process called interleaving. Interleaving uses multiple ADCs to sample the same input waveform but at different relative phases. The clocks of these ADCs are shifted to a phase that reflects the time when an ADC of a higher sample rate would acquire a point. The hardware then interleaves these samples to create the waveform as if only one ADC were sampling the waveform at a higher sample rate. Figure 23 shows an example.

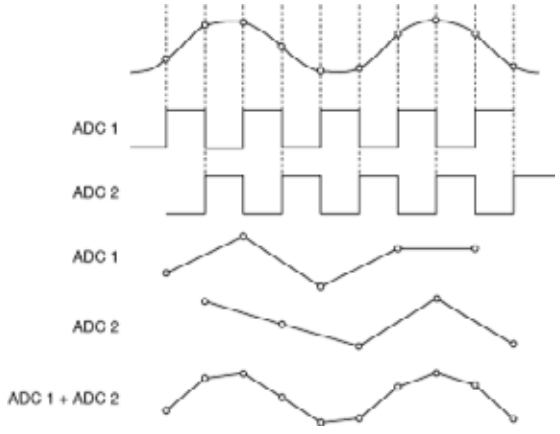

Figure 23. Using two ADCs to sample the same signal at a phase offset effectively doubles the sample rate.

The data returned is in order and acquired in real time; therefore, the input signal is not required to be repetitive. Interleaved sampling is more common in high-speed acquisition applications for which a device with twice the sampling capability may be prohibitively expensive or may not exist. Many of these use pipeline ADCs.

Accuracy and Resolution

Digitization Resolution

Resolution refers to the voltage of one ADC code, where a "code" is a digital representation of an analog voltage. The bitness of an ADC indicates its ability to identify discrete voltages within the operating input range of the device. You can easily calculate the number of codes for an ADC by raising 2 to the power of the number of bits of the ADC:

$$\#\ of\ Codes\ =\ 2^{\#\ of\ bits}$$

For a 16-bit ADC, the number of codes is:

$$\#\ of\ Codes\ = 2^{16} =\ 65{,}536\ codes$$

To understand how the bitness of the ADC translates to the precision of measured voltages, you need to know the input range of the measurement hardware. Many DAQ devices have selectable input ranges, so you can choose the smallest input range that can measure the signal.

To illustrate this point, consider a 3-bit DAQ device with two input ranges: ±10 V and ±2 V (Figure 24).

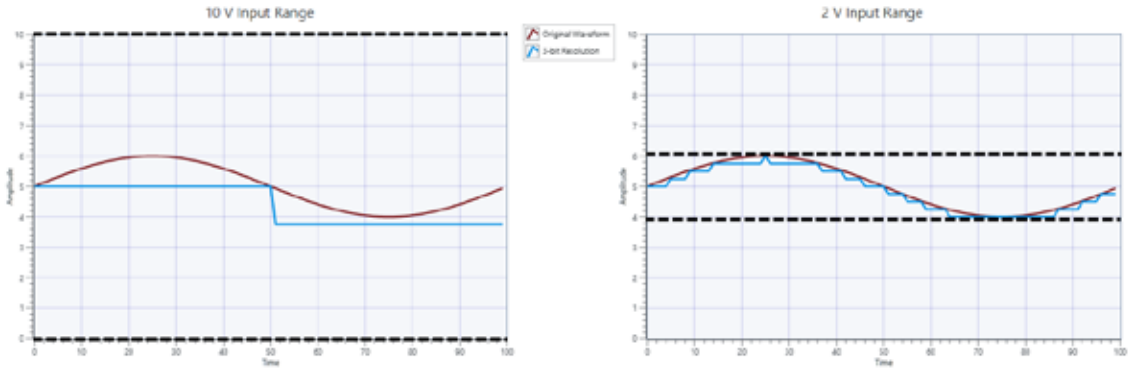

Figure 24. A device's resolution is spread over the full input range being used.

You can calculate the voltage of each code, known as the code width, as follows:

$$\frac{\text{Max V - Min V}}{2^{\#\ of\ bits}} = Code\ Width$$

For a 16-bit ADC with ±10 V and ±1 V input ranges:

Range	±10 V	±1 V
Code Width	$\frac{10\ V - (-10\ V)}{2^{16}} = 305\ \mu V$	$\frac{1\ V - (-1\ V)}{2^{16}} = 30.5\ \mu V$

Changing the input range also changes the level of noise inherent in the measurement system. Attempting to measure a small voltage signal with a ±10 V input range may provide a relatively poor signal-to-noise ratio. Reducing the input range to be more appropriate for the measured signal can reduce the noise to improve such a low-level measurement. However, as ADC quality has improved and ADC cost has decreased over the years, the need for programmable input ranges has become less critical.

Improving Effective Resolution

If you must use a lower resolution device because of design constraints, but you want a higher effective resolution across the measurement range, consider two additional signal manipulation options: averaging and dithering.

First try a signal with small enough variations that it does not exceed the code width of a single code of the ADC you are using. Since the ADC is capable of only placing this signal into a single code "bucket," the resulting measurement does not represent the original signal.

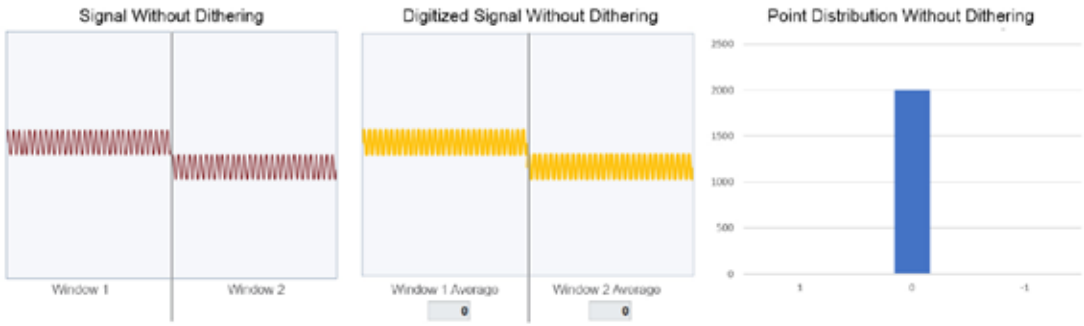

Figure 25. Small changes in a signal can be missed when they fall below the smallest detectable change of the device.

One solution to this is a measurement device with a higher resolution, but this may not be practical for several reasons including cost. As an alternative (and if supported by the DAQ device), you can enable dithering to add uniform white noise to the signal before digitization. When averaged, uniform noise is largely negated.

Since the original signal is now effectively amplified uniformly, it can cross into multiple codes of the ADC over the span of several samples. By then averaging these chunks of acquired samples and removing the injected noise, you achieve a more precise representation of the original signal without changing the ADC.

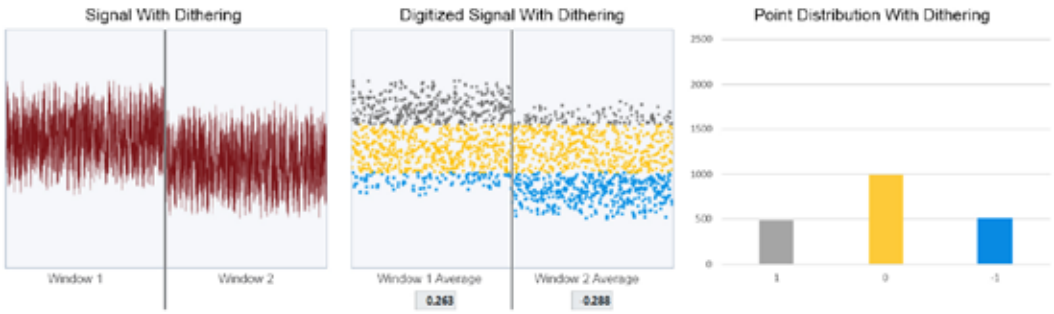

Figure 26. Mixing uniform noise with a small signal allows for a more reliable distribution of data points that can be averaged together in post processing.

Fortunately, many modern ADCs (typically SAR-based) have enough resolution—and, therefore, a small enough code width—that the noise inherent in the device and environment is enough to effectively dither the acquired signals and, thereby, improve effective precision when you average multiple samples together. Even without the ability to actively dither an input signal and without prior knowledge of the noise content, you can use averaging to help smooth the measured waveform while reducing the size of the resulting data set. On the previously discussed frequency versus resolution curve, averaging has the impact shown in Figure 27:

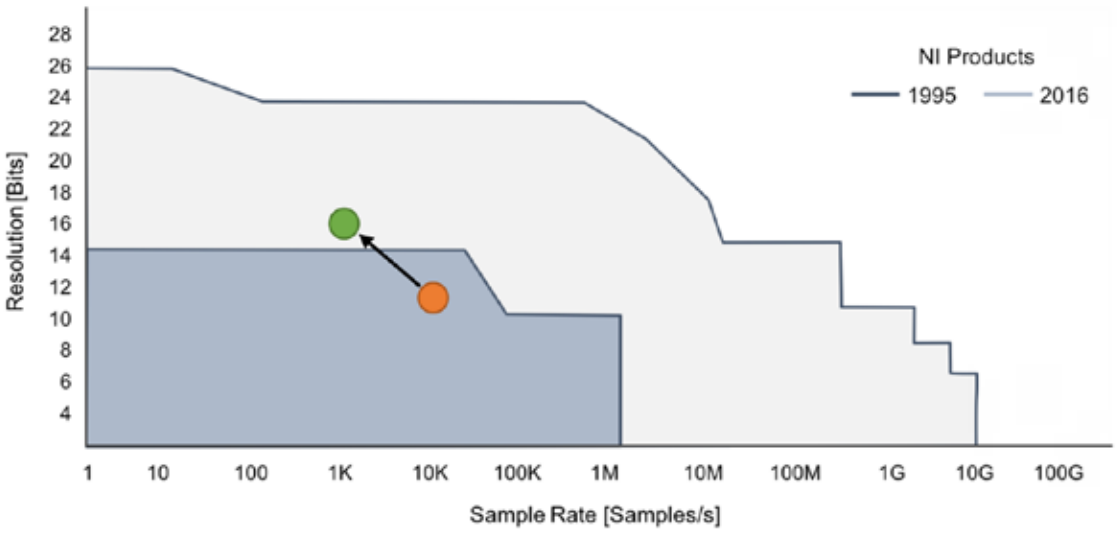

Figure 27. The effective resolution of a device can be increased by oversampling and averaging.

For measurement applications that need more precision, you may want to try oversampling the measurement as much as the device and data transfer bus allow without raising buffering concerns. Then you can improve the effective precision of the measurement by averaging the oversampled data down to your actual desired sample rate.

Measurement Accuracy

Though code width is the smallest detectable voltage an ADC can digitize for a given input range, it is not an indication of the accuracy of that measurement. You need to understand the difference between precision and accuracy in the context of analog measurements:

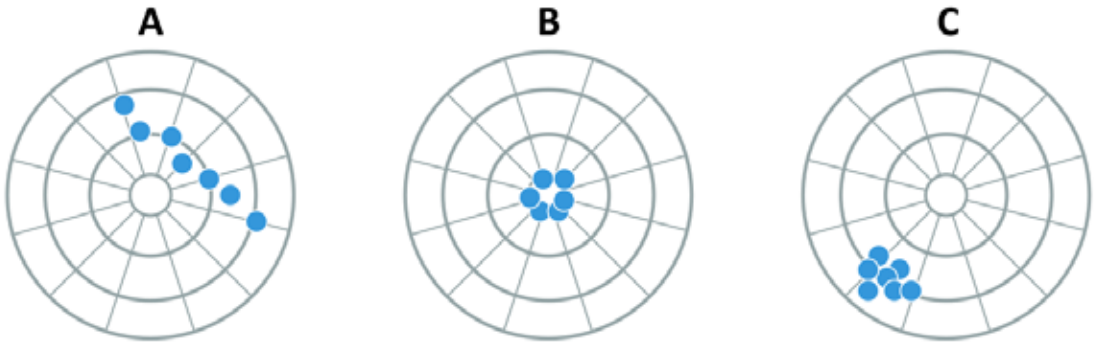

Figure 28. Precision Vs. Accuracy by CK-12 Foundation (raster); User: Adrignola (vector)—File: High School Chemistry.pdf, page 107, Public Domain, https://commons.wikimedia.org/w/index.php?curid=16253226

A high-resolution DAQ device may be able to reliably identify measured voltages within a range as shown in Figure 28 (c). However, this offers no information on how close these voltages are to their actual analog values, as shown in Figure 28 (b). Several elements, from the sensor, to the wiring, to the analog circuitry of the DAQ hardware, can play a role in determining the absolute accuracy of a DAQ measurement.

High-quality measurement hardware characterizes this accuracy, and this information is included with the device's specifications. The absolute accuracy of a device should account for several potential sources of error tied to the analog input circuitry:

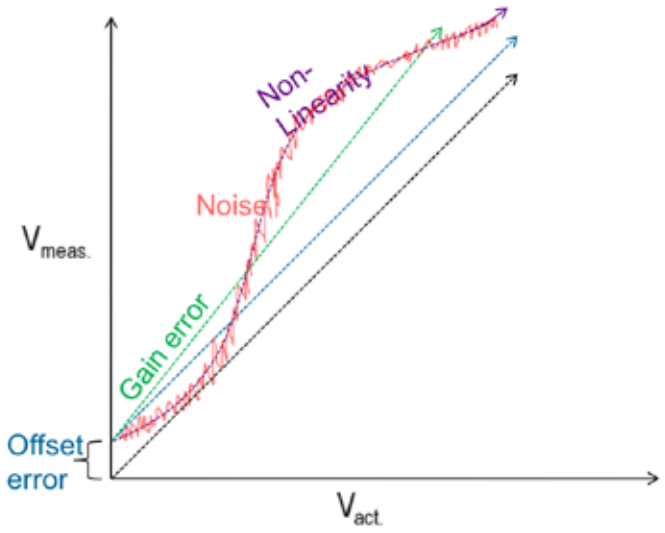

Figure 29. The four types of error compound to result in the final measurement capability.

In Figure 29, the black line indicates the measured voltage relative to the actual voltage, which is a 1-to-1 correlation in the ideal case. Since a real DAQ system is never ideal, offset, gain, nonlinearity, and noise error all compound along with the actual voltage to produce a measured voltage. If these sources of error are properly characterized for the operating ranges of a DAQ device, you can calculate the absolute accuracy. Table 6 is an analog input absolute accuracy table from the specifications of an NI DAQ device:

AI Absolute Accuracy table

Nominal Range		Residual Gain Error (ppm of Reading)	Gain Tempco (ppm/°C)	Reference Tempco	Residual Offset Error (ppm of Range)	Offset Tempco (ppm of Range/°C)	INL Error (ppm of Range)	Random Noise, σ (µVrms)	Absolute Accuracy at Full Scale (µV)	Sensitivity (µV)
Positive Full Scale	Negative Full Scale									
10	-10	75	25	5	20	57	76	244	3,100	97.6
5	-5	85	25	5	20	60	76	122	1,620	48.8
1	-1	95	25	5	25	79	76	30	360	12.0
0.2	-0.2	135	25	5	80	175	76	13	112	5.2

AbsoluteAccuracy = Reading • (GainError) + Range (OffsetError) + NoiseUncertainty

GainError = ResidualAIGainError + GainTempco • (TempChangeFromLastInternalCal) + ReferenceTempco • (TempChange FromExternalCal)

OffsetError = ResidualAIOffsetError = OffsetTempco • (TempChangeFromLastInternalCal) + INL_Error

$$\text{NoiseUncertainty} = \frac{\text{RandomNoise} \bullet 3}{\sqrt{100}}$$ For a coverage factor of 3 σ and averaging 100 points.

Table 6. A sign of a high-quality data acquisition device is a thorough accuracy specification.

You can factor the accuracy specifications in each column into the absolute accuracy equation to determine the absolute accuracy at full scale. In the Table 6 case, a measurement taken at the ±10 V range will be accurate to within 3.1 mV assuming the device is not out of calibration.

Practical Example: Absolute Accuracy

Consider the device profiled in Table 6. You can calculate absolute accuracy assuming the following:

- ±0.2 V input range
- 0.2 V (full scale) reading
- Last external calibration temperature of 23 °C
- Last internal calibration temperature of 24 °C
- Current temperature of 29 °C

$$AbsoluteAccuracy = 0.2\ V \bullet (GainError) + 0.2 \bullet (OffsetError) + NoiseUncertainty$$

$$GainError = 135\ \text{ppm} + 25 \bullet (29\ °C - 24\ °C) + 5 \bullet (29\ °C - 23\ °C) = 290\ \text{ppm}$$

$$OffsetError = 80\ \text{ppm} + 175\ \text{ppm} \bullet (29\ °C - 24\ °C) + 76 = 1031\ \text{ppm}$$

$$NoiseUncertainty = \frac{RandomNoise \bullet 3}{\sqrt{100}} = \frac{13\ \text{ppm} \bullet 3}{\sqrt{100}} = 3.9\ \mu V$$

$$AbsoluteAccuracy = 0.2\ V \bullet 290\ \text{ppm} + 0.2\ V \bullet 1031\ \text{ppm} + 3.9\ \mu V = 268.1\ \mu V$$

So, measuring 5 V with the 5 V scale on this device should be accurate to within ±268.1 µV. Performing a self-calibration, which is discussed later in this guide, helps improve this number slightly, and an external calibration improves it further still.

Think of the nature of absolute accuracy in digitization as more than the resolution of the ADC you are using. You should consider that the full circuit, like the other circuitry in the analog signal chain (amplifiers, multiplexers, and so on), will have errors due to manufacturing variation and component tolerances as well as temperature-induced errors. These errors can compound to be significant. Consider the devices in Table 7:

	Device 1	Device 2
Resolution	18-bit	16-bit
Channels	64	32
Sample Rate	750 kS/s/ch	1 MS/s
Input Range	±10 V	±10 V
Accuracy	**2.20 mV** full scale	**1.66 mV** full scale

Table 7. The quality of a DAQ device cannot be determined by its resolution alone.

AC and DC Accuracy

Accuracy as defined in the previous sections generally refers to DC accuracy. This is because most DAQ devices are intended to provide low-noise, high-resolution, DC-accurate measurements. When considering applications for which AC performance is key, be sure to explore digitization hardware designed with AC accuracy in mind. These devices should specify accuracy in both DC and AC terms to ensure the digitized signals are within tolerance of a particular application. These applications typically involve signals with higher frequency components.

AC accuracy specifications often look like Table 8. Accuracy varies at different input ranges and signal frequencies, and is valid for some calibration interval and over some temperature deviation, in this case, two years and 5 °C.

Range (rms)	Peak Voltage	1 Hz to 40 Hz	>40 Hz to 20 kHz	> 20 kHz to 50 kHz	> 50 kHz to 100 kHz	> 100 kHz to 300 kHz
50 mV	±105 mV	0.1 + 0.02	0.05 + 0.02	0.07 + 0.02	0.3 + 0.02	0.7 + 0.15
50 mV	±1.05 V	0.1 + 0.005	0.05 + 0.005	0.06 + 0.01	0.2 + 0.01	0.7 + 0.15
5 V	±10.5 V					
50 V	±105 V	0.1 + 0.005	0.06 + 0.01	0.12 + 0.05	0.6 + 0.05	3 + 0.15
700 V	±1000 V					

Table 8. AC Voltage Accuracy ± (% of Reading + % of Range), 2 Years, $T_{selfcal}$ ± 5 °C

Calibration

For most DAQ hardware, calibration is the process of comparing a known voltage source with a measured voltage to set adjustment constants that can correct for any errors in the environment from that time onward. High-quality DAQ devices often feature an onboard precision source to allow for self-calibration at run time. Using this source periodically ensures you can account for environmental changes such as temperature drift.

Specifically, self-calibration can correct for the impact of temperature change on gain and offset error, as defined in the absolute accuracy section earlier in this document. Though you have no hard-and-fast rule on when to self-calibrate, you should be able to quantify the impact on the accuracy of a given device using the absolute accuracy equations and coefficients provided by a reputable DAQ vendor. With this information in hand, you can establish temperature change limits beyond which you should perform a self-calibration.

For example, in the accuracy calculation of the previous section, performing a self-calibration improves the full-scale absolute accuracy from 268 µV down to 68 µV.

For effective calibration, you need to thoroughly understand a DAQ device's architecture to correct for all the types of nonlinearities inherent in measurement devices. Periodically, you may also need an external calibration for which you reference a high-precision external voltage source to account for changes in the circuitry performance over time. The resulting correction coefficients, when applied to the device, should ensure that any guaranteed specifications are still valid. Often, DAQ devices specify an interval for calibration to ensure a device does not perform outside its specified tolerances.

Compliant calibration, often referred to as "external" calibration, ensures that a device meets the needs of more advanced quality standards. It should be performed by a lab that is accredited to ISO 17025 with service that complies with ANSI/NCSL Z540-1-1994.

Understanding Noise

What Is Noise?

Noise can be defined many ways, but this guide refers to it as any undesirable electrical signal that interferes with the signal of interest. Noise can be constant, periodic, and even completely sporadic, but it is always present in a DAQ system. The signal of interest is defined as the electrical signal produced by either a device under test or a sensor set up to measure the physical phenomenon, as discussed at the beginning of the guide. Noise is undesirable because it can alter the signal of interest in a manner that makes characterizing and processing the signal extremely difficult or even impossible. Unfortunately, identifying, characterizing, and solving all problems related to measurement noise are not within the scope of this guide; however, the guide examines some introductory topics to ensure you initially select the correct DAQ device.

Sources of Noise

Noise will always be part of a practical DAQ system, but a thorough understanding of its source, coupling channel, and receiver can help you decrease its effects on the signal of interest.

- The source of noise can be anything from AC power cables to electrical transients caused by lightning strikes.
- The coupling channel, on the other hand, can be scoped to conductive, capacitive, inductive, or radiative coupling.
- The receiver can be any affected portion of the DAQ system that also includes the cabling or wiring between the components of the DAQ system (for example, wiring that travels from the strain gage to the signal conditioning of the DAQ device).

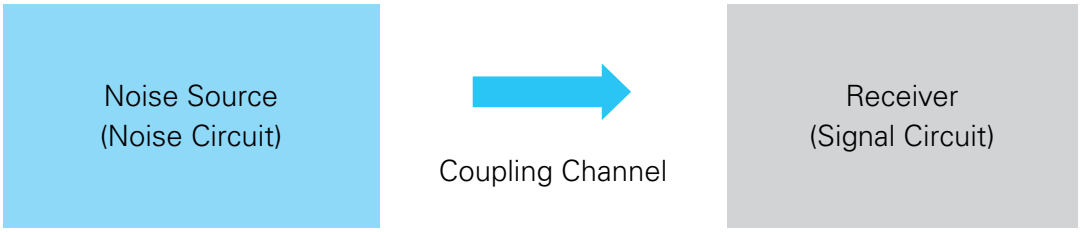

Figure 30. Noise is coupled into a signal by a variety of methods.

Consider the following common types of noise.

Environmental Noise

Environmental noise is an intentionally broad term. As stated earlier, noise sources can vary from application to application and location to location. The term is used generally because the noise produced by the collection of items in the environment is part of, and unique to, that environment. Noise sources often must be identified before a tactic can be installed to reduce their effects on the signal of interest. A common example of noise in a signal's environment is the 50 Hz or 60 Hz noise generated by the power infrastructure in a building.

Identifying the noise source in each environment can be key to taking a measurement with minimal noise. It is also necessary because the solution can be as simple as distancing two cables from each other or as complex as using some of the noise management techniques shown in the managing noise section.

Crosstalk Noise

Crosstalk is a term used in many applications with slightly different implications for each. This guide refers to crosstalk as the transfer of unwanted energy from one conductor to another conductor. In terms of data acquisition, the most common crosstalk sources reside in the DAQ device itself through the physical component layout and the cabling that transfers the signals to the DAQ device.

The device manufacturer determines the DAQ device layout during design; therefore, you should inspect and compare the specifications carefully for the DAQ hardware you are considering. Crosstalk specifications are often specified in decibels (dB) of transmission from one active channel to the active channel with the signal of interest using the following equation:

$$\text{Crosstalk (dB)} = 10 * \log \left(\frac{Power_{\,Measured\ Signal}}{Power_{Noise\ Source}} \right)$$

The lower the crosstalk value in the device specifications, the more adept a device is at shielding adjacent signals from each other. The entry below shows an example crosstalk specification for a DAQ device:

Crosstalk (at 100 kHz)	
Adjacent channels	-75 dB
Nonadjacent channels	-95 dB

Crosstalk present due to cabling is not inherent to the DAQ device you select, but you can take a few quick precautions to minimize crosstalk. When measuring small signals, attempt to keep the cabling carrying these small signals a large distance from higher voltage signals or switching digital signals. Twisted wiring and shielded cables are designed to minimize crosstalk and should be used whenever possible.

Measurement Noise

A lot of time and effort go into the analog design of the signal path before an ADC is inserted in a DAQ device, but these systems are not perfect. All DAQ devices have noise inherent to the analog signal path and the ADC used for the digitization of the signal. Note in the practical example for absolute accuracy that the equation for accuracy uses a term for noise uncertainty. Noise uncertainty in the equation is defined as:

$$Noise\ Uncertainty = \frac{Random\ Noise * 3}{\sqrt{100}}$$

This term is used to account for the random noise, often Gaussian noise, in the digitization hardware. This random noise term is just one of the terms that describe the performance of digitization at that nominal range.

The random noise introduced by the measurement hardware is an important specification when comparing DAQ devices and architectures. Most importantly, you should always ensure the performance specifications meet your application needs. One method to minimize this measurement noise is by effectively using the DAQ device's input ranges. As the input range decreases in size, for example, from -10 V to 10 V to -1 V to 1 V, the random noise decreases. To take advantage of this DAQ device feature, use the lowest range of the device that properly digitizes the signal of interest.

Signal-to-Noise Ratio

Outside measurement noise, you can rely on additional specifications to pinpoint a DAQ device's capability of measuring the signal of interest accurately enough for a specific application. Note that like all noise specifications, the DAQ device's specifications should be carefully read to ensure you calculate noise specifications with equivalent definitions.

Signal-to-noise ratio, commonly referred to as SNR, is defined by the ratio of the average power of the input signal, the signal of interest, over the average power of noise.

$$SNR = \frac{P_{signal}}{P_{noise}}$$

These terms are often calculated using data collected from a fast Fourier transform (FFT) and an input signal of a specified input voltage and frequency. They are usually expressed using a decibel scale. Since the signal and noise amplitudes are volts or amperes, the SNR calculation can be simplified to the equation below:

$$SNR_{dB} = 20 * \log_{10} \left(\frac{Amplitude_{RMS,signal}}{Amplitude_{RMS,noise}} \right)$$

As a point of reference when comparing different DAQ devices, you can calculate the theoretical SNR, in dB, of an ideal N-bit ADC using the equation below.

$$SNR_{dB} = 6.02 * N + 1.76$$

The signal-to-noise specification provides insight into how easily you can measure and identify the signal of interest based on the root mean square of noise present with a measurement. For more in-depth information on noise, explore related topics such as total harmonic distortion (THD), total harmonic distortion plus noise (THD+N), spurious-free dynamic range (SFDR), and signal-to-noise-and-distortion (SINAD) ratio.

Managing Noise

In addition to choosing hardware that acquires the signal of interest with the necessary accuracy, bandwidth, and noise specifications for accurate characterization, you can use other hardware and software features of a device for better characterization even if noise is present in the environment.

Filtering

As one of the most commonly used signal processing techniques for characterizing a signal, filtering can be implemented in many ways. You can implement filtering in several places in a DAQ system: the analog path before the signal of interest is digitized, within the ADC architecture, and after the signal is digitized by the ADC. Regardless of how you implement the filter, the overall goal of this signal processing technique is to reduce noise as much as possible to enable a more accurate representation of the signal of interest.

The more you know about the signal of interest and the noise in the environment, the more effective the filter you make will be. Consider the idealized example in Figure 31 that shows the power of this signal processing technique. Assume that a 1 Hz sine wave is the output of a sensor and, due to the safety of the occupants of a room, the cabling carrying the signal must be run in the walls alongside the building's electrical infrastructure. This signal is measured and the data looks something like Figure 31:

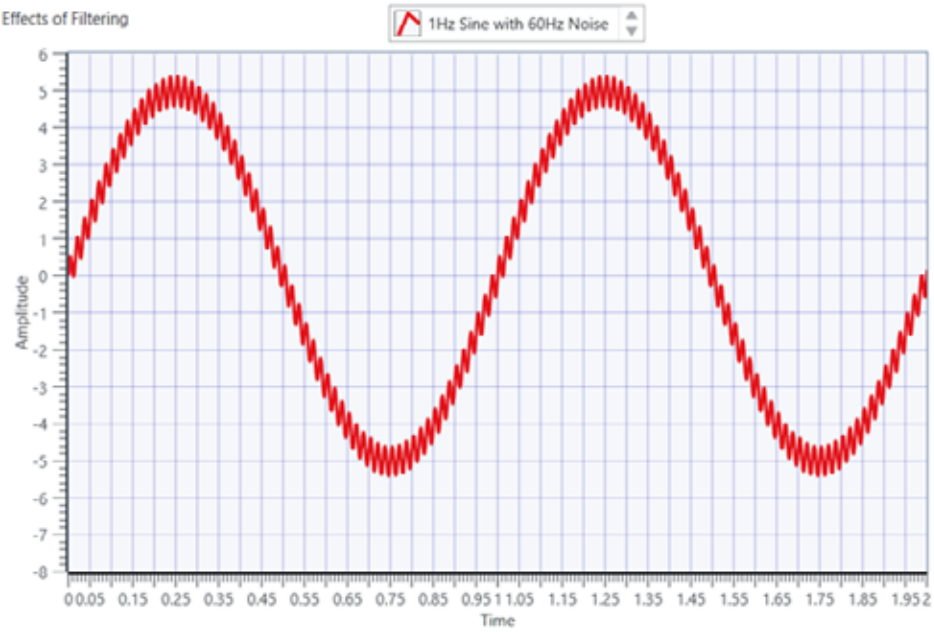

Figure 31. This sine wave shows powerline noise.

An FFT can be an invaluable tool to properly identify the characteristics of the noise that is clearly distorting the output signal of the sensor. Figure 32 shows the output of an FFT formed from the data in Figure 31.

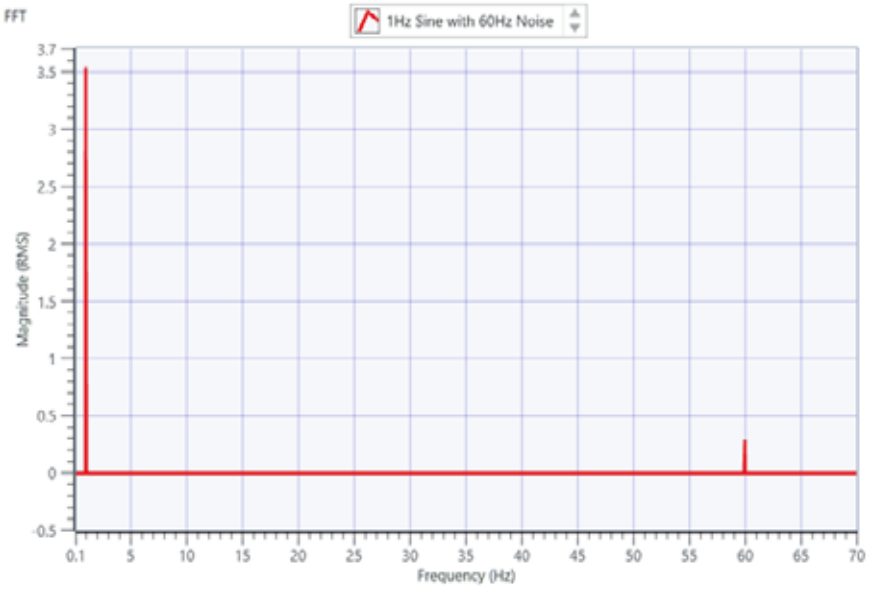

Figure 32. The FFT of a signal with powerline noise shows the characteristic frequency and the 60 Hz powerline frequency.

Note the large magnitude spike from a signal with a frequency of 1 Hz, which is the signal of interest. Quickly surveying the smaller magnitude peak at 60 Hz reveals the main source of noise in this example. The 60 Hz noise, likely the noise coupling onto the signal from the power infrastructure, can be reduced with a filter. In this example, a lowpass filter implemented in software is used to remove the noise and show the effects of this signal processing technique on the acquired signal.

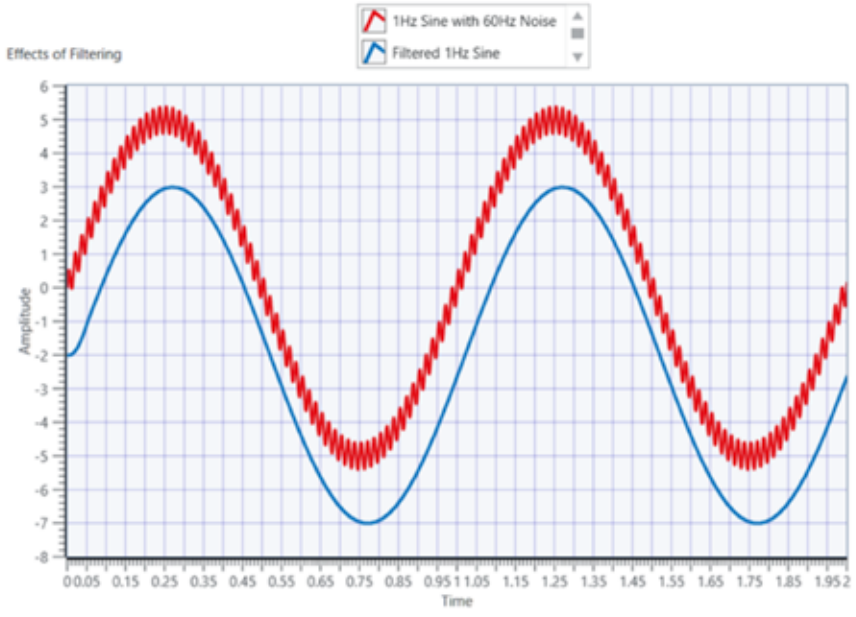

Figure 33. Filtering the signal removes the powerline noise, leaving the signal of interest behind.

Figure 33 shows the results of applying a 4th order lowpass Butterworth filter with a cutoff frequency of 20 Hz to the measured signal (an offset of -2 V was also applied to the filtered signal for visualization purposes; this is not an effect of the filter). With this simple example, you can easily see the power of filtering when encountering noise that interferes with data collection.

Averaging

Another popular signal processing technique that you can use to improve the accuracy of a measurement is averaging. Averaging works well under the premise that noise and measurement errors are random and, therefore, have a Gaussian distribution. By averaging multiple points in the acquired signal with noise, you create a Gaussian distribution from the collection of data points themselves. Calculating the statistical mean, or average, of these multiple points reveals a single point that is statistically close to the actual signal value. In addition, the standard deviation of the multiple points gives the width of the normal distribution around the mean, which describes the probability density for the location of the actual value. As seen in the formula below, the standard deviation is related to the numbers of samples in the average.

$$Standard\ Deviation = \propto \left(\frac{1}{\sqrt{N}} \right)$$

Thus, as the number of samples in the average increases, so does the accuracy of the average.

You should consider several factors before averaging for a signal of interest in an application. Increasing the number of samples in an average increases the probability that the average is closer to the true value, but constraints in the DAQ system or application may limit the maximum number of samples attainable for averaging. As shown in the sampling theory section, you must choose the sample rate based on the frequency content and type of analysis you want to perform on the signal. To comply with these sample rate principles, keep in mind that averaging has a multiplicative impact. For example, if the measured signal is a 10 kHz sine wave, the notable sample rates include the following:

- Sample rate to preserve frequency content (Nyquist frequency)—**20 kHz**
- Sample frequency to preserve shape—**100 kHz**
- Sample frequency to preserve shape with a 10-point average applied—**1 MHz**

Therefore, the entire DAQ system must be capable of a 1 MHz sample rate for each channel you are sampling and averaging, and have the processing power to average the samples as you acquire them.

As with all signal processing techniques, averaging can be powerful, but you should not use the technique in the following cases:

1. If the signal of interest contains transients or high-frequency content, in relation to the sample rate, averaging can affect the frequency information from the signal.
2. Averaging requires many samples to increase the accuracy of the measurement, so it takes time to execute (the time to attain all sampled points in the average and the time for the software to run the averaging algorithm). This slower execution time can be unacceptable in applications that have tight feedback or latency requirements.

3. When using averaging to minimize noise in a signal, you assume that noise has a Gaussian, or random, distribution. If the noise is not random by this definition, using averaging on the measured signal may not provide accurate results. Figure 34 shows how to determine if the noise is Gaussian using a representative graph and comparing the FFT of a 1 Hz sine wave featuring Gaussian noise with a 1 Hz sine wave featuring 60 Hz noise:

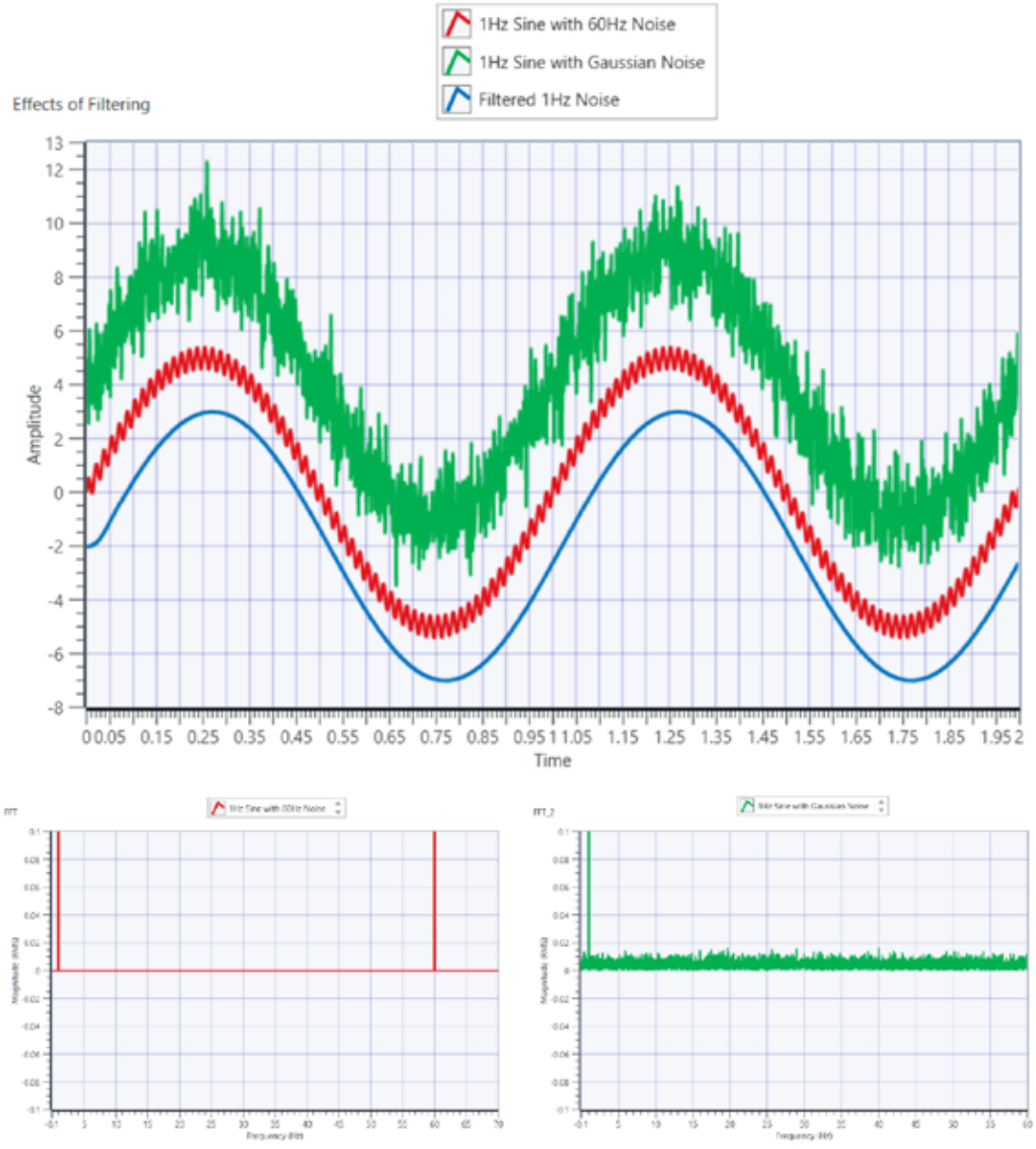

Figure 34. Filtering has an impact in both the time and frequency domains.

To examine the effects of averaging on a highly oversampled 1 Hz sine wave with Gaussian noise, see Figure 35, which uses 40 samples per average:

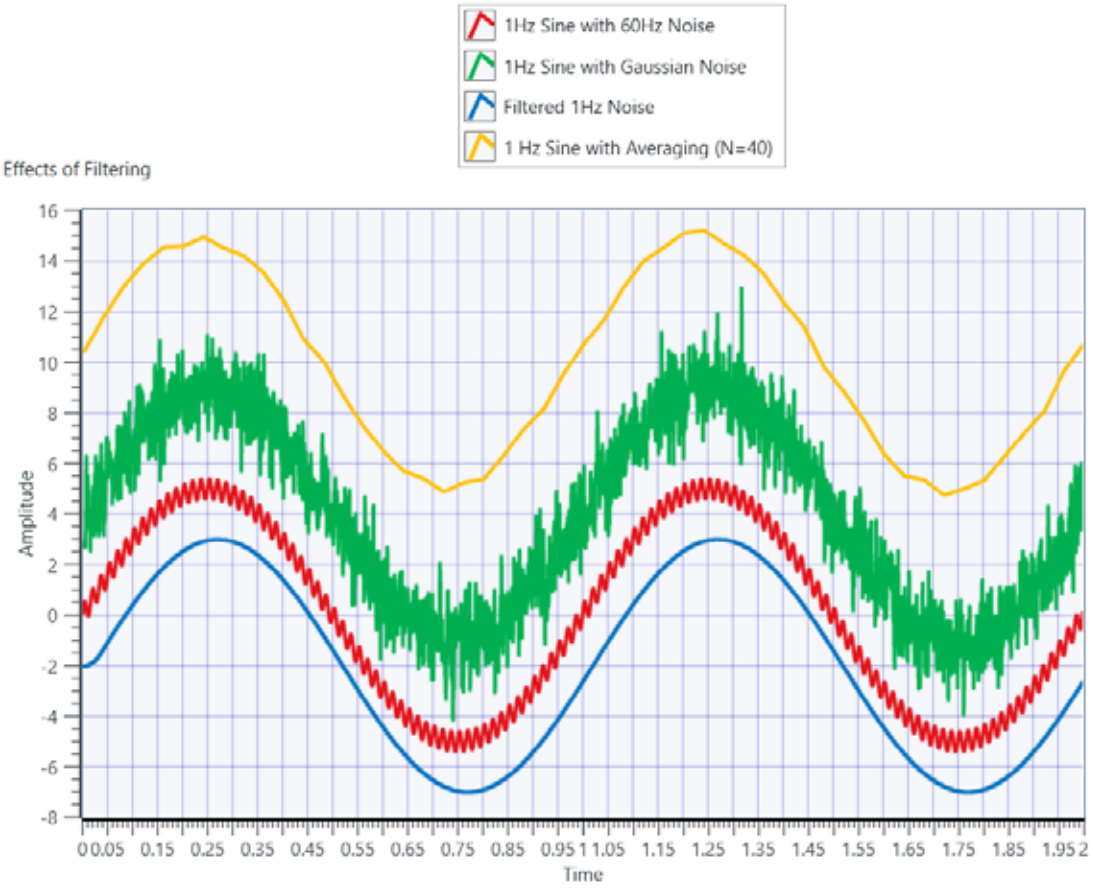

Figure 35. Oversampling and averaging reduces noise that is Gaussian, or evenly distributed, in nature.

NI DAQ Solutions

You depend on measurements to help you make key decisions and discoveries. But with limited tools that cannot meet your specific application needs, you often miss critical data. With a customizable and accurate measurement solution and global support from NI, you can get to the right decision faster.

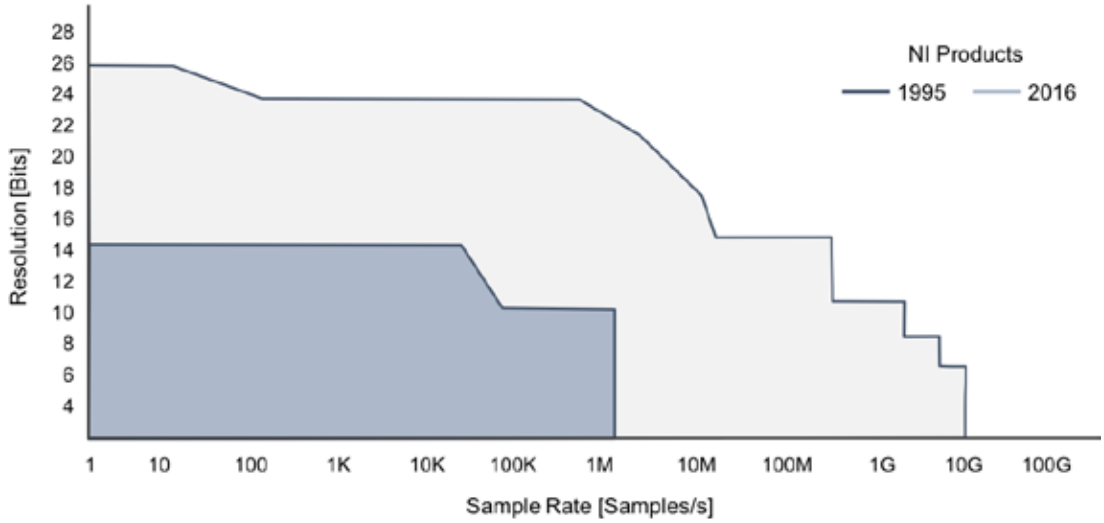

Figure 36. The frequency versus resolution curve continues to be pushed up and to the right.

NI's continual innovation in DAQ technology over two decades pushes the frequency versus resolution curve up and to the right. Figure 36 shows this significant progress. By creating high-quality hardware on any bus or platform that can be tightly coupled with customizable and easy-to-use software, NI has become the de facto choice for DAQ solutions. Browse the NI catalog of DAQ products while considering the information in this guide to help you select the NI DAQ device that best meets your current and future application needs.

NI DAQ Hardware and Software

Why Choose NI DAQ?