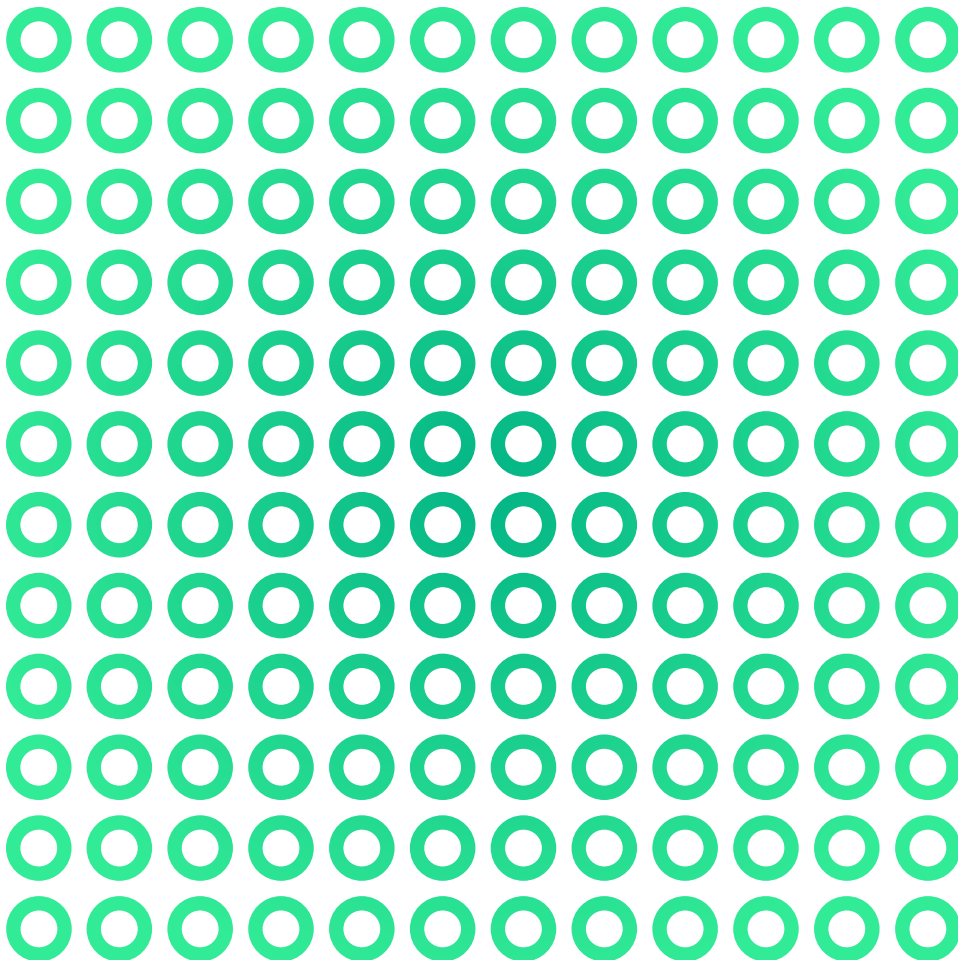WHITE PAPER

The Complete Guide to Building a Measurement System

It's time to get the actionable insight you need from your data. Building a measurement system that will take you into the future involves much more than choosing a DAQ device, you have to consider each part of your system. That includes physical components such as sensors, DAQ devices, and computers, along with software and software components including drivers and reporting tools.

This guide walks walk you through the top seven things you should consider to make the most of your DAQ investment, ensuring that your system is ready for the measurements of today and tomorrow:

How to Choose the Right Sensors

Overview

Before you design a measurement system, you need the right sensors (or transducers). Today's market offers a multitude of sensors that measure different phenomena— you'll even find multiple sensors that measure the same phenomena, including thermocouples and resistance temperature detectors (RTDs) for temperature measurements, for example.

This chapter categorizes and compares the most common sensor types for measuring seven of these phenomena to help you choose the best options for your application:

Temperature

The most common temperature-measurement sensors are thermocouples, thermistors, and RTDs.

Temperature Sensor	Required Signal Conditioning	Accuracy	Sensitivity	Comparison
Thermocouple	• Amplification • Filtering • Cold-junction compensation	Good	Good	• Self-powered • Inexpensive • Rugged • Offers a large temperature range
RTD	• Amplification • Filtering • Voltage excitation	Best	Better	• Very accurate • Very stable
Thermistor	• Amplification • Filtering • Voltage excitation	Better	Best	• High resistance • Low thermal mass

TABLE 1
Common Temperature Sensors

Thermocouples

Thermocouples, the most popular temperature sensors, are effective in applications that require a large temperature range. They are inexpensive ($1 to $50 USD) and have a response time of fractions of a second. Due to material properties, however, it's hard for them to achieve a temperature accuracy of less than 1 °C.

RTDs

RTDs are nearly as popular as thermocouples and can maintain a stable temperature reading for years. In contrast to thermocouples, RTDs have a smaller temperature range (-200 to 500 °C), require current excitation, and have a slower response time (2.5 to 10 s). RTDs primarily are used for accurate temperature measurements (±1.9 percent) in applications that are not time critical. RTDs can cost between $25 and $1,000 USD.

Thermistors

Thermistors have a smaller temperature range (-90 to 130 °C) than the previous sensors. They have the best accuracy (±.05 °C), but are more fragile than thermocouples or RTDs. Thermistors involve excitation like the RTD; however, the thermistor requires voltage rather than current excitation. Typically, a thermistor costs between $2 and $10 USD.

Strain

Strain typically is measured by a resistive strain gage. These flat resistors usually are attached to a surface that is expected to flex or bend—for example, an airplane wing. Strain gages can measure very small surface twists, bends, and pulls. Wiring more than one resistive strain gage together creates a bridge.

Using more strain gages can help you achieve a more sensitive measurement—for example, you can use up to four active strain gages to build a Wheatstone bridge circuit in a full-bridge configuration. You also can choose to use half-bridge (two active strain gages) and quarter-bridge (one active strain gage) configurations. The more active strain gages you use, the more accurate your readings are. Table 2 highlights the different bridge benefits and drawbacks.

Strain gages require current or voltage excitation and are susceptible to temperature drift, bending strain, and axial strain, which can give false readings if you don't use additional resistive strain gages. See Table 2 for common strain gage mounting options.

- Axial bridges measure material stretching or pulling apart.
- Bending bridges measure a stretch on one side of a material and the contraction on its opposing side.
- Torsional and shear bridges measure material twist.

Strain is measured with a dimensionless unit (e or ε), which is equivalent to a small change in length divided by the full length of the object under measure.
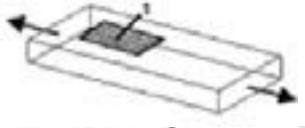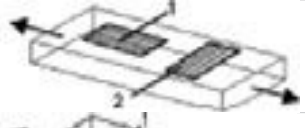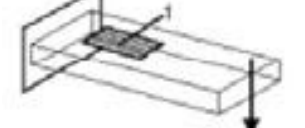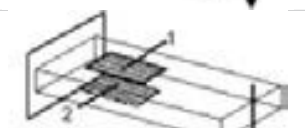
Strain	Gage Setup	Bridge Type	Sensitivity MV/V @100 Ue	Details
Axial		¼	0.5	Good: Simplest to implement, but must use a dummy gage if compensating for temperature. Responds equally to axial strain.
		½	0.65	Better: Temperature-compensated, but sensitive to bending strain.
		½	1.0	Better: Rejects bending strain, but not temperature. Must use dummy gages if compensating for temperature.
		Full	1.3	Best: More sensitive and compensates for both temperature and bending strain.
Bending		¼	0.5	Good: Simplest to implement, but must use a dummy gage if compensating for temperature. Responds equally to axial strain.
		½	1.0	Better: Rejects axial strain and is temperature-compensated.
		Full	2.0	Best: Rejects axial strain and is temperature-compensated. Most sensitive to bending strain.
Torsional and Shear		½	1.0	Good: Gages must be mounted 45 degrees from centerline.
		Full	2.0	Best: Most sensitive full-bridge version of previous setup. Rejects both axial and bending strains.

TABLE 2
Common Strain Gage

Sound

Microphones measure sound. There are several types of microphones to consider when you choose a sensor.

Microphones	Price	Environment	Impedance Level	Sensitivity	Comparison
Prepolarized Condenser	Medium	Tough	Medium	Best	• Most used condenser designs • Best in humid environments
Externally Polarized Condenser	High	Tough	Better	Good	• Most used condenser designs • Best in high-temperature environments
Carbon Microphone	Low	Average	High	Good	• Low quality • Used in early basic telephone handset design
Electret	Low	Average	Low	Better	• Better with high frequencies
Piezoelectric	Medium	Tough	High	Good	• Suitable for shock and blast pressure measurement applications
Dynamic/Magnetic	High	Tough	Medium	Better	• Resistant to moisture • Not good in highly magnetic environment

TABLE 3
Common Sound Sensors

Condenser Microphones

Condenser microphones, the most common microphone, are either prepolarized (meaning that a power source is included within the microphone) or externally polarized. Externally polarized condenser microphones require an additional power source, adding cost. Prepolarized microphones are preferred in humid environments, where a power supply's components could be damaged; externally polarized condenser microphones are preferred in high-temperature environments.

Piezoelectric Microphones

Robust piezoelectric microphones are used in shock and blast pressure measurement applications. These durable microphones can measure high-amplitude (decibel) pressure ranges. Their disadvantage is that they pick up high noise levels.

Dynamic/Magnetic Microphones

In addition to the piezoelectric microphone, dynamic or magnetic microphones function in tough environments. They rely on movement to magnetically induce an electric charge in a way that makes them resistant to water—but obviously, these microphones are not very useful in highly magnetic environments.

Electret Microphones

Electret microphones are small and effective at detecting high-frequency sound. They are used in millions of computers and electronic devices around the globe. While relatively cheap, their only drawback is the lack of bass they provide. In addition, you can use carbon microphones (which are less common) in applications in which sound quality is not an issue.

Vibration

Ceramic Piezoelectric Sensor or Accelerometer

Vibration or acceleration is most commonly measured using a ceramic piezoelectric sensor or accelerometer.

Vibration Sensors	Natural Frequency	Number of	Damping Coefficient	Scale Factor	Comparison
Ceramic Piezoelectric (accelerometer)	>5 kHz	Up to 3	Small	Requires high output	• Used in vibration and shock measurements
Linear Variable Differential Transformer (LVDT)	<80 Hz	Up to 3	Medium	Varies	• Limited to steady-state acceleration or low-frequency vibration measurement
Proximity Probe	<30 Hz	Up to 3	Medium	Varies	• Limited to steady-state acceleration or low-frequency vibration measurement • Spring mass attached to the potentiometer wiper
Variable Reluctance	<100 Hz	Up to 3	Medium	Varies	• Output exists only when mass is in motion • Used in shock studies and oil exploration

TABLE 4

Common Vibration Sensors

Three major factors differentiate vibration sensors: The natural frequency, the damping coefficient, and a scale factor. Scale factor relates the output to an acceleration input and is linked to sensitivity. Together, the natural frequency and damping coefficient determine vibration-sensor accuracy. In a system consisting of a spring and attached mass, if you were to pull the mass back away from equilibrium and release the mass, the mass would vibrate forward (past the equilibrium) and backward until it came to rest. The friction that brings the mass to rest is defined by the damping coefficient, and the rate at which the mass vibrates forward and backward is its natural frequency.

Ceramic piezoelectric vibration sensors are the most commonly used sensors because they are the most versatile. You can use these vibration sensors for shock measurements (explosions and failure tests), high-frequency measurements, and slower, low-frequency vibration measurements. They offer a higher than average natural frequency. However, this sensor has outputs typically in the millivolt range and requires a high-input impedance, low-noise detector to interpret voltages from its piezoelectric crystal.

Proximity Probes and LVDTs

Proximity probes and LVDTs are similar: Both are limited to steady-state acceleration or low-frequency vibration measurement; however, the LVDT vibration sensor has a slightly higher natural frequency, meaning that it can handle/detect more vibration. The proximity probe is simply a spring mass attached to a potentiometer wiper.

Variable Reluctance Vibration Sensor

A variable reluctance vibration sensor uses permanent magnets and movement through coils to measure motion and vibration. This special vibration sensor registers output only when the mass it is measuring is in motion, which makes it particularly useful in earthquake shock studies and oil exploration to pick up vibrations reflected from underground rock strata.

Position and Displacement

Driving factors when selecting a position sensor are excitation, filtering, environment, and whether a line of sight or a direct, physical connection is required to measure distance. There is not one universally preferred sensor type, as with pressure or force. Because sensors have measured position for a long time, both preferences and applications affect this decision.

Position Sensor	Price	Environment	Accuracy	Sensitivity	Comparison
Hall Effect Sensor	Low	Standard	On or off	On or off	• Only certain that target is nearby when depressing sensor
Optical Encoders: Linear and Rotary	Varies	Standard	Varies	High	• Accuracy determined by number of counts per revolution
Potentiometers	Low	Standard	High	High	• Required to be physically attached to moving target
Linear and Rotary Variable Differential Transformers (LVDT) or (RVDT)	High	Known for tolerance of dirty industrial environments and precision	High	High	• Handles a high degree of power • Requires signal conditioning • RVDTs typically operate over any angular range of ±30 to 70 °C
Eddy-Current Proximity Probe	Medium	• Noncontacting • Tolerance of dirty environments • Not sensitive to material between sensor and target	Medium	Varies	• Not good where high resolution is required • Not good for use when a large gap exists between sensor and target (optical and laser sensors are better) • Good when mounted on a reasonably stationary mechanical structure to measure nearby moving machinery
Reflective Light Proximity Sensor	Varies	Standard	Varies	High	• Line of sight to target required for measurement • Good for use when large gap exists between sensor and target • Accuracy determined by quality of sensor

TABLE 5
Common Position Sensors

Hall Effect Sensors

With Hall effect sensors, the presence of an object is determined when that object depresses a button. It is either "on," and the object is touching the button, or "off," and the target could be anywhere. Hall effect sensors have been used in keyboards and even in robot boxing battle competitions to determine when a blow was delivered. These sensors provides no scale as to how far away an object is from the sensor when the button is "off," but they are effective for applications that do not require highly detailed position information.

Potentiometers

Potentiometers use a sliding contact to create an adjustable voltage divider that measures position. While potentiometers provide a slight drag to the system to which they are physically connected, potentiometers are cheap compared to other position sensors and can offer great accuracy.

Optical Encoders

Another common position sensor is the optical encoder, which can be either linear or rotary. These devices determine speed, direction, and position with fast, high accuracy. As the name suggests, optical encoders use light to determine position. A series of striped bars divide the distance to be measured by counts. The more counts, the higher the accuracy. Some rotary optical encoders with up to 30,000 counts offer tremendous accuracy. Also, because of their fast response time, they are ideal for many motion-control applications.

Sensors with physical components that attach to a system, such as potentiometers, add a small amount of resistance to the movement of the system's parts. Encoders hardly produce any friction when they move and are very lightweight. However, several important items add to their cost: They require seals to operate within a harsh or dusty environment, and in high-accuracy applications, they need their own bearings to avoid misalignment when incorporated into products.

LVDTs

LVDTs and their rotary counterparts (RVDTs) use magnetic induction to determine position. They are both effective for industrial and aerospace applications because of their robustness. Both require signal conditioning, which can add to cost. Also, these sensors must be accurately aligned inside heavy, expensive packaging and contain wound coils that are expensive to manufacture. In addition to their cost, they are known for their high precision.

Eddy-Current Sensors

Eddy-current sensors are moderately priced and use magnetic fields to determine position. They are used less in applications that require highly detailed positioning information or where large gaps exist between the sensor and the target. These sensors are better used on assembly lines when mounted on a reasonably stationary mechanical structure to measure nearby moving machinery or products. For more precise positioning information, use a light proximity sensor instead.

Reflective Light Proximity Sensors

Reflective light proximity sensors use a beam's travel time to and from a reflective target to determine distance. They have a quick response time and are excellent for applications in which large gaps exist between the sensor and target. These sensors require line of sight, and accuracy and quality directly affect their price.

Pressure

High or low pressure is relative—like with heat. It can be "hot" in a room, but the temperature in that room is nothing compared to the temperature on the surface of the sun. With pressure, the comparison makes the measurement.

Pressure Relative Measurement Types	Tire Example	Comparison
Absolute	Absolute pressure = standard atmospheric pressure + gauge pressure	Relative to 0 Pa, the pressure in a vacuum
Gauge	Reading from tire pressure gauge	Relative to local atmospheric pressure
Vacuum	Typically negative value when relative to local atmospheric pressure Flat tire = 0 kPa on vacuum gauge	Relative to either absolute vacuum (0 Pa) or local atmospheric pressure
Differential	Differential pressure = pressure difference between two different tires	Relative to another pressurized container
Sealed	Sealed pressure = gauge pressure + difference between local atmospheric pressure and sea level pressure	Relative to sea level pressure

TABLE 6

Relative Pressure Measurements

There are five common pressure measurement types: Absolute, gauge, vacuum, differential, and sealed. Consider the following example of measuring the pressure within a tire, and note how each major type is relative to a different reference pressure.

- An absolute pressure measurement includes the standard pressure from the weight of the atmosphere (101.325 kPa) and the additional pressure within the tire. The typical tire pressure is 34 PSI, or about 234 kPa. The absolute pressure is 234 kPa plus 101.325 kPa, or 331.325 kPa.

- A gauge pressure measurement is relative to the local atmospheric pressure and is equal to 234 kPa or 34 PSI.

- Vacuum pressure is relative to either an absolute vacuum or local atmospheric pressure. A flat tire could have the same pressure as the local atmosphere or 0 kPa (relative to atmospheric pressure). This same vacuum pressure measurement could equal 234 kPa (relative to an absolute vacuum).

- Differential pressure is just the difference between any two pressure levels. In the tire example, this means the difference in pressure between two tires. It also could mean the difference between atmospheric pressure and the pressure inside a single tire.

- Sealed pressure measurements are differential pressure measurements taken with a known comparison pressure. Typically this pressure is sea level, but it could be any pressure, depending on the application.

Each of these measurement types could alter your pressure values, so you need to know which type of measurement your sensors are acquiring.

Bridge-based (strain gages), or piezoresistive, sensors are the most commonly used pressure sensors because of their simple construction and durability. These characteristics lower cost and make them ideal for higher channel systems.

These common pressure sensors can be either conditioned or nonconditioned. Typically, conditioned sensors are more expensive because they contain components for filtering and signal amplification, as well as excitation leads and regular measurement circuitry. If you are working with nonconditioned pressure bridge-based sensors, your hardware needs signal conditioning.

Check the sensor's documentation so that you know whether you need additional components for amplification or filtering.

Force

Load Cell Sensors	Price	Weight Range	Accuracy	Sensitivity	Comparison
Beam Style	Low	10–5k lb	High	Medium	• Used with tanks and platform scales • Strain gages are exposed and require protection
S Beam	Low	10–5k lb	High	Medium	• Used with tanks, platform scales • Better sealing and protection than bending beam
Canister	Medium	Up to 500k lb	Medium	High	• Used for truck, tank, and hopper scales • Handles load movements • No horizontal load protection
Pancake/Low Profile	Low	5–500k lb	Medium	Medium	• All stainless steel • Used with tanks, bins, and scales • No load movement allowed
Button and Washer	Low	Either 0–50k lb or 0–200 lb typically	Low	Medium	• Loads must be centered • No load movement allowed

TABLE 7

Common Load Cell Sensors

At one time, mechanical lever scales primarily measured force. Today, strain-gage-based load cells are the most common because they do not require as much calibration and maintenance as scales do.

Load cells can be either conditioned or nonconditioned. Typically, conditioned sensors are more expensive because they contain components for filtering, signal amplification, and excitation leads, as well as regular measurement circuitry. If you are working with nonconditioned bridge-based sensors, your hardware needs signal conditioning. Check the sensor's documentation so that you know whether you need additional components for amplification or filtering.

Beam-style load cells are useful when a linear force is expected, and they are typically used to weigh both small and large items (10 lb up to 5k lb). They have an average sensitivity but are highly accurate. These load cells offer simple construction and low cost.

The S beam load cell is similar to the beam style except for its design. Because of this design difference (the load cell's characteristic S shape), the sensor is effective for high side-load rejection and measuring the weight of a load that is not centered. This low-cost load cell's design is also simple.

The canister load cell can handle larger loads than both S- and beam-style load cells. It also can handle load movement easily and is highly sensitive; however, the sensor requires horizontal load protection.

Pancake or low-profile load cells are designed in such a way that they require absolutely no movement to achieve an accurate reading. If your application has time constraints or requires quick measurements, consider using a canister load cell instead.

Button and washer load cells typically measure the weight of smaller objects (up to 200 lb). Like pancake or low-profile load cells, the object being weighed must not be moving to obtain an accurate measurement. The load also must be centered on what is usually a small scale. The benefit to these load cells is that they are inexpensive.

Read more about common sensor types and terminology

Download the Engineer's Guide to Accurate Sensor Measurements

How to Choose the Right DAQ Device: Specifications

Overview

With so many DAQ devices to choose from, it can be difficult to select the right one for your application. By understanding the signal you are trying to capture, you can build a list of requirements that ensures that the DAQ system you select has the accuracy and precision you need.

This chapter outlines five questions that you should ask before selecting your hardware, to ensure the DAQ system you consider can meet your sampling and accuracy needs:

What types of signals do I need to measure or generate?

While some DAQ devices can perform a single function, such as measuring and generating analog voltage signals, others are purely digital, while others perform multiple functions. A data acquisition application typically relies on a multitude of signals, so it is important to understand which functionality you need for your application.

DAQ Device Functions

- Analog inputs measure analog signals.
- Analog outputs generate analog signals.
- Digital inputs/outputs measure and generate digital signals.
- Counter/timers count digital events or generate digital pulses/signals.

A DAQ device that performs multiple functions typically is called a multifunction I/O device. Both DAQ devices that perform a single function and multifunction I/O devices have a fixed number of channels. Your application dictates which device is the right fit; however, it is good practice to consider if you might need to scale the system in the future. If so, it's probably cost-efficient to select a device that offers more channels than you currently require. This is especially true if you are using the DAQ device for a limited amount of time on one application and plan to move on to another application, expecting to use the same device.

Another option is a modular platform that you can customize to your exact requirements. A modular system consists of a chassis to control timing and synchronization along with a variety of I/O modules. An advantage of a modular system is that you can select different modules that have unique purposes for more configurations. With this option, you can find modules that perform one function more accurately than a multifunction device. Another advantage is that you can select the number of slots for your chassis. While a chassis has a fixed number of slots, you can purchase a chassis that has more slots than you need now to give you the ability to expand in the future.

Do I need signal conditioning?

A typical general-purpose DAQ device can measure or generate +/-5 V or +/-10 V. Some sensors generate signals too difficult or dangerous to measure directly with this type of DAQ device. Most sensors require signal conditioning (such as amplification or filtering) before a DAQ device can effectively and accurately measure the signal.

For example, thermocouples output signals in the mV range that require amplification to optimize the limits of the analog-to-digital converters (ADCs). Additionally, thermocouple measurements benefit from lowpass filtering to remove high-frequency noise. Signal conditioning provides a distinct advantage over a DAQ device alone because it enhances both performance and measurement accuracy.

Table 8 summarizes common signal conditioning for different types of sensors and measurements.

	Amplification	Attenuation	Isolation	Filtering	Excitation	Linearization	CJC	Bridge Completion
Thermocouple	X			X		X	X	
Thermistor	X			X	X	X		
RTD	X			X	X	X		
Strain Gage	X			X	X	X		X
Load, Pressure,Torque (mV/V, 4–20 mA)	X			X	X	X		
Accelerometer	X			X	X	X		
Microphone	X			X	X	X		
Proximity Probe	X			X	X	X		
LVDT/RVDT	X			X	X	X		
High Voltage		X	X					

TABLE 8

Sensors and Measurement Signal Conditioning

If your sensor is listed in Table 8, you should be conditioning your sensor signals. You can build external signal conditioning or use a DAQ device with built-in signal conditioning. Many devices include direct connectivity to common sensors for convenient integration. For a more in-depth guide on sensor signal conditioning, download the **Engineer's Guide to Accurate Sensor Measurements**.

How fast do I need to acquire or generate signal samples?

One of the most important specifications of a DAQ device is the sampling rate—the speed at which the DAQ device's ADC takes signal samples. Typical sampling rates are either hardware- or software-timed and can reach rates of up to 14 MS/s. The sampling rate for your application depends on the maximum frequency component of the signal that you are trying to measure or generate.

The Nyquist theorem states that you can accurately reconstruct a signal by sampling 2X the highest frequency component of interest. However, in practice, you should sample at least 10X the maximum frequency to represent your signal's shape.

Choosing a DAQ device with a sample rate at least 10X the frequency of your signal will 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. According to the Nyquist theorem, you must sample at 2 kHz at least, but you should ideally sample at 10 kHz to measure or generate a more accurate signal representation. Figure 1 compares a 1 kHz sine wave measured at 2 kHz and 10 kHz.

Once you know the maximum frequency component of the signal that you want to measure or generate, you can choose a DAQ device with the appropriate sampling rate.
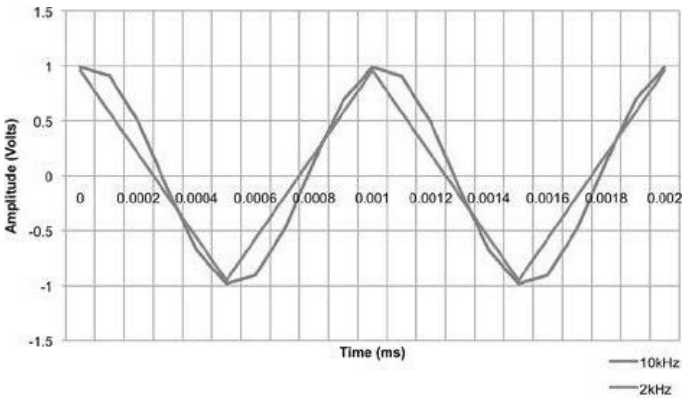

FIGURE 1

10 kHz versus 2 kHz Representation of a 1 kHz Sine Wave

What is the smallest change in the signal that I need to detect?

The smallest detectable change in the signal determines your DAQ device's resolution. Resolution refers to the number of binary levels an ADC can use to represent a signal. To illustrate, imagine how a sine wave would be represented if it were passed through an ADC with different resolutions. Figure 2 compares a 3-bit ADC and a 16-bit ADC. A 3-bit ADC can represent eight (2^3) discrete voltage levels. A 16-bit ADC can represent 65,536 (2^{16}) discrete voltage levels. The representation of the sine wave with a 3-bit resolution looks more like a step function than a sine wave, whereas the 16-bit ADC provides a clean-looking sine wave.

Typical DAQ devices have voltage ranges of ±5 V or ±10 V. Represented voltage levels are distributed evenly across a selected range to take advantage of the full resolution. For example, a DAQ device with a ±10 V range and 12 bits of resolution (2^{12} or 4,096 evenly distributed levels) can detect a 5 mV change, whereas a device with 16 bits of resolution (2^{16} or 65,536 evenly distributed levels) can detect a 300 µV change.

Many application requirements are met with devices that have 12, 16, or 18 bits of resolution. However, if you are measuring sensors with small and large voltage ranges, you likely can benefit from the dynamic data range available with 24-bit devices. The voltage range and resolution required for your application are primary factors in selecting the right device.
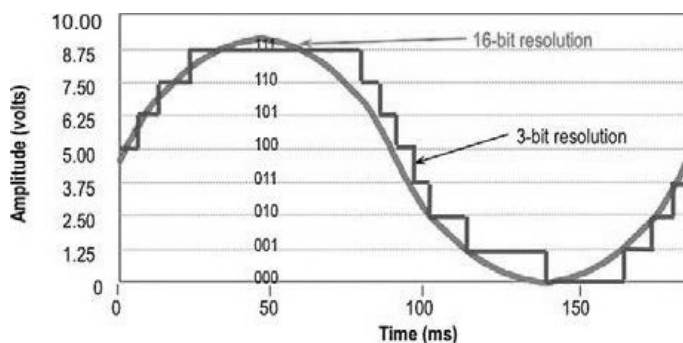

FIGURE 2
Sine Wave 16-Bit Resolution versus 3-Bit Resolution

How much measurement error does my application allow?

Accuracy is defined as a measure of the capability of an instrument to faithfully indicate the value of a measured signal. This term is not related to resolution; however, accuracy can never be better than the resolution of the instrument. How you specify the accuracy of your measurement depends on the type of measurement device. An ideal instrument would always measure the true value with 100 percent certainty, but in the real world, instruments report a value with an uncertainty specified by the manufacturer. The uncertainty can depend on many factors, such as system noise, gain error, offset error, and nonlinearity.

A common specification for a manufacturer's uncertainty is absolute accuracy. This specification provides the worst-case error of a DAQ device at a specific range. An example calculation for an NI multifunction device's absolute accuracy is:

ABSOLUTE ACCURACY = ([READING * GAIN ERROR] + [VOLTAGE RANGE * OFFSET ERROR] + NOISE UNCERTAINTY)

ABSOLUTE ACCURACY = 2.2 MV

Note that an instrument's accuracy depends not only on the instrument, but on the type of signal it's measuring. If the signal being measured is noisy, the measurement's accuracy is adversely affected. There are many DAQ devices with varying degrees of accuracy and price points. Some devices provide self-calibration, isolation, and other circuitry to improve accuracy. While a basic DAQ device may provide an absolute accuracy of more than 100 mV, a higher-performance device with such features may have an absolute accuracy of around 1 mV.

Once you understand your accuracy requirements, you can choose a DAQ device with an absolute accuracy that meets your application needs.

Download the Engineer's Guide to the Digitization of Analog Signals

Compare different DAQ hardware products for your application

How to Choose the Right DAQ Device: Bus and Form Factor

Overview

Choosing a DAQ device involves more than selecting the right specifications for quality measurements. Your environment and equipment should shape your choice of DAQ device. Which form factor and ruggedness you choose determines where you can use your system—whether in a controlled lab or the unpredictable field. The bus you select determines not only your DAQ system's throughput and latency, but also your measurement device's portability.

This chapter examines the most common PC bus options and outlines the technical considerations to keep in mind when choosing the right bus and form factor for your measurement application:

QUICK REFERENCES

How much data will I be streaming across this bus?

All PC buses have a limit to the amount of data that can be transferred over a certain period of time. Known as the bus bandwidth, it's 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 that 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 sampling rate and resolution based on how fast your signal is changing. You can calculate the minimum required bandwidth by taking the number of bytes per sample (rounded up to the next byte), multiplied by the sampling speed, and then multiplied by the number of channels.

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

$$\frac{2 \text{ BYTES}}{S} * \frac{4 \text{ MS}}{SEC} * 4 \text{ CHANNELS} = 32 \text{ MB/S}$$

Your bus bandwidth needs to be able to support the speed at which data is being acquired, and it is important to note that the actual system bandwidth will be 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 high-resolution data from several channels, bandwidth may be the most important consideration when choosing your DAQ bus.

What are my single-point I/O requirements?

Applications that require single-point reads and writes often depend on I/O values being updated immediately and consistently. Based on how bus architectures are implemented in both hardware and software, single-point I/O requirements could be the determining factor for the bus that you choose. For example, in a proportional integral derivative (PID) control system, single-point I/O is hugely important, and bus latency can directly impact the maximum control loop speed.

Bus latency is I/O responsiveness. It is the time delay between when a driver software function is called and the actual hardware value of the I/O is updated. Depending on the bus you choose, this delay could range from less than a microsecond to a few milliseconds.

Another important factor in single-point I/O applications is determinism, which is a measure of how consistently I/O can execute on time. Determinism is important for control applications because it directly impacts control loop reliability, and many control algorithms are designed with the expectation that the control loop always executes at a constant rate. Any deviation from the expected rate makes the overall control system less effective and less reliable. Therefore, when implementing closed-loop control applications, you should avoid buses such as wireless, Ethernet, or USB that are high-latency with poor determinism. In general, internal buses such as PXI Express or the FPGA in an NI CompactRIO chassis are better for low-latency single-point I/O applications than external buses such as USB or wireless.

Communication bus software implementation plays a large role in bus latency and determinism. Buses and software drivers that support an RTOS provide the best determinism, and therefore give you the highest performance.

How portable should this system be?

Portable computing brings new ways to innovate with PC-based data acquisition. Portability could easily be the primary reason to choose one bus over another. In-vehicle DAQ applications, for example, benefit from compact, easily transported hardware.

External buses such as USB and Ethernet are particularly good for portable DAQ systems because of their quick hardware installation and laptop compatibility. Bus-powered USB devices which are powered via the USB port offer additional convenience because they do not require a separate power supply.

How far will my measurements be from my computer?

The distance between measurements you need and where the computer is located can drastically vary from application to application. To achieve the best signal integrity and measurement accuracy, you should place your DAQ hardware as close to the signal source as possible. This can be a challenge for large distributed measurements like those used for structural health monitoring or environmental monitoring.

Running long cables across a bridge or factory floor is costly and can result in noisy signals. One solution to this problem is to use a portable computing platform to move the entire system closer to the signal source. With wireless technology, the physical connection between the computer and the measurement hardware is removed altogether, and you can take distributed measurements and send the data back to a central location.

How rugged will this system need to be?

More tests than ever are being taken outside of pristine lab environments—such as in the field, where humidity, shock, and vibration are concerns, or in test cells, where extreme temperatures and spray-downs are commonplace.

Consider the environments into which you may need to take your DAQ system, and ensure that your DAQ system can handle it. Look for operating temperature and shock and vibration ratings: For extreme environments, consider the ingress protection (IP) rating, which tells you a device's dust- and water-resistance level. For example, NI FieldDAQ™ devices have an IP rating of IP65 and IP67, which means that they are dust-tight, water-jet resistant, and water-submersion resistant.

Do I need to synchronize multiple devices?

Many measurement systems have complex synchronization needs, whether that's synchronizing hundreds of input channels or multiple types of instruments. A stimulus-response system, for example, might require the output channels to share the same sample clocks and start triggers as the input channels to correlate the I/O and better analyze the results.

DAQ devices on different buses provide different ways to accomplish this. Synchronization techniques are often categorized into signal-based or time-based.

Signal-based techniques offer the tightest synchronization between devices, but can introduce cabling complexity. Timing signals—including a clock signal and a trigger pulse—are shared directly, with devices connected by a physical cable. These cables can be external (for synchronizing two USB DAQ devices together, for example), or internal (for example, the synchronization cabling built into PXI chassis for synchronizing modules together).

The PXI platform, which includes PXI and PXI Express, offers the tightest synchronization between multiple devices. This open standard was designed specifically for high-performance synchronization and triggering, offering several ways to synchronize I/O modules within the same chassis as well as synchronizing multiple chassis.

Time-based synchronization techniques simplify synchronization between devices, providing accurate synchronization without additional hardware or cabling. In a time-based system, multiple devices on a network such as Ethernet individually set their clocks to a common time source.

An increasingly popular time-based synchronization technology is time-sensitive networking (TSN). TSN is an open-source update to the IEEE Ethernet standard designed to address measurement and control system needs. Some DAQ devices support a subset of the TSN standard denoted as 802.1AS, which offers tight synchronization between devices using a standard Ethernet cable. Leveraging 802.1AS, these devices support synchronization within <1 µs over a distance of 100 meters. Devices that support TSN, or a subset of the standard, include certain NI CompactDAQ chassis, NI FieldDAQ, and certain NI CompactRIO devices.

Common Bus Selection Guide

Based on the five questions previously outlined, Table 9 shows a selection guide for the most common DAQ buses.

Bus	Waveform Streaming	Single-Point I/O	Portability	Distributed Measurements
PCI	132 MB/s (shared)	Best	Good	Good
PCI Express	250 MB/s (per lane)	Best	Good	Good
PXI	132 MB/s (shared)	Best	Better	Better
PXI Express	250 MB/s (per lane)	Best	Better	Better
USB	60 MB/s	Better	Best	Good
Ethernet 2.0	125 MB/s (shared)	Good	Best	Best

TABLE 9
Bus-Selection Guide Based on Application Requirements with NI Products

DAQ Buses for PC-Based Systems

While you can choose from many buses and form factors, this section focuses on the most common buses for a PC-based system:

- PCI and PCI Express
- USB
- PXI and PXI Express
- Ethernet

In Figure 3, all of these buses are represented in this collection of NI DAQ products, from internal plug-in options to hot-swappable external buses.
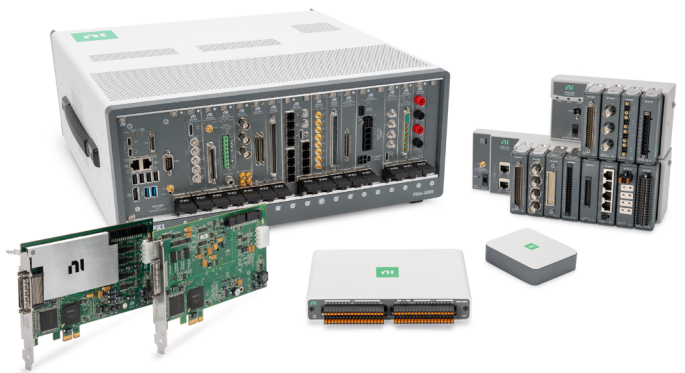

FIGURE 3

These NI DAQ products use several buses and form factors, including PXIe, PCIe, USB, and Ethernet.

PCI and PCI Express

The PCI bus is an older, yet still commonly used, internal computer bus. With a shared bandwidth of 132 MB/s, PCI offers data streaming and data transfer for single-point control applications.

PCI Express is an evolution of PCI, offering a new level of PC performance. The single biggest benefit of PCI Express architecture is the dedicated bus bandwidth provided by independent data-transfer lines. PCI Express uses independent data lanes that are each capable of data transfer of up to 250 MB/s, meaning that a single device can reach several GB/s of bandwidth.

There are several DAQ devices to choose from in the PCI/PCIe form factor, with multiple choices of resolutions, sample rates, and signal conditioning.

Compare NI PCI and PCI Express DAQ devices

FIGURE 4
PCI Express X Series Multifunction DAQ

USB

USB delivers an inexpensive and easy-to-use connection between DAQ devices and PCs. USB 2.0 has a maximum theoretical bandwidth of 60 MB/s, which is shared among all devices connected to a single USB controller.

USB devices are inherently latent and nondeterministic. This means that single-point data transfers may not happen exactly when expected, and therefore USB is not recommended for closed-loop control applications such as PID.

On the other hand, the USB bus has several characteristics that make it easier to use than internal PC buses. USB devices are hot-swappable and plug-and-play, meaning that the PC will detect a newly connected device and, with the right driver, automatically install it.

Compare NI USB DAQ devices

FIGURE 5
USB DAQ adds data acquisition to any computer with a USB port.

PXI

PXI bridges the gap between desktop PC systems, high-end VXI, and GPIB systems. The PXI Systems Alliance, with more than 200 members, maintains this open standard, and in 2006, passed the PXI Express specification to deliver PCI Express data-transfer technology to the PXI platform.

Based on CompactPCI, PXI incorporates instrumentation extensions and more rigid system-level specifications to ensure an open yet high-performance measurement and automation specification. Benefits of PXI-based DAQ systems include rugged packaging that can withstand often-harsh industrial conditions. Also, PXI systems offer a modular architecture, which means that you can fit several devices in the same space as a single stand-alone instrument, and you can expand your system far beyond the capacity of a desktop computer with a PCI bus. One of the most important benefits PXI offers is its integrated timing and triggering features. Without any external connections, you can synchronize multiple devices using the internal buses resident on the PXI chassis backplane.

Compare NI PXI DAQ options

Ethernet

Ethernet is the backbone of almost every corporate network in the world and is, therefore, widely available. As a DAQ bus, Ethernet is ideal for taking portable or distributed measurements at distances beyond the 5 m length of a USB cable. A single Ethernet cable can extend 100 m before needing a hub, switch, or repeater. This distance, combined with a large install base of networks in labs, offices, and manufacturing facilities, makes Ethernet an ideal choice for distributing measurements to remote locations.

Select Ethernet DAQ devices use the TSN (802.1AS) IEEE standard for easy time-based synchronization between devices using a regular Ethernet cable. Some CompactDAQ and CompactRIO devices and all FieldDAQ devices use this standard for easy multidevice synchronization.

Compare NI Ethernet DAQ options

Learn more about DAQ buses and form factors

Discover more about NI DAQ devices

FIGURE 6

The PXI platform is composed of chassis, controllers, and I/O modules.

FIGURE 7

Supporting 100 m per segment and the ability to use existing network infrastructure, Ethernet data acquisition can extend the reach of your measurement system.

How to Choose the Right Computer

Overview

Once you have chosen your DAQ device, it's important to select the right computer for your application. The computer can be the most crucial part of your data acquisition system. It provides flexibility over traditional boxed systems by housing the DAQ device, running the software to control the device, analyzing the measurements, and saving the results.

This chapter explores the questions you should ask to choose the right computer for your application:

QUICK REFERENCE

How much processing power do I need?

Nearly every computer has three main components that affect data-management capabilities: The processor, the RAM, and the hard drive.

The processor is the part of the computer that interprets and executes instructions—think of it as the brain. Processors in most new computers have multiple cores, meaning that the computer can use multiple independent processors, or "cores," to read and execute program instructions.

Computer processing power also consists of the RAM, the hard drive size, and the processor speed. With more RAM, you improve speed and can run more applications simultaneously. More hard drive space gives you the ability to store more data.

Finally, faster processors translate to faster application operation. In general, faster is better, but processor speeds across brands may not be equivalent. If you need to analyze or save the data you acquired from your application, processing power is a key feature to consider for your computer.

Do I need my computer to be portable?

Portability is imperative if you move frequently between applications or locations. For example, a portable computer is essential for taking measurements in the field and returning to the lab to analyze the data. Portability also is important if you need to monitor applications in different locations.

Two considerations when assessing portability are product size and weight (acknowledging that lighter PCs can reduce performance). If you need a powerhouse of a computer, but need to take remote data, consider building a distributed DAQ system, extending your measurement system into the field over Ethernet while keeping your computer safe in a control room or lab.

How rugged does the computer need to be?

Ruggedness can be crucial if you are monitoring your application in an extreme environment. Operating conditions—for example, operating and storage temperature, relative humidity, and maximum operating and storage altitude—determine computer ruggedness. Standard off-the-shelf PCs are not designed to withstand industrial-environment conditions.

Typical specifications are 50 °F to 95 °F (operating temperature), -13 °F to 113 °F (storage temperature), 10,000 feet (operating altitude), and 15,000 feet (storage altitude); any computers featuring specifications greater than these are considered rugged.

When designing your system, consider your environment. If heavy vibration or temperature swings could result in critical data loss, it may be worth investing in a rugged or industrial PC.

Do I need my computer to be modular?

Computer modularity is a factor if you are considering future applications or working on multiple applications. Modularity describes the degree to which you can separate and recombine a system's components. You can modify and adapt the system to meet your current needs and plan future expansion, as well as upgrade individual components, without having to buy a whole new system.

For example, with a modular tower PC featuring PCIe slots, you can install a new hard drive if you need more space or install a DAQ device with a faster analog-to-digital converter if you need faster sampling. Laptops and tablets provide portability, but they are more integrated, which makes them harder to upgrade. Modularity can be an important feature if you need to adapt your current application to future demands.

Do I need an RTOS?

The OS is an important feature to consider when choosing a DAQ computer. By far, the most common general-purpose OS is Windows, but DAQ and control applications can require a more specialized OS.

An RTOS can operate deterministically, so that applications execute according to precise timing requirements. An RTOS is deterministic because it does not determine which process happens when; rather, you define process order and timing, giving you more control over your application and the ability to execute at faster rates than with a nondeterministic OS. This is especially relevant in control applications in which you need to prioritize critical tasks above others.

Learn more about RTOSs

Computer Selection Guide

Based on the previous six questions, Table 10 shows a selection guide for the most common types of computers.

	PXI System	Desktop	Industrial Controller, NI CompactRIO	Laptop	Tablet
Processing Power	Best	Best	Better	Better	Good
OS Compatibility	Best	Best	Better	Best	Good
Modularity	Best	Better	Better	Better	Good
Ruggedness	Better	Better	Best	Good	Good
Portability	Better	Good	Best	Best	Best

TABLE 10
Computer Selection Features

How to Choose the Right Driver Software

Overview

While you might be tempted to overlook driver software when selecting a DAQ device, the driver behind your DAQ device can be one of the most important factors in development time and device performance.

The driver handles the communication layer between hardware devices and application software, giving you access to both high-level functions for quick measurements and low-level control for fine-tuning complex tasks.

This chapter answers the questions you need to keep in mind when evaluating a DAQ device's driver software:

Is the DAQ driver compatible with my OS?

You can choose from OSs including Windows, macOS, and Linux, which offer different advantages for different tasks, operations, and deployments. Each of these OSs also may have different versions, distributions, or designs for specific processors. For example, Windows offers versions for 32-bit and 64-bit processors, and open-source Linux OSs include hundreds of varieties. Each type, release, and version of an OS functions differently and may or may not be cross-compatible.

As a result, DAQ drivers generally do not support every single OS type and version. Most DAQ drivers work with Windows OS releases, as they are the most common. NI's DAQ driver, NI-DAQmx, supports most Windows and several Linux variants.

If you use an alternative OS, remember to confirm whether the DAQ device driver supports it before choosing a DAQ device. You generally can find the OS and version support in the driver readme files.

How well does the driver integrate with my application software?

There are varying degrees of driver integration with application software. At the core of every driver is a library (often a DLL). This library manages the communication with DAQ hardware. Normally, the library comes with documentation and wrappers for various programming languages. These wrappers are thin layers of code that translate the library's functions into a compatible interface for a particular programming language.

In a perfect scenario, the provided driver natively integrates with your application software. In this case, the driver is rewritten for the native language. This provides better performance and a more seamless experience because functions and documentation are directly built into the application software.

In some cases, a wrapper may not be provided for your preferred language, or even at all, so you must manually write your own wrapper to interface with your application software. When evaluating a DAQ system, check to see what languages the driver can support, and if possible, access the documentation for that language. Ideally, you can get full driver functionality from the driver in a wide variety of application software.

What documentation comes with the driver?

Drivers feature many forms of documentation, including user manuals, functions references, release notes, known issues, and example code. Having to navigate through poor, incomplete, or muddled documentation wastes time. When a driver's programming interface is poorly documented, you can spend an unnecessary and frustrating amount of time running functionality trial-and-error tests. Although trial and error can be a great way to learn functions and syntax, you need to be able to refer to the manual. Therefore, having well-organized, thorough documentation is extremely valuable.

The best driver software documentation is complete, easy to navigate, and simple to follow. Ideally, it offers example code specific to your preferred programming languages and provides detailed and useful error messages. By evaluating the driver software's documentation ahead of time, you can save yourself potential headaches in the future.

Does the driver include any setup or diagnostic utilities?

In addition to documentation, setup and diagnostic utilities can help you get your application up and running quickly and diagnose problems.

- With test panels, you can test hardware functionality at the most basic level before designing the end application. You can generate and measure raw signals and troubleshoot the DAQ hardware independently of other software and programming factors that could insert an extra level of uncertainty.

- Calibration utilities walk you through the steps to self-calibrate your device to ensure it measures accurately.

- Sensor scaling wizards help you easily map raw voltage values to engineering units without having to program the math yourself.

- Some drivers even include complete configuration wizards that encapsulate all of these utilities, walk you through setting up your measurement task, and help you take your first measurement in your application software.

Overall, setup and diagnostic utilities are very useful when getting started with your DAQ device or diagnosing problems. Not all DAQ drivers include these utilities, though.

Is the driver scalable to other DAQ devices?

You might not be able to anticipate which changes and expansions your current DAQ system may need in the future. You may need to upgrade the device to higher-performance specifications or incorporate additional measurements. Some DAQ drivers are designed for a single device, and others are designed to work with a wide range of devices.

Single-device drivers typically are more lightweight than drivers that work with a wide range of devices. While these drivers initially do the job, adding a new device or replacing an existing one could require significant programming to integrate the corresponding new driver. The driver's programming interface might be structured differently, requiring significant code changes.

On the other hand, drivers that support a wide range of devices are more easily scaled to additional functionality and new devices. The programming interface is consistent among all devices, so adding a new device is essentially a drop-in replacement and requires little to no changes to your code. These drivers also may support other features that make synchronizing and combining measurements from multiple devices easier.

Read more about NI-DAQmx Driver Software for DAQ Applications

Explore the NI-DAQmx Driver Manual for compatibility information

How to Choose the Right DAQ Software

Overview

Software lies at the core of modern DAQ systems. Most of your time with a DAQ system is spent in the software environment, so it's imperative that you select a software tool that fits the needs of your application today and easily scales as your system matures.

DAQ software ranges from ready-to-run application software to a fully customizable programming language. Take into account your measurement requirements, available development time, and programming expertise to help you choose the right tool for your DAQ system.

Consider these questions before choosing your application software:

QUICK REFERENCE

What is the difference between application software and development environment?

DAQ software tools range from ready-to-run application software (no programming required) to fully customizable development environments. You can use either to build a robust and flexible measurement system, but there are tradeoffs to both.

Application software prioritizes ease of use so that you can acquire and process data with minimal (or no) training. You can use the software to set up hardware, visualize measurement channels, log data, and more, using drop-down menus and prebuilt screens.

Typical application software is workflow-based—NI FlexLogger™ software, for example, is designed around data-logging applications, including all configuration, visualization, and events/alarming you might need to record mixed measurements. You could encounter limitations when you require functionality outside of that workflow (for example, in postprocessing and reporting). In those cases, you may need to develop plug-ins or use another tool such as NI DIAdem.

Development environments prioritize custom functionality so that you can meet virtually any challenge in your measurement system. Development environments are extremely flexible because you can integrate DAQ drivers into the software and develop a custom user interface (UI) and code to perform the exact measurements or test routines that you need.

A development environment such as NI LabVIEW can function as a data logger control system, postprocessor, report generator, and more. The trade-off for development environments is that you need to spend time upfront to learn the programming language and develop the applications yourself. While this may seem like a large time commitment, modern-day development environments provide a variety of tools to help you get started, reducing overall time spent.

The following sections refer to both application software and development environments as two valid options for your measurement system.

How much time do I have to build my measurement system?

When choosing software, it's important to understand your own time constraints. You need time to learn the software, set up your system, and potentially debug your measurement code. When evaluating the time you need to build your measurement system, consider the long-term, too—if you plan to scale this test in the future, the time you invest in training today may yield greater returns as your system grows and changes.

How long will it take to learn the software?

Ready-to-run application software tools are the easiest and fastest to learn because they have abstracted user programming details, typically only requiring a few details for setup. When deciding among ready-to-run software tools for your DAQ system, ensure that the tool has the hardware support, processing capabilities, and analysis libraries necessary to meet your requirements. Also confirm that it offers proper resources—whether user manuals, in-product help information, online communities, or support forums—to help you quickly learn the tool.

Application development environments often take longer to learn, but the majority of that time is spent learning the language used within the environment to program your applications. If you can find an application development environment that uses a language you are familiar with, you can reduce the time required to become a proficient programmer within a new application development environment. Many application development environments can integrate with—and even compile—several different languages within a single framework.

When evaluating application development environments that require you to learn a new language, consider those that direct your focus to your engineering problem, not low-level programming-language details. For example, you might find that text-based languages, such as ANSI C/C++, are often more challenging to learn because of all the complex grammar and syntax rules that you must follow to successfully compile and run the code.

Graphical programming languages, such as the one offered in NI LabVIEW, are often easier to learn because the implementation is more intuitive and visually consistent with the way an engineer thinks.

```
int32 CreateDAQTaskInProject(TaskHandle *taskOut1)
{
    int32       DAQmxerror=DAQmxSucess;
    TaskHandle taskOut;

    DAQmxErrChk (DAQmxCreateTask("",&taskHandle));
    DAQmxErrChk (DAQmxCreateAIVoltageChan(taskHandle,"Dev1/ai0","",
        DAQmx_Val_Cfg_Default,-10.0,10.0,DAQmx_Val_Volts,NULL));
    DAQmxErrChk (DAQmxCfgSampClkTiming(taskHandle,"",10000.0,
        DAQmx_Val_Rising,DAQmx_Val_FiniteSamps,1000));
    DAQmxErrChk (DAQmxStartTask(taskHandle));
    DAQmxErrChk (DAQmxReadAnalogF64(taskHandle,1000,10.0,
        DAQmx_Val_GroupByChannel,data,1000,&read,NULL));
    *taskOut1 = taskOut;
Error:
    return DAQmxError;
}
```

<div align="center">ANSI C Code</div>

<div align="center">LabVIEW Code</div>

FIGURE 8

Sample Code

Does my software have training options to help me get started?

Also consider getting-started resources that come with the application software. These resources can help you get up and running with a new software tool in less time. Here are a few helpful resources for getting started with any software tool:

- Evaluation—A free evaluation gives you the opportunity to test things out for yourself and determine if the tool meets your application needs.
- Online curriculum—Online tutorials, videos, and white papers are valuable when learning basic application software concepts.
- Classroom instruction—An application software class is the perfect way to get up to speed and begin developing your DAQ system. Course price and level of detail depends on the instructional setting. Often, you can find options ranging from free seminars to formal classroom instruction, to instructor-led online courses.
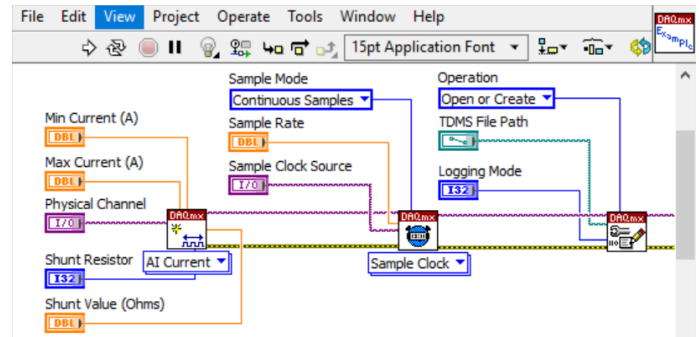
- Shipping examples—Good shipping-example sets have enough code for the most common types of DAQ applications. With these examples, you never need to start from scratch.

You can save time by simply modifying the shipping examples to meet the needs of your system development.

Is there a community to reach out to when I get stuck?

The ecosystem that surrounds software is just as important as the software tool itself. A healthy ecosystem provides a wealth of resources that make it easy for you to learn a new software tool and guide you with feedback as you develop your application. Before you purchase, browse a community's forums and determine how active it is and the kind of information being shared (code, discussions, tips, and tricks). You want a community that is heavy with activity and includes shared information that is closely aligned with the problems you are solving.

Additionally, an application software's ecosystem of users often drives future development. Check to see

if the organization behind an application software is responsive to its community's needs and whether the user base can provide input that guides future software features.

What kind of analysis do I need to perform?

Raw data is not always the most useful way to communicate. Data transformations—removing signal noise, compensating for environmental effects such as temperature and humidity, and calibrating equipment error, for example—help turn raw data into useful data. Most engineering applications aim to produce useful data, and that requires comprehensive signal processing for any analysis tool used in data acquisition. There are two main ways that you can analyze or process data:

- Offline analysis, also known as postprocessing, takes place after you've saved your data to file. Offline analysis is a good fit for an application such as FlexLogger software, which emphasizes logging data to be postprocessed after you run the test. See the Data Management chapter for more details on postprocessing and offline analysis.
- Inline analysis implies that data is analyzed in the same application in which you acquired it. If your application involves monitoring a signal and changing testing variables based on incoming data characteristics, inline analysis is the right choice. By measuring and analyzing certain signal aspects, you can make the application adapt to certain circumstances and incorporate execution parameters—perhaps saving the data to disk in case of alarm or increasing the sampling rate if the incoming values exceed a threshold limit. To perform inline analysis, your application software must have built-in signal analysis functions or the ability to easily integrate external IP.

Most data-analysis tool vendors produce a well-documented listing of their tool functions, which is helpful if you know your specific signal-processing needs. But if you don't know exactly what you need, look for a tool offering functions related to your field or application type.

Proper data-analysis tools contain more than 600 built-in functions. While basic and complex math operations are beneficial, you need functions specific to your area of interest. If your application deals with control, look for proportional integral derivative (PID) control functions. If you have an optical character recognition (OCR) application, make sure your tool contains those functions. Look for software products with an ecosystem of add-ons to extend product functionality, including third-party industry-expert analysis. This way, you reduce time and cost associated with developing these functions yourself.

If you need inline analysis, be sure that your application software includes built-in or expansion capabilities. If your needs involve offline analysis, your application software must be able to save data to a format that your offline analysis package can consume.

What kind of data visualization do I need?

Data visualization—from simply graphing an acquired signal to correlating measurement data with video, sound, or 3D model projection—is common in almost all measurement systems. Selecting the right visualization technique could mean the difference between being able to appropriately derive actionable information from raw data and missing important insight.

As with analysis, visualization can happen inline or offline. The final chapter discusses offline visualization (typically considered report generation).

Inline reporting happens in the DAQ software so that you can follow data trends, view critical system information, and create an engaging UI. For example, you can display acquired data on a monitor so that a technician can see the signal being measured and ensure proper connections.

If you run inline analysis with inline visualization, the monitor may display a filtered version of the same signal. While this architecture gives you "instant feedback," since you can visualize acquired data in

near-real time, it means that your chosen application software must contain the required visualization tools.

For visualization, most engineers require, at a minimum, basic charting and graphing capabilities. Luckily, almost every data visualization tool on the market can make simple charts and graphs, and dedicated visualization tools offer robust additional capabilities that help you learn more from your data.

It's important to consider visualization scalability and customizability. Out-of-the-box application software can include a wide variety of charts and graphs and present multiple plots in a single graph. They might have indicators and visualization techniques, but may not give you complete control over how you show your data. Development environments, on the other hand, offer extensive, in-depth customizability so that you can control every facet of how you visualize your data. Be sure to choose a development environment that makes GUI design easy—some text-based programming languages make visualizing data challenging.

Can I integrate my own custom or legacy IP?

You might have a proprietary analysis algorithm that simply can't be purchased as add-on software.

Or, because application requirements change over time, you've invested time and money creating analysis routines or custom IP in older or alternative tools. In these cases, look for a data analysis package that can incorporate these external analysis routines. Don't reinvent the same functionality in the newer tool when your existing algorithms are validated to work correctly.

Whether you created your analysis routine in another programming language, used a script in an older financial-analysis tool, or inherited some configuration file, confirm with your software vendor that you can incorporate your legacy analysis routine in their data-analysis tool. If you can't do so easily, you could spend precious time recreating your functionality in the new tool. Modern data-analysis tools should be open to using IP created in other environments.

Software Selection Table

Based on the questions and considerations covered above, Table 11 highlights the differences between application software and two types of development environments: Graphical (as in LabVIEW) and text-based (as with Python or C++).

Selection Criteria	Application Software (No Programming)	Development Environment (Graphical Programming)	Development Environment (Text-Based Programming)
Example	FlexLogger	LabVIEW	Python, C, C++
Ease of Use	Best	Better	Good
Time to First Measurement/Test	Best	Better	Good
Customizability	Good	Best	Best
DAQ Driver Integration	Best	Best	Good
Inline Analysis Tools	Good	Best	Better
Inline Visualization Tools	Best	Best	Better
External Hardware Integration	Good	Best	Best
External IP Integration	Good	Best	Best

TABLE 11
DAQ Software Selection Guide

Read about NI FlexLogger software, ready-to-use application software for data logging

Find out how NI LabVIEW, the development environment that accelerates engineering, can help you meet your application and system needs

How to Choose the Right Data Management Software

Overview

You acquire data for a reason: To help you make decisions. While technology is making data retention faster and richer, storing, processing, and sharing it remains a real challenge.

Most DAQ systems collect data to analyze and ultimately present or share it in an exchangeable report. You can choose from a wealth of data management tools, but you should carefully consider the capabilities of your tool of choice to ensure that it doesn't become a bottleneck in your system.

This chapter outlines six questions to consider when choosing a reporting tool for your application:

Can my data management software handle my data?

We all have used spreadsheet programs to manipulate, analyze, and share raw data with graphs and charts. But classic spreadsheet programs are designed for financial analysis. When you're looking for a measurement system data-management tool, consider two major differences: First, the data-storage file format, and second, the volume of data. It's imperative that a reporting tool not only load data from your chosen file format, but handle as much data as you need it to.

File Format

Traditional file types rarely meet all of the requirements you need in a file format. For example, ASCII files are exchangeable, but very large and slow to read and write. Binary files read and write speeds can keep up with high-speed hardware, but these files are difficult to share.

The technical data management streaming (TDMS) file format eliminates these challenges. TDMS files are based on the TDM data model for saving well-organized and documented test and measurement data.

With TDMS-formatted files, you do not have to redesign your application as your DAQ requirements increase. You simply extend the data model to meet your needs. Because it was developed to meet the needs of all engineers, TDMS is easy to use and offers high-speed streaming and exchangeability.

Traditional financial analysis tools use the cell as their fundamental building block. Cells form rows and columns to make up a spreadsheet, an architecture that is ideal for budgets and balance sheets. Simple, single-point DAQ applications—for example, those that collect one single data point an hour over the course of a day—are often easily mapped to this architecture because each individual data point holds more importance when fewer data points are collected. Each data point exists as a cell in a spreadsheet and must be manipulated using this cell-based paradigm.

DAQ applications collecting dozens of data channels at megasample-per-second (MS/s) rates are also commonplace. In these applications, data manipulation and interaction happen on a signal (or channel) as a whole. If you manipulate columns of individual cells, you risk losing signal unity.

Manipulating entire columns at one time is cumbersome. Columns often contain descriptive information, such as a name or unit, in addition to raw numeric data. So when you select a subset of the column (for example, the range A3:A999), you introduce overhead and the potential for inaccuracy or errors.
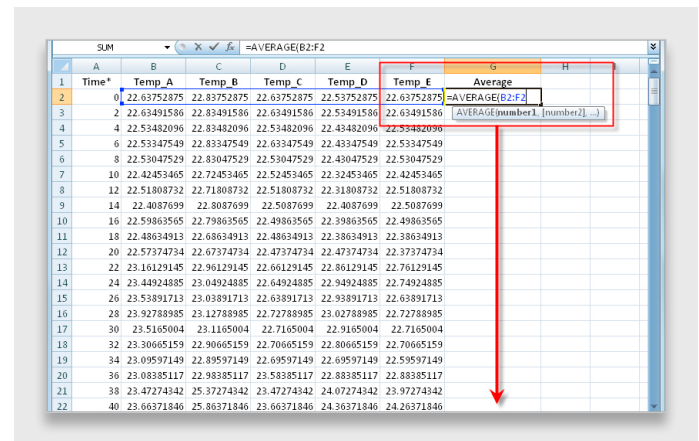

FIGURE 9

Microsoft Excel uses the cell as its fundamental building block. Even simple data analysis must be applied to a cell and then repeated for all cells in a column (channel).

Figure 9 shows how you can use Microsoft Excel to perform a common engineering task: averaging five temperature channels stored in columns to create a resultant average channel. You must first implement the averaging calculation in the cell building block and then copy it to or fill in all cells in the resultant column.

Data Volume

These days, common application data-streaming speeds often reach or exceed MS/s rates. An application that collects one single channel of data at 1 MS/s collects a total of 1,000,000 data points in a one-second acquisition. In a matter of minutes, billions of data points occupy gigabytes of hard drive space.

Traditional data management tools attempt to open a large data file by loading every single data point into memory, which takes valuable time. Their cell-centric flexibility is ideal for business spreadsheets where cell-level visibility is key, but it adds unnecessary memory overhead for data sets with millions of values. To avoid potential memory problems, traditional reporting tools often impose a limit to the maximum number of data values that can be loaded for a given column. This usually requires readjusting your storage strategy by either choosing a new file format (possibly having to rearchitect your application after the fact) or segmenting data into many small files just so that your reporting tools can open them.

When designing a DAQ system, be sure that your data management tool can handle your file format as well as the volume of data you intend to acquire (leaving flexibility for a change in requirements that might add to the volume of data collected in the future).

Does my data management software offer the analysis that I need?

Inline analysis, or analysis performed as your data is being acquired, is not always appropriate. If you don't need to make decisions as you acquire data, you may opt to perform offline analysis, also known as postprocessing. This involves saving acquired data to disk for unlimited interaction and having your data management software perform the analysis.

Because you perform this analysis after you acquire the data, you are not limited by DAQ timing and memory constraints, freeing up your processor for more challenging tasks or trend analysis that requires the entire dataset. Histograms, trending, and curve-

fitting are all examples of postprocessing tasks. Postprocessing offers far greater data interactivity, giving you the ability to truly explore both the raw data and analysis results. Furthermore, it means that you don't need to worry about analysis bottlenecking a live acquisition, considering the amount of time that intense signal-processing algorithms can take when operating on large data sets.

When choosing data management software, consider your post-processing needs. Check the data-management software manual to confirm that it offers the appropriate math, plotting, and data-reduction functions for your system.

Does my data management software offer the visualization that I need?

When it comes to reporting, you probably require, at a minimum, basic charting and graphing capabilities. Luckily, almost every data management tool on the market can make simple charts and graphs. However, be certain that those charts can handle graphing the volume of data you intend to plot, as many impose a limit to data-point count.

If you think you might need to graph different curves with drastically different y-scales on the same chart, your reporting tool needs to be able to distinguish between these scales. While many tools can, they also have a limiting maximum y-axis count.

In addition, think about whether your reporting needs go beyond basic 2D graphing. For example, if you need to represent data using polar plots, or if your data would be best represented in a 3D graph, your reporting tool must support that.

Can I use templates to simplify repetitive reports?

Many times, you need the same type of report for a series of raw data files. For example, if you run the same tests every week and have to report standardized results, you reuse the same report layout across multiple data sets. Traditional reporting tools save the report display along with the raw data in a common spreadsheet file, which makes it much harder to use a particular report display for multiple data sets. Each data set contains its own report layout and formatting, which means that if you need to make a modification to the layout or formatting—for example, something as simple as changing the color of a curve—you must edit every file to standardize on that change.

By creating templates, you can more easily create custom reports to update with new data and results. If you anticipate having to create the same report multiple times across several data sets, you need a data management tool that you can use to produce a report template and apply it to different raw data files.

Can I automate reporting to save time?

Typically, a DAQ application uses one of two types of reporting: infrequent or repetitive. Infrequent reporting happens on an occasional basis, usually in an interactive, customized way. Alternatively, repetitive reporting is frequent and usually standardized, and often uses templates.

If you have repetitive reporting needs, your data management software should be able to automate reporting. Even most traditional tools support macros or scripts that make this easier. Many have recording modes so that you can interactively record scripts that automate lengthy evaluations or calculations.

Does my data management software export reports in the right format?

Reporting tools usually present the final output in an easily exchangeable format that you can email, print, or present, regardless of the original raw data f ile format. With most reporting tools, you can export reports to several formats, but be sure that your tool supports your most common format (whether that's PDF, PowerPoint, imagery, or HTML).

Additionally, if you have immense reporting needs—for example, if your reports often span dozens of pages—ensure that the reporting tool you choose can export your report in your desired format at your desired size. The last thing you want to do is recreate all of your work at the very end of your system's design simply because your reporting tool can't create reports at your required length.

Find out how NI DIAdem can help you manage and report your measurement data

Next Steps

Let's get started designing your measurement system. We'll help you pick the right sensors to measure the physical phenomena you're interested in and select the proper DAQ device to read those signals to analyze, visualize, and report out on your data.
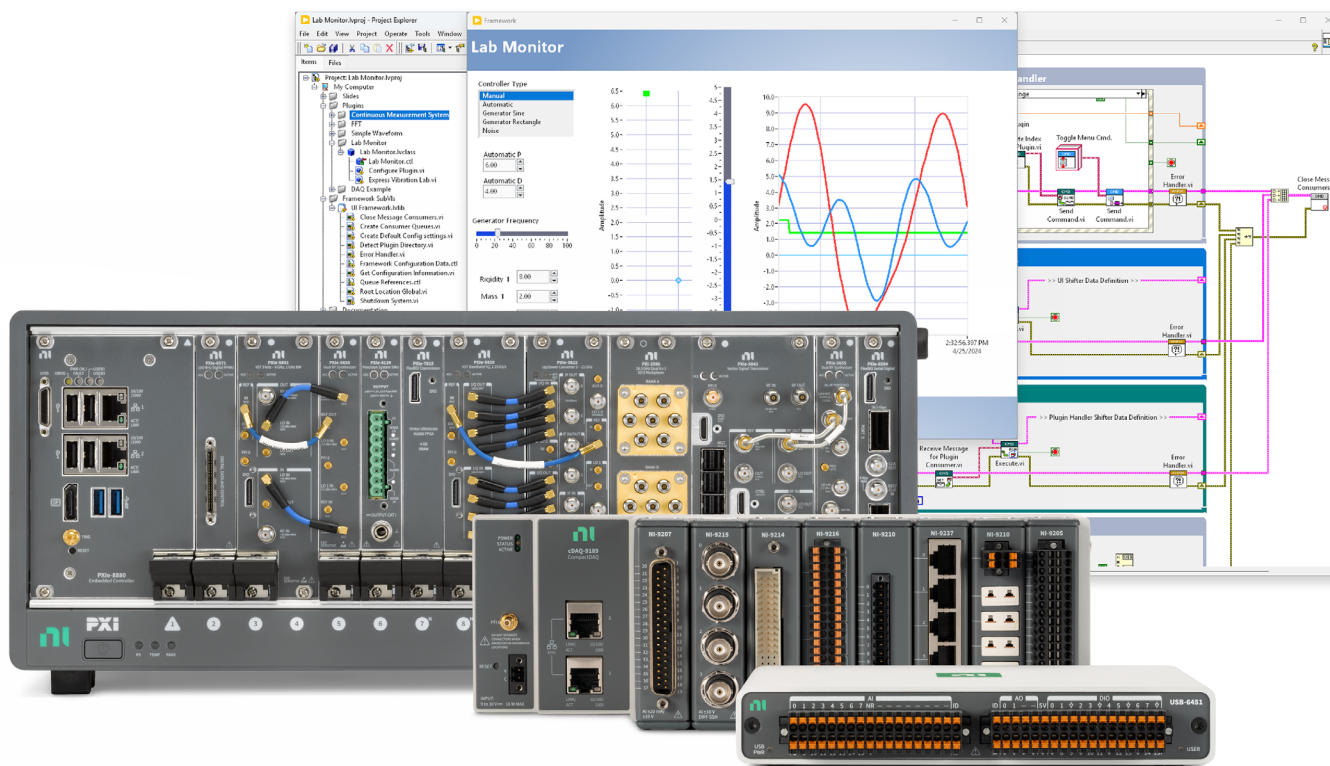

FIGURE 10

Whatever your measurement challenge, NI has hardware and software to meet your needs.

Explore NI DAQ hardware and software

Consult an NI Sales Representative for your measurement system

NI
11500 N Mopac Expwy
Austin, TX 78759-3504

Linkedin.com/company/niglobal/

x.com/niglobal

Youtube.com/@NIGlobalYoutube

Instagram.com/niglobal/

Facebook.com/NationalInstruments