Complete Guide to Building a Measurement System

How to Choose the Right Sensor for Your Measurement System

Overview

You can choose from many different sensors on the market today to measure all types of natural phenomena. This white paper categorizes and compares the most common sensors for measuring seven of these phenomena to help you choose the best option for your application.

Temperature

The most common sensors for measuring temperature are thermocouples, thermistors, and resistance temperature detectors (RTDs). Fiber-optic sensors, while more specialized, are growing in popularity for temperature measurements.

Temp. Sensor	Signal Conditioning Required	Accuracy	Sensitivity	Comparison
Thermocouple	▪ Amplification ▪ Filtering ▪ Cold-Junction Compensation	Good	Good	▪ Self-Powered ▪ Inexpensive ▪ Rugged ▪ Large Temperature Range
RTD	▪ Amplification ▪ Filtering ▪ Current Excitation	Best	Better	▪ Very Accurate ▪ Very Stable
Thermistor	▪ Amplification ▪ Filtering ▪ Voltage Excitation	Better	Best	▪ High Resistance ▪ Low Thermal Mass
Fiber Optics	▪ Little or No Amplification ▪ Filtering	Best	Best	▪ Good for Hazardous Environments ▪ Good for Long Distances ▪ Immune to Electromagnetic Interference (EMI)-Induced Noise ▪ Small, Lightweight

Table 1. Comparison of 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 and other factors, temperature accuracy of less than 1 °C can be hard to achieve.

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 are primarily 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 previously mentioned sensors. They have the best accuracy (±.05 °C), but they are more fragile than thermocouples or RTDs. Thermistors involve excitation like the RTD; however, the thermistor requires voltage excitation rather than current excitation. A thermistor typically ranges between $2 and $10 USD in price.

Fiber Optics

Another alternative is the use of fiber optics to measure temperature. Fiber-optic temperature sensors are effective for environments that are hazardous or where there could be regular electromagnetic interference. They are nonconductive, electrically passive, immune to electromagnetic interference (EMI)-induced noise, and able to transmit data over long distances with little or no loss in signal integrity.

Strain

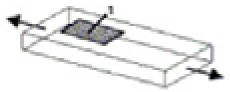
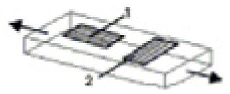
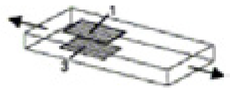
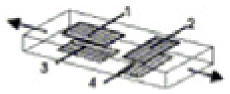
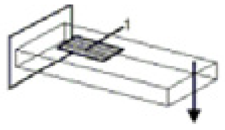
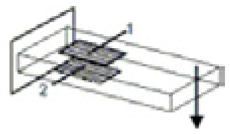
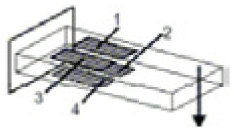
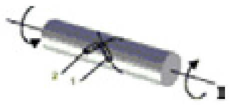
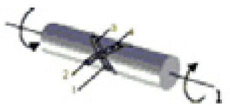
Strain	Gage Setup	Bridge Type	Sensitivity MV/V @100 uE	Details
Axial		¼	0.5	Good: Simplest to implement, but must use a dummy gage if compensating for temperature. Responds equally to axial strain.
		½	0.65	Better: Temperature compensated, but it is sensitive to bending strain.
		½	1.0	Better: Rejects 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 at 45 degrees from centerline.
		Full	2.0	Best: Most sensitive full-bridge version of previous setup. Rejects both axial and bending strains.

Table 2. Comparison of Common Strain Gage Configurations

Strain is typically measured by a resistive strain gage. These flat resistors are usually attached to a surface that is expected to flex or bend. One use case for resistive strain gages is structural testing of airplane wings. Strain gages can measure very small twists, bends, and pulls on surfaces. When more than one resistive strain gage is wired together, a bridge is created.

A more sensitive measurement is available with the purchase of more strain gages. You can use up to four active strain gages to build a Wheatstone bridge circuit; this is called a full-bridge configuration. There are also 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 will be.

Strain gages require current or voltage excitation and are susceptible to temperature drift, bending strain, and axial strain, which can give false readings without the use of additional resistive strain gages.

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

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 an object under measure.

Similar to temperature systems, fiber-optic sensors can be used to measure strain in hazardous environments, where a regular electrical measurement could be altered by electromagnetic interference. Fiber-optic strain sensors are nonconductive, electrically passive, immune to EMI-induced noise, and able to transmit data over long distances with little or no loss in signal integrity.

Sound

Microphones	Price	Environment	Impedance Level	Sensitivity	Comparison
Prepolarized Condenser	Medium	Tough	Medium	Best	▪ Condenser designs are most used ▪ Best in humid environments
Externally Polarized Condenser	High	Tough	Better	Good	▪ Condenser designs are most used ▪ Best in high-temperature environments
Carbon Microphone	Low	Average	High	Good	▪ Low quality ▪ Used in early basic design of telephone handset
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. Comparison of Common Sound Sensors

Microphones are used to measure sound, but you have many different types of microphones to consider when choosing a sensor for an application.

Condenser Microphones

The most common are condenser microphones. These may come either prepolarized (meaning that a power source is included within the microphone) or externally polarized. Externally polarized condenser microphones require an additional power source, which adds cost to projects. Prepolarized microphones are preferred in humid environments where a power supply's components could be damaged, and 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 (decibels) pressure ranges. Their disadvantage is the high noise levels they also pick up.

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. They are relatively cheap, and their only drawback is the lack of bass they provide. In addition, carbon microphones, which are less commonplace today, can be used in applications in which the quality of sound is not an issue.

Vibration

Vibration Sensors	Natural Frequency	Number of Axes	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 wiper of potentiometer
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. Comparison of Common Vibration Sensors

Ceramic Piezoelectric Sensor or Accelerometer

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

Three major factors differentiate vibration sensors: the natural frequency, the damping coefficient, and a scale factor. The scale factor relates the output to an acceleration input and is linked to sensitivity. Together, the natural frequency and damping coefficient determine the accuracy level of a vibration sensor. 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 sensors. These vibration sensors can be used in shock measurements (explosions and failure tests), high-frequency measurements, and slower low-frequency vibration measurements. This is shown by their higher than average natural frequency. However, this sensor typically has outputs in the millivolt range and requires a high-input-impedance, low-noise detector to interpret voltages from its piezoelectric crystal.

Proximity Probes and Linear Variable Differential Transformers (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 the wiper of a potentiometer.

Variable Reluctance Vibration Sensor

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

Position and Displacement

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 DifferentialTransformers (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. Comparison of Common Position Sensors

You can choose from many different types of position sensors. The driving factors in selecting a position sensor are excitation, filtering, environment, and whether 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. Position has been measured with sensors for a long time, so both preference and application play a role in making this decision.

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. This sensor provides no scale for how far away an object is from the sensor when the button is "off," but it is effective for applications that do not require highly detailed position information.

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

Optical Encoders
Another position sensor commonly used is the optical encoder, which can be either linear or rotary. These devices can 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 up the distance to be measured by counts. The more counts, the higher the accuracy. Some rotary optical encoders can have up to 30,000 counts to 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, like the potentiometer, add a small amount of resistance to the movement of the system's parts. However, encoders hardly produce any friction when they move and are very lightweight, but they must have seals to operate within a harsh or dusty environment, which adds to cost. An additional cost is also typically incurred in high-accuracy applications because optical encoders require their own bearings to avoid misalignment when incorporated into products.

Linear Variable Differential Transformers (LVDTs)
Linear variable differential transformers (LVDTs) and their rotary counterpart (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 use magnetic fields to determine position and are moderately priced. 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 in applications where large gaps exist between the sensor and target. Line of sight is required when using this sensor, and the accuracy and quality of this sensor is directly related to its price.

Pressure

High or low pressure is all relative – like 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.

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.

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. Comparison of Relative Pressure Measurement Types

- 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 could also 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. This is due to their simple construction and durability. These characteristics allow for 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 the regular circuitry for measurement. 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, 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. Comparison of Common Load Cell Sensors

At one time, mechanical lever scales were primarily used to measure force. Today, strain gage based load cells are the most common because they do not require the amount of calibration and maintenance that scales need.

Load cells can be either conditioned or nonconditioned. Typically conditioned sensors are more expensive because they contain components for filtering, signal amplification, as well as excitation leads, and the regular circuitry for measurement. 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 are typically used in weighing applications of both small and large items (10 lb up to 5k lb). They have an average sensitivity, but are highly accurate. This load cell has simple construction and a low cost.

The S beam load cell is similar to the beam style with the exception of 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 can also 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, you may consider using the canister load cell instead.

Button and washer load cells are typically used to measure the weights 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 must also be centered on what is usually a small scale. The benefit to these load cells is that they are inexpensive.

▷ Learn more about specific sensor types

How to Choose the Right DAQ Hardware for Your Measurement System

Overview

With many data acquisition (DAQ) devices to choose from, it can be difficult to select the right one for your application. This white paper outlines five questions that you should ask when selecting your hardware.

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

Different types of signals need to be measured or generated in different ways. A sensor (or transducer) is a device that converts a physical phenomenon into a measureable electrical signal, such as voltage or current. You can also send a measureable electrical signal to your sensor to create a physical phenomenon. For this reason, it is important to understand the different types of signals and their corresponding attributes. Based on the signals in your application, you can start to consider which DAQ device to use.

Functions of DAQ Devices

- 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

There are devices that are dedicated to just one of the functions listed above, as well as multifunction devices that support them all. You can find DAQ devices with a fixed number of channels for a single function, including analog inputs, analog outputs, digital inputs/outputs, or counters; however, you should consider purchasing a device with more channels than you currently need so that you can increase channel count if necessary. If you purchase a device that only has the capabilities for your current application, it will be difficult to adapt the hardware to new applications in the future.

Multifunction DAQ devices have a fi ed channel count, but offer a combination of analog inputs, analog outputs, digital inputs/outputs, and counters. Multifunction devices support different types of I/O, which gives you the ability to address many different applications that a single-function DAQ device would not.

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 and a variety of I/O modules. An advantage of a modular system is that you can select different modules that have unique purposes, so more configurations are possible. With this option, you can find modules that perform one function more accurately than a multifunction device. Another advantage of a modular system is the ability to select the number of slots for your chassis. A chassis has a fixed number of slots, but 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, like 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 low-pass filtering to remove high-frequency noise. Signal conditioning provides a distinct advantage over DAQ devices alone because it enhances both the performance and measurement accuracy of DAQ systems.

Table 1 provides a summary of 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, 4-20mA)	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 1. Signal Conditioning for Types of Sensors and Measurements

If your sensor is listed in Table 1, you should consider signal conditioning. You can add external signal conditioning or choose to use a DAQ device with built-in signal conditioning. Many devices also include built-in connectivity for specific sensors for convenient sensor integration. For a more in depth guide on signal conditioning, please see The Engineer's Guide to Signal Conditioning.

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

One of the most important specifications of a DAQ device is the sampling rate, which is the speed at which the DAQ device's ADC takes samples of a signal. Typical sampling rates are either hardware- or softwaretimed and are up to rates of 2 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.

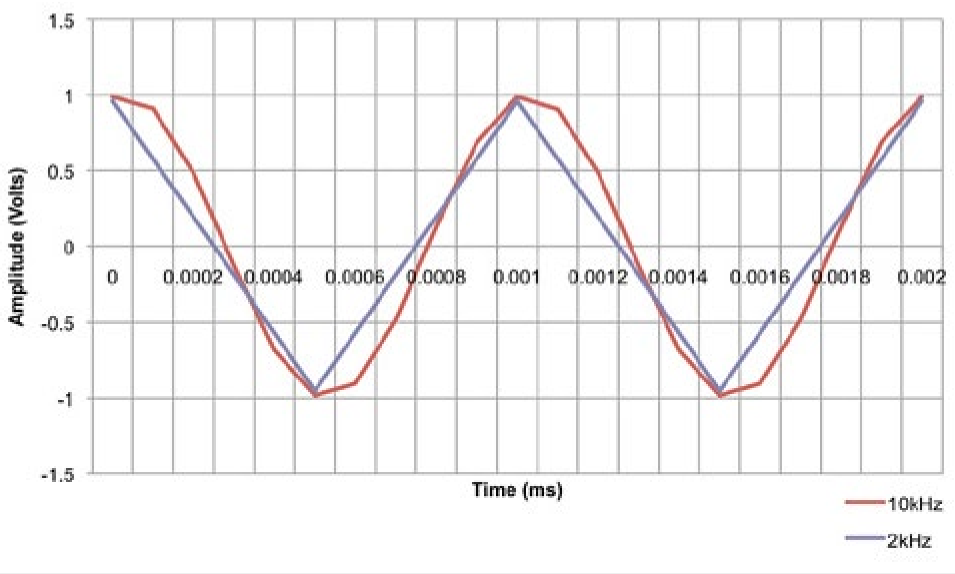


Figure 1. 10 kHz Versus 2 kHz Representation of a 1 kHz Sine Wave

The Nyquist 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 in your application 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 it is highly recommended to sample at 10 kHz to measure or generate a more accurate representation of your signal. 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 for the application.

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

The smallest detectable change in the signal determines the resolution that is required of your DAQ device. Resolution refers to the number of binary levels an ADC can use to represent a signal. To illustrate this point, 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 (23) discrete voltage levels. A 16-bit ADC can represent 65,536 (216) discrete voltage levels. The representation of the sine wave with a 3-bit resolution looks more like a step function than a sine wave where the 16-bit ADC provides a clean-looking sine wave.

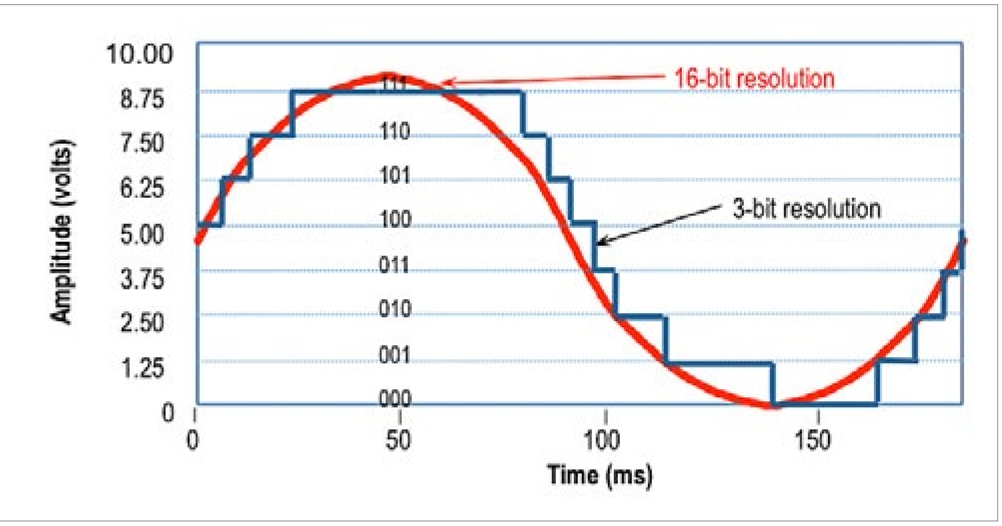


Figure 2. 16-Bit Resolution Versus 3-Bit Resolution Chart of a Sine Wave

Typical DAQ devices have voltage ranges of +/-5 V or +/- 10 V. The voltage levels that can be represented 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 (212 or 4,096 evenly distributed levels) can detect a 5 mV change, where a device with 16 bits of resolution (216 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 can likely 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.

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 always measures the true value with 100 percent certainty, but in reality, 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 a National Instruments multifunction device's absolute accuracy is given below:

Absolute Accuracy = ([Reading*Gain Error] + [Voltage Range*Offset Error] + Noise Uncertainty)

Absolute Accuracy = 2.2 mV

It is important to note that an instrument's accuracy depends not only on the instrument, but also on the type of signal being measured. If the signal being measured is noisy, the measurement's accuracy is adversely affected. There are wide ranges of DAQ devices with varying degrees of accuracy and price points. Some devices may provide self-calibration, isolation, and other circuitry to improve accuracy. Where a basic DAQ device may provide an absolute accuracy over 100 mV, a higher performance device with such features may have an absolute accuracy around 1 mV. Once you understand your accuracy requirements, you can choose a DAQ device with an absolute accuracy that meets your application needs.

▷ Compare different DAQ hardware products for your application

How to Choose the Right Bus for Your Measurement System

Overview

When you have hundreds of different data acquisition (DAQ) devices to choose from on a wide variety of buses, it can be difficult to select the right bus for your application needs. Each bus has different advantages and is optimized for throughput, latency, portability, or distance from a host. This white paper examines the most common PC bus options and outlines the technical considerations to keep in mind when choosing the right bus for your measurement application.

Other Topics

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 in a certain period of time. This is known as the bus bandwidth and 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 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 have a certain sampling rate and resolution that need to be achieved 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:

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

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 data on a large number of 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 are often dependent on I/O values to be 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.

Bus latency is the responsiveness of I/O. 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.

In a proportional integral derivative (PID) control system, for example, this bus latency can directly impact the maximum speed of the control loop.

Another important factor in single-point I/O applications is determinism, which is a measure of how consistently I/O can execute on time. Buses that always have the same latency when communicating with I/O are more deterministic than buses that can vary their responsiveness. Determinism is important for control applications because it directly impacts the reliability of the control loop, 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 in latency with poor determinism.

The software side of how a communications bus is implemented plays a big part in bus latency and determinism. Buses and software drivers that have support for real-time OSs provide the best determinism and therefore give you the highest performance. In general, internal buses such as PCI Express and PXI Express are better for low-latency single-point I/O applications than external buses such as USB or wireless.

Do I need to synchronize multiple devices?

Many measurement systems have complex synchronization needs, whether it is 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. The simplest way to synchronize measurements across multiple devices is to share a clock and a trigger. Many DAQ devices offer programmable digital lines for importing and exporting both clocks and triggers. Some devices even offer specialized trigger lines with BNC connectors. These external trigger lines are common on USB and Ethernet devices, as the DAQ hardware is located outside of the PC enclosure. Certain buses, however, have additional timing and triggering lines built in to make multidevice synchronization as easy as possible. PCI and PCI Express boards offer the real-time system integration (RTSI) bus, on which multiple boards in a desktop system can be cabled directly together inside the case. This removes the need for additional wiring through the front connector and simplifies I/O connectivity.

The best bus option for synchronizing multiple devices is the PXI platform, including PXI and PXI Express. This open standard was designed specifically for high-performance synchronization and trigging, with a number of different options for synchronizing I/O modules within the same chassis as well as synchronizing multiple chassis.

How portable should this system be?

The dramatic adoption of portable computing is undeniable, and has offered new ways to innovate with PC-based data acquisition. Portability is an important factor for many applications, and could easily be the primary reason to choose one bus over another. In-vehicle DAQ applications, for example, benefit from hardware that is compact and easily transported. External buses like USB and Ethernet are particularly good for portable DAQ systems because of quick hardware installation and compatibility with laptop computers. Bus-powered USB devices offer additional convenience because they do not require a separate power supply to be present, and are powered off the USB port. Using wireless data transfer buses is another good option for portability because the measurement hardware itself becomes portable while the computer can remain stationary.

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.

Selection Guide for the Most Common Buses

Based on the five questions previously outlined, Table 1 shows a selection guide for the most common DAQ buses.

Bus	Waveform[1] Streaming	Single-Point I/O	Multidevice	Portability	Distributed Measurements	Example
PCI	132 MB/s (shared)	Best	Better	Good	Good	M Series
PCI Express	250 MB/s (per lane)	Best	Better	Good	Good	X Series
PXI	132 MB/s (shared)	Best	Best	Better	Better	M Series
PXI Express	250 MB/s (per lane)	Best	Best	Better	Better	X Series
USB	60 MB/s	Better	Good	Best	Better	NI CompactDAQ
Ethernet	125 MB/s (shared)	Good	Good	Best	Best	NI CompactDAQ
Wireless	6.75 MB/s (per 802.11g channel)	Good	Good	Best	Best	Wi-Fi DAQ

[1]Maximum theoretical data streaming rates are based on the following bus specifications: PCI, PCI Express 1.0, PXI, PXI Express 1.0, USB 2.0, Gigabit Ethernet, and Wi-Fi 802.11g

Table 1. This table shows a bus-selection guide based on application requirements with example NI products.

Overview of DAQ Buses

While there are many different buses and form factors to choose from, this section focuses on the seven most common buses:

- PCI
- PCI Express
- USB
- PXI
- PXI Express
- Ethernet
- Wireless

In Figure 1, the buses are organized into a PC-bus hierarchy of National Instruments DAQ products, from internal plug-in options to hot-swappable external buses.

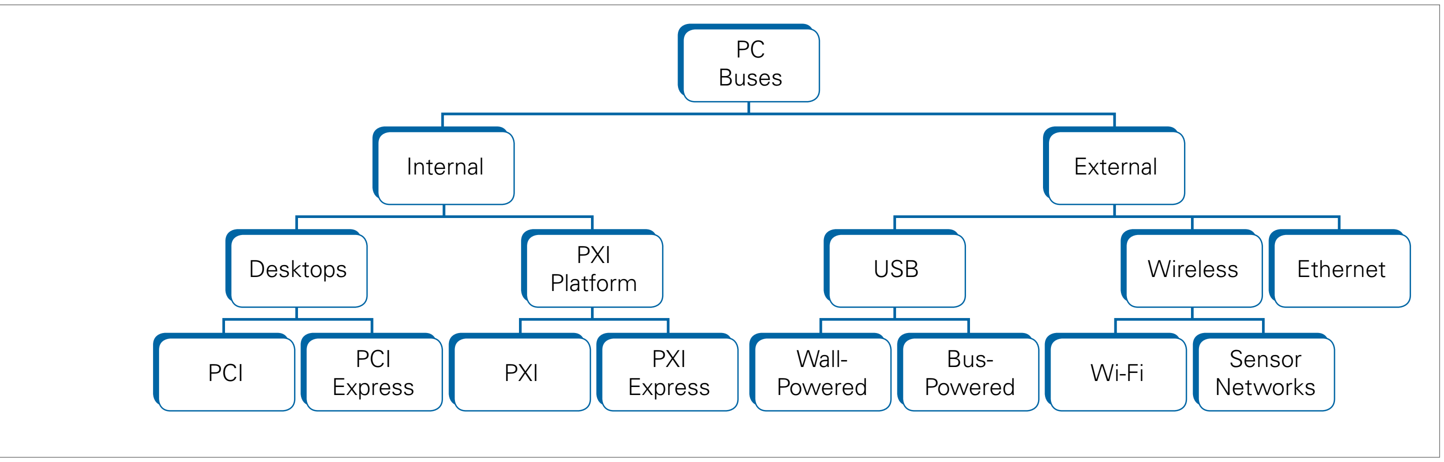


Figure 1. You can choose from many buses to meet your DAQ requirements.

Figure 2. PCI M Series Multifunction DAQ

PCI

The Peripheral Component Interconnect (PCI) bus is one of the most commonly used internal computer buses today. With a shared bandwidth of 132 MB/s, PCI offers high-speed data streaming and deterministic data transfer for single-point control applications. There are many different DAQ hardware options for PCI, with multifunction I/O boards up to 10 MS/s and up to 18-bit resolution.

▷ Learn about PCI DAQ devices

Figure 3. PCI Express X Series Multifunction DAQ

PCI Express

PCI Express, an evolution of PCI, offers a new level of innovation in the PC industry. The single biggest benefit of PCI Express architecture is the dedicated bus bandwidth provided by independent data transfer lines. Unlike PCI, in which 132 MB/s of bandwidth is shared among all devices, PCI Express uses independent data lanes that are each capable of data transfer up to 250 MB/s.

The PCI Express bus is also scalable from a single x1 (pronounced "by one") data lane to x16 data lanes for a maximum throughput of 4 GB/s, enough to fi a 200 GB hard drive in less than a minute. For measurement applications, this means higher-sustained sampling and data throughput rates, and multiple devices do not have to compete for time on the bus.

▷ Learn about PCI Express DAQ devices

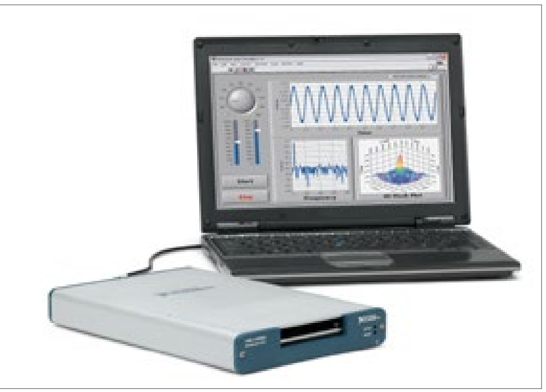


Figure 4. YUSB X Series adds data acquisition to any computer with a USB port.

USB

The Universal Serial Bus (USB) was originally designed to connect peripheral devices, such as keyboards and mice, with PCs. However, it has proven useful for many other applications, including measurement and automation. 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 recommend 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 some traditional internal PC buses. Devices that connect using USB are hot-pluggable, so they eliminate the need to shut down the PC to add or remove a device. The bus also has automatic device detection, meaning that users do not have to manually configure their devices after plugging them in. Once the software drivers have been installed, the OS should detect and install the device on its own.

▷ See NI options for USB data acquisition

Figure 5. The PXI platform is composed of chassis, controllers, and I/O modules.

PXI Platform

PCI eXtensions for Instrumentation (PXI) were developed to bridge the gap between desktop PC systems and 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 specification for measurement and automation. The benefits of PXI-based DAQ systems include rugged packaging that can withstand the harsh conditions that often exist in industrial applications. PXI systems also offer a modular architecture, which means that you can fit several devices in the same space as a single stand-alone instrument, and you have the ability to 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, multiple devices can be synchronized by using the internal buses resident on the backplane of a PXI chassis.

▷ Compare NI options for data acquisition with PXI

Ethernet

Ethernet is the backbone of almost every corporate network in the world and, therefore, is widely available. As a bus for DAQ, 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 in combination with a large install base of networks in labs, offices, and manufacturing facilities makes Ethernet an ideal choice for distributing measurements to remote locations. Though available network bandwidth is dependent on the number of networked devices, 100BASE-T (100 Mbit/s) Ethernet can accommodate multiple Ethernet DAQ devices running at full speed. In addition, Gigabit Ethernet (1000BASE-T) can aggregate data from multiple 100BASE-T networks or higher speed devices for larger systems.

Figure 6. Supporting 100 m per segment and the ability to use existing network infrastructure, Ethernet data acquisition can extend the reach of your measurement system.

▷ See NI options for Ethernet data acquisition

Wireless

Wireless technology extends the flexibility and portability of PC-based data acquisition to measurement applications where cables are inconvenient or impractical, such as wind farms or civil structures. Wireless can dramatically reduce costs by eliminating cables and installation time. However, wireless also has the highest latency of any other DAQ bus, so applications requiring high-speed control or determinism are not recommended. There are many different wireless technologies available. The most popular is IEEE 802.11 (Wi-Fi).

Figure 7. Wi-Fi data acquisition uses standard 802.11 networks to eliminate the need for wires between measurement hardware and the host PC.

Wi-Fi is among the easiest of wireless technologies to set up. Connecting to a Wi-Fi "hotspot" is as familiar to most as plugging in a USB cable. After 10 years of use in the IT sector, Wi-Fi is also secure. IEEE 802.11i (WPA2) is the highest commercially available wireless security standard to date with 128-bit AES encryption and IEEE 802.1x authentication. For streaming dynamic waveform signals, Wi-Fi provides more bandwidth than other wireless technologies, making it ideal for machine condition monitoring and other high-speed applications.

▷ Learn about NI wireless data acquisition

▷ Compare different DAQ hardware products for your application

How to Choose the Right Computer for Your Measurement System

Overview

Once you have chosen your data acquisition device, you can easily take for granted the process of selecting 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 data acquisition device, running the software to control the device, analyzing the measurements, and saving the results. This white paper examines the information you need to choose the right computer for your application.

Other Information About Computers

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 of the computer. Processors in most new computers are either dual- or quad-core, meaning the computer can use two or more independent actual processors (called "cores") to read and execute program instructions. The processing power for a computer also consists of the RAM, the hard drive size, and the processor speed. More RAM improves speed and enables more applications to run at the same time. More hard drive space gives you the ability to store more data. Finally, faster processors enable faster operation of your application. In general, faster is better, but processor speeds across brands may not be the same. 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 a key feature to consider for your computer if you are going back and forth between applications or locations frequently. For example, a portable computer is essential for taking measurements in the field and then returning to the lab to analyze the data. Portability is also an important feature if you need to monitor applications in different locations. The key considerations when assessing portability are the product size and weight. You wouldn't want to carry around a heavy computer that is difficult to hold.

How much does the computer cost?

Budget is a concern for nearly every project, and more than likely your computer makes up a large portion of the total system cost. Features and form factor contribute the most to overall cost. Choosing a computer for your application becomes a trade-off between price and performance; better features cost more and drive up the price. For example, a computer with a faster processor is more expensive. Additionally, form factor makes a difference in the cost of the computer. Typically, between a laptop and a desktop with similar features, a laptop is more expensive due to the added portability. Finally, industrial specifications or optimizations for instrumentation may increase system cost with the added benefit of providing a rugged test platform.

How rugged do I need the computer to be?

The ruggedness of your computer can be a crucial feature if you are monitoring your application in an extreme environment. The specifications of a computer that refer to ruggedness are the operating conditions. Standard off-the-shelf PCs are not designed to withstand the conditions of industrial environments. For example, the operating conditions for computers consist of operating and storage temperature, relative humidity, and maximum operating and storage altitude. 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); therefore, any computers featuring specifications greater than these are considered rugged. You need to pay attention to this feature only if your application requires it.

Do I need my computer to be modular?

The modularity of your computer can be crucial if you are considering future applications or are working on multiple applications. Modularity describes the degree to which a system's components may be separated and recombined. If you want to switch modules in your system or modify your applications easily, having a modular system is crucial. The flexibility you can achieve with a modular computer is unmatched. You can modify and adapt the system to meet your particular needs and expand for the future as well as upgrade individual components without having to buy a whole new system. With a modular system, you can install a new hard drive if you need more space or use a data acquisition device with a faster analog-to-digital converter if you need faster sampling. Keep in mind that 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 needs.

Do I need a real-time operating system?

The operating system is an important feature to consider when choosing a computer for data acquisition. By far the most common general-purpose operating system is Windows, but data acquisition and control applications occasionally require a more specialized operating system. A real-time operating system gives you the ability to operate deterministically, meaning applications can execute according to precise timing requirements. A real-time operating system is deterministic because the operating system does not determine which process happens when; rather, the user defines the order and timing of the processes. This gives you more control over your application and the ability to execute at faster rates than with a nondeterministic operating system. If you need a computer with a deterministic operating system, then you want to look for computers that meet those requirements.

▷ Learn more about a real-time operating system

Computer Selection Guide

Based on the six questions previously examined, Table 1 shows a selection guide for the most common types of computers.

	PXI System	Desktop	Industrial PC	Laptop	Tablet
Processing Power	Best	Best	Better	Better	Good
OS Compatibility	Best	Best	Good	Better	Good
Modularity	Best	Better	Better	Good	Good
Ruggedness	Better	Better	Best	Good	Good
Portability	Better	Good	Good	Best	Best
Cost	Good	Better	Good	Better	Best

Table 1. This table shows a computer selection guide based on the six most important features.

Overview of Computer Types

Five different types of computers are typically used for data acquisition. The computer communicates with your data acquisition hardware; therefore, which computer you choose depends on your data analysis needs, the environment that the system operates in, and the channel count needed for your system. This section focuses on the following types of computers:

- PXI system
- Desktop
- Industrial PC
- Laptop
- Tablet

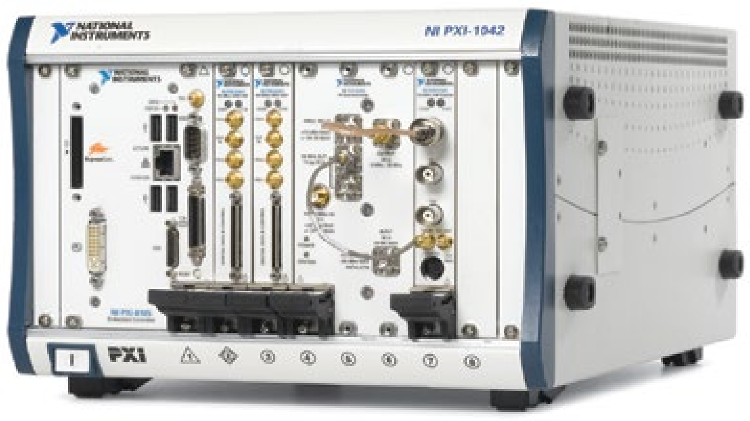


Figure 1. A PXI system contains a controller, chassis, and up to 17 modular instruments.

PXI System

PXI (PCI eXtensions for Instrumentation) is a modular, rugged, PC-based platform for measurement and automation systems. A PXI system consists of a controller, chassis, and instrumentation modules. The PXI system controller runs the operating system and serves as the "computer" for the system; it contains a processor, RAM, hard drive, and so on. The chassis houses the controller and contains anywhere from 4 to 18 slots, which you can use to combine your computer and instrumentation into a single, compact package. If your application incorporates a variety of measurements and requires tight synchronization between instruments, or if your instrumentation needs to adapt to applications in the future, your best solution may be PXI. PXI is a powerful, flexible instrumentation platform; however, the modularity of a PXI system may involve a higher initial cost than a laptop or desktop with USB instrumentation. That being said, choosing a PXI system upfront could save time and money in the future as your instrumentation needs change.

▷ Learn more about PXI systems

Figure 2. Desktop computers often include the latest PC technology at a reasonable price.

Desktop

A desktop computer is a PC intended for regular use at a single location. Desktop computers are typically used in offices, labs, or anywhere that is not an extremely harsh environment. These computers consist of a few parts: a monitor, a keyboard, a mouse, and the computer itself. Since there are so many parts to a desktop computer, you do not want to repeatedly move it to different locations. However, because desktop computers are larger, they can dissipate more heat, allowing them to house larger, more powerful processors. Therefore, the biggest attribute of the desktop computer is the processing power. If you need to analyze data or log data to disk at fast rates, but you don't require mobility, the desktop computer may be exactly what you need for your application.

Figure 3. Industrial computers feature all of the components of a standard PC with a rugged mechanical design.

Industrial PC

Industrial PCs, as the name suggests, are special computers optimized for use in industrial or harsh environments. Industrial PCs are mechanically stronger and environmentally rated for extreme vibration, shock, temperature, and humidity. However, due to their robust design, these PCs are more expensive than other types of computers. These ratings are essential for many applications, so if you need to monitor applications in harsh environments, the industrial PC is a great choice for your application.

Figure 4. You can use lightweight, portable laptop computers to acquire data in the field.

Laptop

A laptop computer is a PC intended for mobile use. Because of their size, laptops are typically used with portable measurement systems. All of the parts are housed together in one laptop, making it very easy to move it from place to place. A laptop gives you the freedom to monitor different applications at multiple locations with ease. But since all of the parts are together in one unit, you need to make sure the environment you expose the laptop to is not going to damage it. For instance, most laptops may not be rated for dust or moisture. If you are looking for a portable, general-purpose computer with the ability to analyze and store data, a laptop computer may meet your needs.

Figure 5. Tablets sacrifice processing power, but offer lower cost and smaller size.

Tablet

Tablets have become an increasingly popular computing platform optimized for mobile and touch performance. There are many tablet hardware and software configurations available in the market today and when choosing a tablet for a data acquisition, these configurations should be evaluated. For data acquisition tasks, first make sure that the data acquisition software and hardware will be compatible with your tablet hardware and operating system (OS). An Android or iOS tablet may be useful for some job tasks, but unless your data acquisition hardware supports these operating systems, a tablet running Windows may be a more suitable option. Overall tablets are extremely convenient and portable devices, but make sure to double check compatibility and performance for your task before deciding on this computing platform.

How to Choose the Right Driver Software for Your Measurement System

Overview

Driver software is usually overlooked when selecting a data acquisition (DAQ) device. It handles the communication layer between hardware devices and application software. While hardware specifications are very important, poor driver software can greatly impact the development time and performance of your entire system. This white paper examines the questions you need to keep in mind when evaluating a DAQ device's driver software.

Is the driver compatible with my operating system?

You can choose from many operating systems, such as Windows, Mac OS, and Linux, which offer different advantages for different tasks, operations, and deployments. Each of these operating systems may also have different versions, distributions, or designs for specific processors. For example, releases of Windows range from Windows Vista to Windows 10, which feature different versions for 32-bit and 64-bit processors. Since Linux is open source, you can choose from hundreds of varieties. Each type, release, and version of an operating system functions differently and may or may not be cross-compatible.

As a result, DAQ drivers generally do not support every single operating system type and version. Most DAQ drivers work with Windows operating system releases since they are the most common. However, if you use an alternative operating system, you should always remember to confirm if the DAQ device driver supports it before choosing a DAQ device. You can generally find the operating system 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 to the DAQ hardware. Normally, the library is provided with documentation and distributed with 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 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.

The best integration is when 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 since functions and documentation are directly built into the 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 documentation that is muddled and incomplete can be a huge waste of time. When a driver's programming interface is poorly documented, you can spend an unnecessary and frustrating amount of time running trial and error tests on its functionality. Although trial and error can be a great way to learn the functions and syntax, you need to be able to refer back to the manual when necessary. 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, so you should carefully consider this when selecting a DAQ device.

Is the driver scalable to other devices?

You may have difficulty determining 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.

Drivers that are designed for a single device are typically more lightweight than drivers that work with a wide range of devices. While these drivers may get the job done initially, adding a new device or replacing an existing one may require a significant amount of programming work to integrate the corresponding new driver. The driver's programming interface may be structured differently and may require significant changes to your code.

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 may also support other features that make synchronizing and combining measurements from multiple devices easier.

▷ Read more about NI driver software for data acquisition

How to Choose the Right Application Software for Your Measurement System

Overview

Application software lies at the core of modern data acquisition (DAQ) systems, making it imperative that you select a software tool that fits the needs of your application today and easily scales as your system matures. The last thing you want to do is rewrite all of your code using new application software simply because your old code got too unwieldy to expand. Determining trade-offs to make when selecting the best application software tool for your DAQ system depends on the requirements that need to be met. This paper walks through five questions to consider when choosing your application software.

Is the software flexible enough to meet my future needs?

DAQ software tools range from ready-to-run programs (no programming required) to fully customizable application development environments. Although it is easy to make an application software decision based on current system development needs, it is important to consider how this tool can scale and solve problems as the system matures.

Ready-to-run software tools often have set functionality and are designed to perform specific measurement or test routines, usually with a limited subset of hardware options. This type of software tool is a good choice for your DAQ system if it meets the needs of your current development and you do not intend to modify or extend the functionality of your system. The major trade-off is that ready-to-run application software does not always easily scale to incorporate new functionality into an existing DAQ system.

To take advantage of an application software tool that meets the needs of your current system and scales over time, you should choose an application development environment in which you can create custom applications. Application development environments are extremely flexible in the sense that 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. The only trade-off is that you need to spend time up front to learn the programming language and develop the applications yourself. Although this may seem like a large time commitment, modern day development environments provide a variety of tools to help you get started including online and live training, getting started examples, code generating assistants, community forums to share code and discuss challenges, as well as personal help from applications engineers or general support teams.

How long will it take me to learn the software?

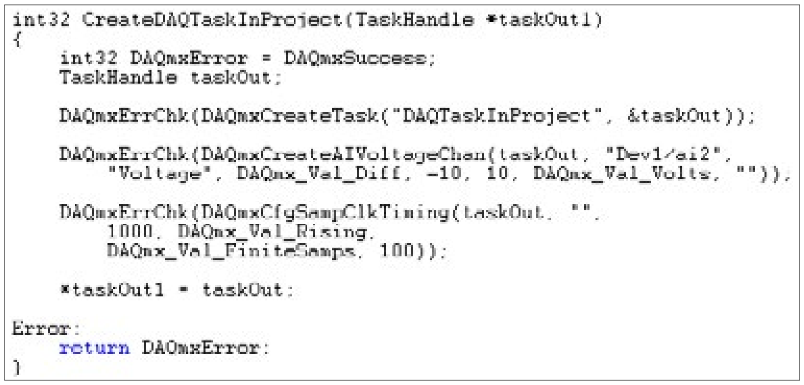
The time it takes to learn a new piece of software is different for everyone and depends on the type of software tool you choose and/or the language you use to program your DAQ applications.

Ready-to-run software tools are the easiest and fastest to learn because they have abstracted programming details from the user and typically only require a few details from the user for set up. When deciding among ready-to-run software tools for your DAQ system, you should ensure the tool has the capabilities to complete your data acquisition needs and you should ensure that proper resources are available to help you quickly learn the tool. Some of these resources include user manuals, in-product help information, online communities, and support forums.

Application development environments often take longer to learn, but the majority of that time is spent learning the language that is used within the environment to program your applications. If you can find an application development environment that uses a language you are already familiar with, you can definitely 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.

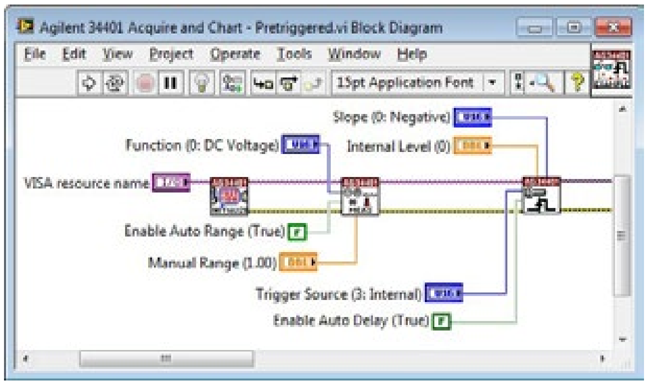
When evaluating application development environments that require you learn a new language, you should consider those that give you the ability to focus on the engineering problem at hand, rather than the low-level details of a programming language. Text-based languages, such as ANSI C/C++, are often more challenging to learn because of all the complex grammar and syntax rules that must be adhered to in order 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 is visually consistent to the way in which an engineer thinks.



```
int32 CreateDAQTaskInProject(TaskHandle *taskOut1)
{
    int32 DAQmxError = DAQmxSuccess;
    TaskHandle taskOut;

    DAQmxErrChk(DAQmxCreateTask("DAQTaskInProject", &taskOut));

    DAQmxErrChk(DAQmxCreateAIVoltageChan(taskOut, "Dev1/ai2",
        "Voltage", DAQmx_Val_Diff, -10, 10, DAQmx_Val_Volts, ""));

    DAQmxErrChk(DAQmxCfgSampClkTiming(taskOut, "",
        1000, DAQmx_Val_Rising,
        DAQmx_Val_FiniteSamps, 100));

    *taskOut1 = taskOut;

Error:
    return DAQmxError;
}
```

ANSI C Code



LabVIEW Code

You should also consider the getting started resources that are provided with the application software. These resources can help you get up and running with a new software tool in a shorter amount of time. The following are a few helpful getting started resources for any software tool:

- Evaluation – A free evaluation of the software gives you the opportunity to test things out for yourself and determine if the tool meets the needs of your application.
- Online Curriculum – Online tutorials, videos, and white papers are valuable when learning the basic concepts of application software.
- Classroom Instruction – A class on the application software is the perfect way to get up to speed and begin developing your DAQ system. The price and level of detail that a course covers depends on the type of instructional setting. Often, you can find options ranging from free seminars to formal classrooms to instructor-led online courses.
- Shipping Examples – Good shipping example sets have enough code for all of 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.

Does the software integrate my chosen driver and other productivity tools (analysis, visualization, storage)?

Too often, developers assume the existence of a device driver is sufficient enough for integrating their measurement device into a DAQ system. What these developers do not always consider is how this driver integrates with the application software they are using to develop the DAQ system. It is important that you choose a driver and software tool that are compatible with one another to successfully integrate your entire DAQ system.

DAQ systems often require integration with system and data management software to perform post processing, analysis, or data storage. Be certain that the application software you choose provides an easy way to manage the data after is has been acquired.

Analysis is common in a measurement system and most application software designed for data acquisition provides these routines through a signal manipulation tool or API. You want to ensure that the analysis routines needed for your system are provided within the application software, or else you have to face the added challenge of learning two environments – one for acquisition and one for analysis – as well as the pain point of shuffling data between them.

Visualization and data storage often go hand in hand within a DAQ system. The application software that you choose should have an easy way to visualize the data you acquire, whether it's through a predefined UI or through customizable UI controls, so that you can display the acquired data to a user. Additionally, the application software should have a simple way of integrating with system and data management software to store large amounts of data or numerous tests. Engineers frequently need to store data for manipulation at a later date, and your application software should have a variety of tools to accommodate a wide range of storage and sharing options. This provides you with the added flexibility of post processing data and generating standardized professional reports for collaboration.

Is there a community of resources to use when I get stuck?

The ecosystem that surrounds the application 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 own applications. You should take time to 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. You should 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 features of the software.

Does this software have a proven history of stability and success?

The last thing to consider when choosing application software for your DAQ system does not come in the form of formal documentation or feature specifications, but rather word of mouth. Browse through case studies where individuals were successful with the application software, or talk to those who have used the software tool in their own projects. Getting perspective from those outside the company in which the software is developed gives you a true indication of the history of stability and success. Choosing application software with proven stability and longevity helps ensure the reuse and scalability of your system such that your environment of choice doesn't go obsolete a short time into using your system.

▷ Read about LabVIEW to see if it fits your measurement application software needs

How to Choose the Right Analysis Tools for Your Measurement System

Overview

Raw data is not always the best way to communicate useful information. Data transformations like removing signal noise, compensation for environmental effects like temperature and humidity, and calibration for equipment error are needed to help turn raw data into useful data. Producing useful data is a primary outcome of engineering applications, so comprehensive signal processing is a fundamental need for any analysis tool used in data acquisition. This white paper outlines five questions to consider when choosing analysis tools for your DAQ system.

Will I need to analyze my data inline, offline, or both?

Most applications require some form of signal processing but a key decision that you need to make is where this processing takes place: inline, offline, or both.

Inline

Inline analysis implies that data is analyzed in the same application where it is acquired. If your application involves monitoring a signal and changing testing variables based on the characteristics of the incoming data, you should perform analysis inline. By measuring and analyzing certain aspects of the signals, you can make the application adapt to certain circumstances and enable the appropriate execution parameters – perhaps saving the data to disk in the case of an alarm scenario 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 an ability to easily integrate external IP.

The caveat to performing signal processing inline is that it takes time to execute these calculations. If your acquisition application has strict timing requirements, then make sure that your signal processing algorithm does not take too long to execute because you could miss some data while it is performing that operation. While you are developing your application, you can benchmark how long it takes to acquire and analyze your data and make sure that you aren't missing any data points. Something else that could help is parallelizing your code so that one section acquires data while another does the signal processing. This helps by using the multiple CPUs available in most machines, but benchmarking this type of application (for example, producer/consumer architecture) also makes sure it meets timing and data gathering requirements.

Point-by-point analysis is a subset of inline analysis where results are calculated after every individual sample rather than on a group of samples. Such analysis is essential when dealing with control processes featuring high-speed, deterministic, single-point data acquisition. The point-by-point approach simplifies the design, implementation, and testing processes because the application flow closely matches the natural flow of the real-world processes that the application is monitoring and controlling.

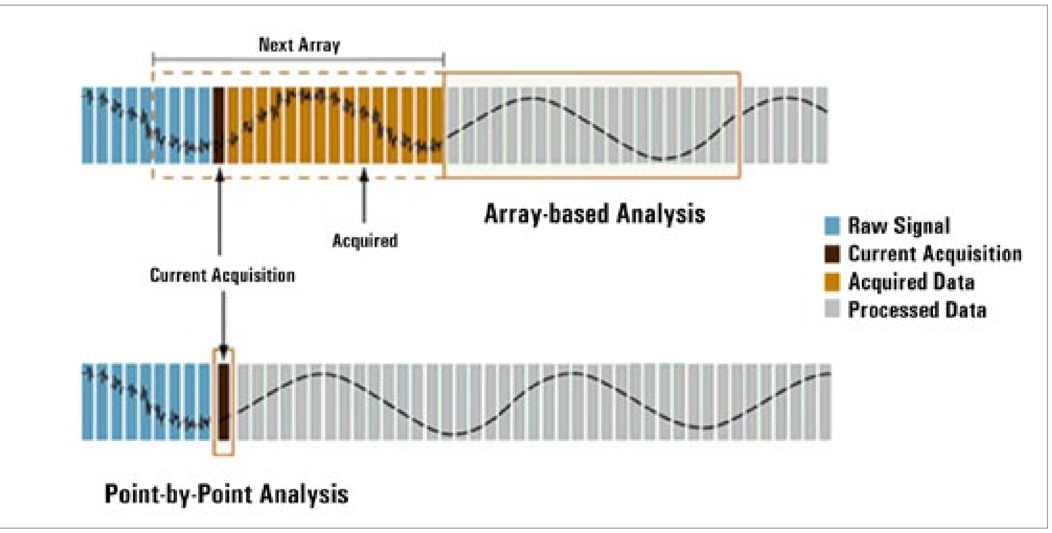


Figure 1. Array-Based Analysis Versus Point-by-Point Analysis

With streamlined point- by-point analysis, the acquisition and analysis process can move closer to the point of control because the latency between acquisition and decision is minimized. You can further decrease this acquisition latency by deploying your analysis to field-programmable gate arrays (FPGAs), digital signal processing (DSP) chips, embedded controllers, dedicated CPUs, and ASICs.

When you add powerful algorithms and routines to applications, you eliminate the guesswork and create intelligent processes that can analyze results during run time, improving efficiency and iteratively correlating input variables to experiment or process performance.

Offline

Inline analysis is not always the correct methodology when implementing your analysis routines. You may choose to perform offline analysis when you do not need to make decisions as you acquire the data. This involves saving acquired data to disk for unlimited interaction at a later time. Typically, the intent of an offline analysis application is to identify the cause and effect of variables by correlating multiple data sets. Because you perform this analysis after you acquire the data, you are not limited by the timing and memory constraints of data acquisition; such analysis requires only that sufficient computational power is available. This provides several advantages to performing analysis. First, offline analysis offers far greater data interactivity, giving you the ability to truly explore both the raw data and results of the analysis implementation. Histograms, trending, and curve-fitting are all common offline analysis tasks. Furthermore, analysis as a bottleneck for a live acquisition is no longer a concern, considering the amount of time that intense signal processing algorithms can take when operating on large data sets.

There are some applications that need a combination of both inline and offline data analysis. Normally the inline analysis routines are less intensive in these cases, and the offline analysis does most of the heavy lifting of intensive algorithms and data set comparison. An example of less intensive inline analysis would be logic to write data to a file or basic temperature conversion. If your application requires a combination of inline and offline, make sure your analysis tool can adequately support your needs.

Can my analysis tool(s) handle my data (volume, speed)?

A growing concern when choosing data analysis tools is the size and speed of the data it can process. DAQ machines are becoming faster and transducers are becoming cheaper. This means that engineers are collecting more data from more places faster than ever before. If the data analysis tools they use daily can't keep up with these new trends, then engineers have more data than ever, but nothing to effectively analyze it with.

Engineers and scientists are starting to find that their rudimentary data analysis tools do not keep up with their needs. Data analysis tools that were created for financial analysis and not proper data acquisition run into those limits. If you are trying to manipulate or correlate large data sets, then it would be beneficial to use analysis tools that are built for large data sets. Without proper data analysis instruments, you will see that performing any analysis is time-consuming, or you may not be able to analyze at all due to the sheer volume of data.

Talking with vendors about your data size, speed, and analysis requirements helps determine if a data analysis tool is right for you. It is always better to find a tool built specifically for large data sets since this is the trend for data acquisition.

Does my analysis tool(s) offer the functions I need?

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

Most data analysis tool vendors have a well-documented listing of what functions are included with their tool. If you know your specific needs for signal processing, searching the vendor function list works great. If you don't know exactly what you need, look for a tool that has a lot of functions related to your field or application type. Proper data analysis tools have several built-in functions (over 600). While basic and complex math operations are good to have, make sure that there are also functions specific to your area of interest. If your application deals with control, then look for proportional integral derivative (PID) control functions. If your application deals with optical character recognition (OCR), then look for those functions. By having specific functions available (built in or as an add-on), you can be more effective at developing your application because you won't have to spend time creating these functions yourself.

It is also important to remember that your analysis needs often grow over time; consider any analysis requirements you have today, but also verify that your application software leaves room for the inevitable expansion of your analysis needs.

Can I expand my application's analysis options through add-ons?

Investigating whether a data analysis tool has an ecosystem of add-ons is a necessary step. In the early days of software consideration, people picked a tool and were forced to accept only what the vendor put in the product. This meant they were at the mercy of the vendor to add any functionality or features they wanted. As software development progressed, vendors started creating add-ons that people could buy to extend the functionality of their product. This worked well, at the time, because users could purchase the specific functionality they needed; however, they were still limited by what add-ons the vendor offered.

Today, people expect an ecosystem of add-ons to extend their product not only from the vendor, but also third-party affiliates and other customers. This trend is validated by the emergence of "app stores" that extend a product's functionality. When deciding on a data analysis tool, look for one that has a strong ecosystem of add-ons so that you can extend the functionality of your product when you need it. Also look for vibrant communities of users sharing add-ons or IP for the product as these communities often offer added functionality for little to no cost.

Can I integrate my own custom or legacy analysis routines?

Sometimes engineers have proprietary analysis algorithms that simply can't be purchased as add-on software. Additionally, because application requirements change over time, most engineers often invest time and money creating analysis routines or custom IP in older or alternative tools. You should look for a data analysis package that is open to incorporating these external analysis routines. There is no need to 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, speak with the vendor to make sure you can incorporate the legacy analysis routine in their data analysis tool. If you can't easily add it, then you will spend a lot of time up front recreating the functionality in the new tool. Modern data analysis tools should be open to using IP created in other environments so that their users can be more efficient.

▷ Read more about built-in analysis and signal processing in NI LabVIEW

How to Choose the Right Visualization Techniques for Your Measurement System

Overview

Data visualization is a common aspect of almost all measurement systems. From simply graphing an acquired signal to correlating measurement data with video, sound, or 3D model projection, visualization options for data acquisition (DAQ) systems are extensive. Selecting the right visualization technique could mean the difference between being able to appropriately derive actionable information from raw data and missing important insight. This white paper covers five questions to consider when choosing your visualization tools.

Do I need to visualize my data inline, offline, or both?

Most applications require some form of visualization, but a key decision that you need to make is where this processing takes place: inline, offline, or both.

Inline

Inline visualization implies that data is displayed in the same application that acquired it. For example, acquired data may be displayed on a computer screen so that a technician can literally see the signal being measured and ensure that all connections are properly made. If inline analysis is run with inline visualization, a filtered version of the same signal might also be displayed on the monitor. This type of architecture gives you "instant feedback" since you can visualize acquired data in near real time, but it requires that your chosen application software contain all of the visualization tools you may need.

Just like with performing inline analysis, the caveat to visualizing data inline is that it takes extra processing power to execute required calculations and display data. User interface updates are one of the most processor-intensive actions that a CPU performs, which means that if your acquisition application has strict timing requirements, you must make sure that your visualization does not become a system bottleneck and cause you to miss some data. While you are developing your application, you can benchmark how long it takes to acquire, analyze, and visualize your data and make sure that you aren't missing any data points. Another option is parallelizing your code so that one thread performs the data acquisition while another handles all of the signal processing and visualization with a lower priority only as CPU resources become available. This helps by taking advantage of the multiple CPUs in most machines.

Offline

Though common, inline visualization is not always the correct methodology when implementing your system. In fact, it isn't even necessary in some applications. You may choose to perform offline visualization when you do not need to see data as it is being acquired or when you want to ensure that your computer processor can focus entirely on acquiring and streaming data to disk. Offline visualization involves storing data for inspection at a later date and requires an appropriate storage format and the selection of a dedicated offline visualization tool. However, opting to view data offline gives you unlimited flexibility when interacting with your data since you have access to the original raw data as it was acquired. Additionally, you are not limited by the timing and memory constraints of data acquisition, and visualization as a bottleneck during live acquisition is no longer a concern since the CPU no longer has to perform computationally intense graphics updates.

Many applications combine both inline and offline data visualization. Normally, inline visualization is limited to the minimal processing required to confirm the correct behavior of the system (for example, by slowing down the update rate of a graph). You can use offline visualization in conjunction with inline visualization to inspect and correlate data in detail when doing so no longer affects the acquisition itself.

Can my visualization tool(s) handle my data?

When choosing a data visualization tool, you must consider the volume of data that you can represent and the format of incoming data. If your data is being visualized inline using the application software you have chosen for acquisition, formatting shouldn't be an issue. However, remember that data rates influence the amount of data that needs to be represented, which ultimately affects the graphics processing power necessary to render the visualization.

If your visualization tool is offline, then format is a concern. You should make sure that the visualization tool that you choose is capable of interpreting the file or database format you intend to use to store data.

Additionally, even offline data analysis tools are limited by the amount of memory allocated by the operating system and therefore can load only a certain subset of larger data sets. Many visualization tools impose a data constraint because of this limitation and prevent engineers from both loading and graphing more than a predetermined volume of data. For example, many financial tools impose a loading limit of 1,048,576 (220) data points per column and a graphing limit of 32,000 points per chart. Selecting a visualization tool that was designed to handle engineering data sets helps you access and visualize your data appropriately and often includes data reduction techniques that simplify working with extreme data sets.

Does my visualization tool(s) offer the features I need?

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 a robust set of additional capabilities that you can use to learn more from your data.

If you anticipate needing to graph different curves that have drastically different y-scales on the same chart, you need to ensure that your graphing tool has the capability to distinguish between these scales. Many tools have this capability but also have a limited maximum y-axis count.

In addition, you should consider your visualization needs that 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 the form of a 3D graph, then your visualization tool must support these capabilities.

Can I tailor my visualization to my application's needs?

The scalability and customizability of visualization is important to consider. Because every engineering measurement application is different – from containing different measurement types to having different goals – a visualization tool should be flexible enough to be tailored to the needs of your application. Out-of-the-box tools that incorporate data acquisition and visualization in a closed package often impose rigid limitations on the type of visualization offered. While this may be suitable initially, you may need a more extensive look into acquired data to make informed decisions; this requires that more data be graphed per curve, more curves be plotted per graph, and more graphs be viewed. Or it may require even simply zooming, scrolling, and scaling tools that many visualization packages restrict. If you expect your visualization needs to grow as your application expands, make sure you choose a visualization tool that scales as well.

Do I need to visualize measurement data synchronized with data from other sources?

Advanced engineering data postprocessing tools feature synchronized visualization that extends beyond simple spreadsheets and static graphs. In addition to zooming and scrolling graph axes, cursors between graphs can be synchronized – usually using a common timebase – to correlate information viewed between one graph and another. For example, cursors on a graph may allow an engineer to specify a beginning and ending x-region subset of a curve that is used to dynamically calculate and display the fast Fourier transform (FFT) of the data in the resulting region. That band can then be panned back and forth along the superset of data, extended, or reduced to isolate the region of interest.

In additional to synchronizing graphs of measurement data with other graphs of data or resultant calculations, advanced tools can synchronize measurement data on graphs with data from other sources such as video, sound, 3D models, or GPS. By correlating measurement data with information from these other sources that often provide even more of an engineering context than simple curves on graphs, you are empowered to discover more from your measurement investment. For example, these synchronization tools enable advanced visualization capabilities that play back measurement data linked to video so you can see what happened during a measurement, sound so you can hear what happened, or GPS so you can determine where something happened.

▷ Read more about data visualization in NI LabVIEW

How to Choose the Right Data Storage Format for Your Measurement System

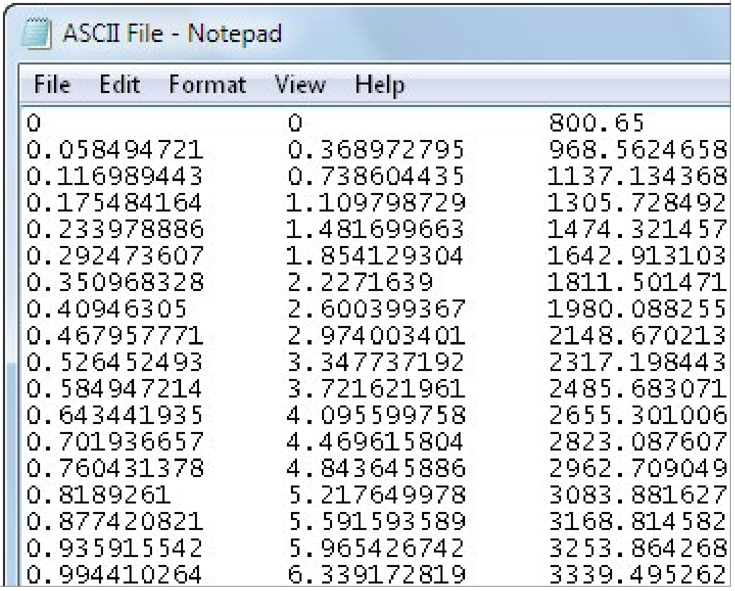
Overview

For many new measurement systems, choosing the right data storage approach is an afterthought. Engineers often end up selecting the storage strategy that most easily meets the needs of the application in its current state without considering future requirements. Yet storage format choices can have a large impact on the overall efficiency of the acquisition system as well as the efficiency of postprocessing the raw data over time.

Today's complex products require data acquisition throughout the complete design and development process. Companies make a significant investment in the data they collect. Increasing microprocessor speed and storage capacity together with decreasing costs for hardware and software have resulted in an explosion of data stored in files and databases. While technology is enabling faster and richer data retention, managing and making good use of this data remains the real challenge. This white paper covers five things to consider when choosing your data storage format.

Can my data storage format handle my data volume and data streaming speeds?

The first step to achieving a cohesive data management solution is ensuring that data is stored in the most efficient, organized, and scalable fashion. All too often data is stored without descriptive information, in inconsistent formats, and scattered about on arrays of computers, which creates a graveyard of information that makes it extremely difficult to locate a particular data set and derive decisions from it.



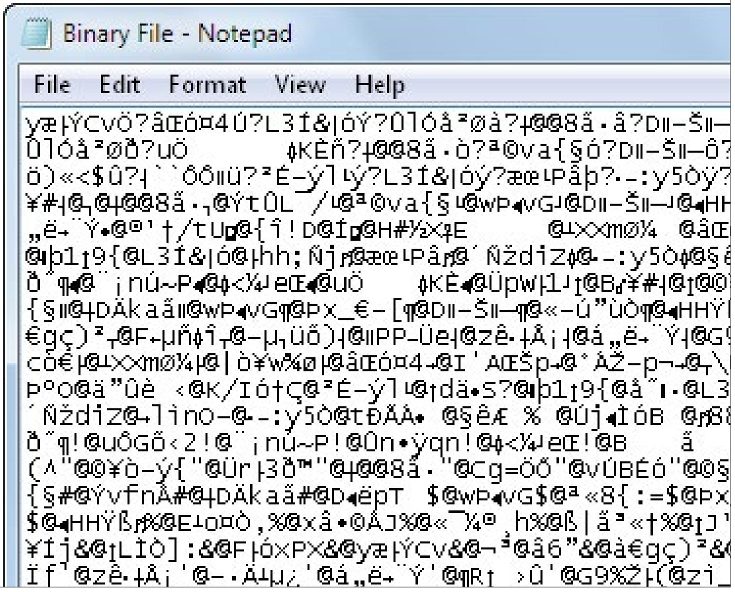
```
0               0               800.65
0.058494721     0.368972795     968.5624658
0.116989443     0.738604435     1137.134368
0.175484164     1.109798729     1305.728492
0.233978886     1.481699663     1474.321457
0.292473607     1.854129304     1642.913103
0.350968328     2.2271639       1811.501471
0.40946305      2.600399367     1980.088255
0.467957771     2.974003401     2148.670213
0.526452493     3.347737192     2317.198443
0.584947214     3.721621961     2485.683071
0.643441935     4.095599758     2655.301006
0.701936657     4.469615804     2823.087607
0.760431378     4.843645886     2962.709049
0.8189261       5.217649978     3083.881627
0.877420821     5.591593589     3168.814582
0.935915542     5.965426742     3253.864268
0.994410264     6.339172819     3339.495262
```

Figure 1. ASCII files are easy to exchange but can be too slow and large for many applications.

Depending on the application, you may prioritize certain characteristics over others. Common storage formats such as ASCII, binary, and XML have strengths and weaknesses in different areas.

ASCII Files

Many engineers prefer to store data using ASCII files because of the format's easy exchangeability and human readability. ASCII files, however, have several drawbacks, including a large disk footprint, which can be an issue when storage space is limited (for example, storing data on a distributed system). Reading and writing data from an ASCII file can be significantly slower compared to other formats. In many cases, the write speed of an ASCII file cannot keep up with the speeds of acquisition systems, which can lead to data loss.



```
yæ¦ŸCvÖ?âŒó¤4Ü?L3Í&¦óŸ?Û1óå²øå?‡@@8ã·â?D‖–Š‖–
Û1óå²øð?uÖ        ¢KÈñ?‡@@8ã·ò?ª@va{§ó?D‖–Š‖–ô?
ö)«<$û?¦``ôÔ‖ü?²É–ý1Ly?L3Í&¦óý?æœLPåþ?-–:y5Òý?
¥#¦@¬@‡@@8ã·¬@ÝtÛL /Ľ@ª@va{§Ľ@wP◂vGɺ@D‖–Š‖–┘@◂HH
„ë← Ÿ•@º'†/tU@{î!D@Í@H#½×‡E       @┴××m@¾ @âŒ
@‡þ1↕9{@L3Í&¦ó@¦hh;Ñj@æœLPå@´Ñždîz¢@--:y5Ò¢@§é
ð¨¶@¨¡nú~P◂@¢<¾┘eŒ◂@uÖ     ¢KÈ◂@ÜpW├1┘↕@B/¥#¦@↕@@
{§‖@┤DÄkaå‖@wP◂vG¶@Px_€–[¶@D‖–Š‖–¶@«–ú"ùÒ¶@◂HHŸ
€gç)²┬@F←µñ¢î┬@–µ┬üõ)¦@‖PP–Üe¦@zê·┼Å¡¦@á„ë← Ÿ¦@G
cô€µ@┴××m@¾µ@|ö¥w%øµ@âŒó¤4–@I'AŒŠp┬@˚ÀŽ–p¬¬@┬\
pºo@ä"ûè <@K/Ióf Ç@²É–ý1@↕dä•S?@þ1↕9{@å˜ı·@L3
´Ñždîz@┬1ino–@–:y5Ò@tÐÄÄ• @§êÆ % @Új◂ìóB @ñ8
ð˜¶!@uÕGõ‹2!@¨¡nú~P!@Ûn•ÿqn!@¢<¾┘eŒ!@B     ã
(^"@©¥ò–ý{"@Ür├3ð™"@‡@@8ã·"@Cg=ÖÕ"@vÚBÉÖ"@@§
{§#@ŸvfnÅ#@┤DÄkaå#@D◂ëpT  $@wP◂vG$@ª«8{:=$@Px
$@◂HHŸß@%@E┴¤¤Ò,%@xâ•@ÅJ%@«¯¼º¸h%@ß|ä²«†%@↕J'
¥Íj&@↕LÍÒ]:&@F├óXPX&@yæ¦ŸCv&@¬³@â6"&@à€gç)²&@
Ïf'@zê·┼Å¡'@–·Ä┴µ¿'@á„ë← Ÿ'@¶R↑ ›û'@G9%Ž├(@zî_
```

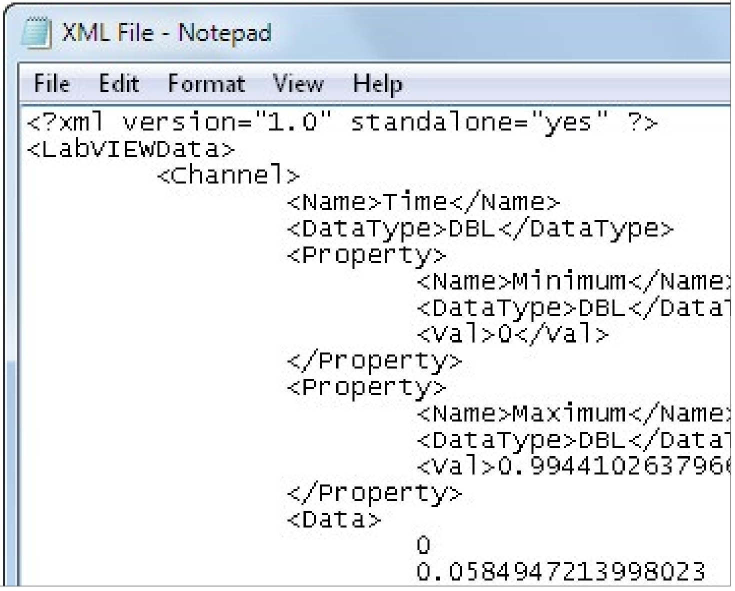
Figure 2. Binary files are beneficial in high-speed, limited-space applications but can cause exchangeability issues.

Binary Files

Another typical storage approach that is somewhat on the opposite end of the spectrum from ASCII is binary files. In contrast to ASCII files, binary files feature a significantly smaller disk footprint and can be streamed to disk at extremely high speeds, making them ideal for high-channel-count and real-time applications. A drawback to using binary is its unreadable format that complicates exchangeability between users.

XML Files

Over the last several years, the XML format has been gaining in popularity due to its ability to store complex data structures. With XML files, you can store data and formatting along with the raw measurement values. Using the flexibility of the XML format, you can store additional information with the data in a structured manner. XML is also relatively human-readable and exchangeable. Similar to ASCII,



```
<?xml version="1.0" standalone="yes" ?>
<LabVIEWData>
        <Channel>
                <Name>Time</Name>
                <DataType>DBL</DataType>
                <Property>
                        <Name>Minimum</Name
                        <DataType>DBL</Data
                        <Val>0</Val>
                </Property>
                <Property>
                        <Name>Maximum</Name
                        <DataType>DBL</Data
                        <Val>0.99441026379
                </Property>
                <Data>
                        0
                        0.0584947213998023
```

Figure 3. XML files can help you define complex structures but are significantly larger and slower than other formats.

XML files can be opened in many common text editors as well as XML-capable Internet browsers, such as Microsoft Internet Explorer.

Database Files

Database files are composed of a series of tables, built using columns and rows, and information may or may not be linked between tables. Though databases are advantageous because they are searchable, database files are impractical for time-based measurement applications given the amount of data acquired and the need to either purchase or build a formal database solution from scratch.

TDMS Files

TDMS is a binary-based file format, so it has a small disk footprint and can stream data to disk at high speeds. At the same time, TDMS files contain a header component that stores descriptive information, or attributes, with the data. Some attributes such as file name, date, and file path are stored automatically; however, you can easily add your own custom attributes as well. Another advantage of the TDMS file format is the built-in three-level hierarchy: file, group, and channel levels. A TDMS file can contain an unlimited number of groups, and each group can contain an unlimited number of channels. You can add attributes at each of these levels describing and documenting your test data for better understanding. This hierarchy creates an inherent organization of your test data.

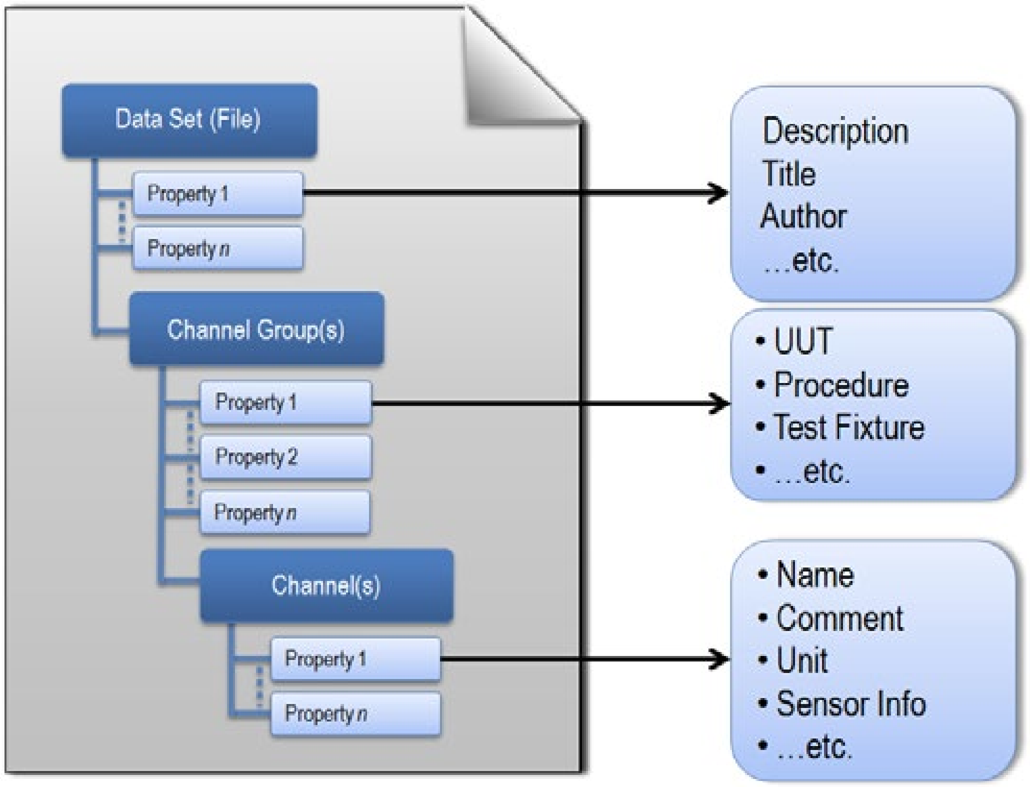


Figure 4. The hierarchical TDMS file format is designed to meet the needs of engineers collecting measurement data.

Do I need to exchange data with my colleagues?

Often simulation and test systems grow over several years, usually independently of each other and with equipment from different suppliers. As a result, data is stored in different file formats, with little or no descriptive information and in different locations. All of these factors present roadblocks that impede the optimal exchange of information and make it extremely difficult to locate a particular data set and derive decisions from it. As a result, many companies are seeing a loss in efficiency and increasing development costs as the amount of data they store increases.

ASCII Files

Many engineers prefer to store data using ASCII files because of the format's easy exchangeability and human readability. ASCII files make it simple to quickly open files written from acquisitions and view data immediately as well as to easily share the data with colleagues because the files can be opened in common software applications found on most computers today such as Notepad, WordPad, and Microsoft Excel.

Binary Files

The shortcoming of saving data to a custom binary file format is that it is not human-readable and therefore difficult to exchange between users. Binary files cannot be immediately opened by common software; they have to be interpreted by a specialized application or program. Different applications may interpret binary data in different ways, which causes confusion. To share the files with colleagues, you must provide them with an application that interprets your specific binary file correctly. Also, if you make changes to how the data is written in the acquisition application, these changes must be made in the application that is reading data. This can potentially cause long-term application versioning issues and headaches that can ultimately result in lost data.

XML Files

In its raw form, XML includes tags within the file that describe the structures. These tags also appear when XML files are opened in these applications, which somewhat limits the readability because you must be able to understand these tags. This may be a challenge for colleagues less involved in the file architecture development. Also, a downside to being able to store these complex structures is that they may require considerable planning when you design the layout, or schema, of the XML structures.

Database Files

Database files can be shared among team members and usually opened via common applications. With measurement data, they have limitations similar to ASCII and XML files: large footprint, risk of lost data, and limited streaming to disk.

TDMS Files

Although TDMS files are binary, you can open them in many common applications, such as Microsoft Excel and OpenOffice Calc, for sharing with colleagues. TDMS files give you the benefits of easy exchangeability and attribute inclusion without sacrificing speed and size.

Can my analysis or visualization software use my storage format?

When choosing a file format, you need to know which formats your analysis or visualization software can accept.

ASCII Files

ASCII fi can be opened by nearly every data analysis and visualization package on the market, but, as is typical of the format, they are limited in their disk read speeds. Processing and visualizing ASCII files is drastically slower than their binary equivalent.

Binary Files

Custom binary files need to be changed and updated over time, which means that analysis and visualization software needs to accommodate these version changes. When application software is updated to interpret the binary file, analysis and visualization speeds are radically improved using the binary format since the software doesn't need to handle ASCII overhead.

XML Files

XML is built on a tree model and, therefore, mapping the trees to the analysis or visualization API is required. This mapping can be difficult and complex, potentially causing delays and introducing errors into your project. In addition, errors you encounter while processing an XML file can disrupt your postprocessing and delay your project.

Database Files

Database files which are constructed in a structured fashion similar to XML files face the same challenges in terms of mapping the content and being increasingly complicated as your data set gets larger and more diverse. Application software must be able to interface with the database to read and potentially convert the data sets before analysis or visualization.

TDMS Files

Since TDMS is a binary file format, analyzing or visualizing data from a TDMS file is extremely efficient. TDMS is an open standard that can use plug-ins or a free DLL to load TDMS data into a variety of different software packages.

Is my storage format flexible enough for my future needs?

You can choose from a variety of format options for measurement data storage. Unfortunately, careful consideration of data storage options is not typically at the forefront of application planning. The file format choice is often overlooked in favor of higher-visibility decisions such as hardware system design or software architecture. Data storage decisions are sometimes made arbitrarily or on an as-needed, per-application basis without second thought for reusability and scalability. This leads to complex and costly software rearchitecture. You should carefully consider the scalability of your data storage format choice to ensure reuse for future requirement changes.

How do I retrieve my stored data at a future date?

More than ever in today's fiercely competitive business environment, companies need to rapidly turn test and simulation data into usable information to efficiently drive product development and reduce time to market. In many acquisition systems, the application is designed to quickly collect the raw data and store it to disk, possibly with some inline processing for decision making. However, to truly move from the raw data to usable results, you need tools to quickly find trends and outliers in the data, analyze these trends, and report them in a way that can be shared with others.

Only database and TDMS files are searchable without having to open each file and manually search for the data you need. Depending on how a database file is structured, the search process can be slow. In addition, the database files you want to search need to be in the same location and structured exactly the same. With diverse teams across the world, implementing a strict database file structure can be virtually impossible as every engineer, department, and company already has its own data management policy that is likely different from one another.

In contrast, TDMS files are searchable and do not need to be formatted and stored in precisely the same location. In fact, the more custom properties you use to document your measurement data, the more easily you can locate it at a later date. The descriptive information located in the file, a key benefit of TDMS, provides an easy way to document the data without you having to design your own header structure. As your documentation requirements increase, you do not have to redesign your application; you simply extend the model to meet your specific needs.

Summary

	ASCII	Binary	XML	Database	TDMS
Exchangeable	✓		✓		✓
Small Disk Footprint		✓			✓
Searchable				✓	✓
Inherent Attributes			✓		✓
High-Speed Streaming		✓			✓
NI Platform Supported	✓	✓	✓	✓*	✓

*May require a toolkit or add-on module.

Table 1. The TDMS file format combines the benefits of several data storage options in one file format.

Several roadblocks can impede the optimal exchange of technical information. The most notorious is the improper storage of information at the time of test or simulation. You should always make sure that your data is stored with descriptive information, in consistent formats, and easily locatable. Table 1 summarizes the pros and cons of some of the most commonly chosen storage options for measurement data.

▷ Read about data management and mining with NI DIAdem

How to Choose the Right Reporting Tools for Your Measurement System

Overview

Data is acquired for a reason: to make decisions from information derived from raw data. While technology is enabling faster and richer data retention, storing, managing, and sharing data remain a real challenge. The goal of most data acquisition (DAQ) systems is to collect data for analysis that is ultimately presented or shared in the form of an exchangeable report. You can choose from a wealth of reporting options, but you should carefully consider the capabilities of your reporting tool of choice to ensure that it doesn't become a bottleneck in your system. This white paper outlines five things to consider when choosing a reporting tool for your application.

Can my reporting tool(s) handle my data?

Once you have pinpointed a storage format, you must make sure that your desired reporting tool is compatible with the data format. This involves examining two key factors: file format and data volume. A reporting tool must be able to not only load data from a chosen file format but also handle the volume of data saved.

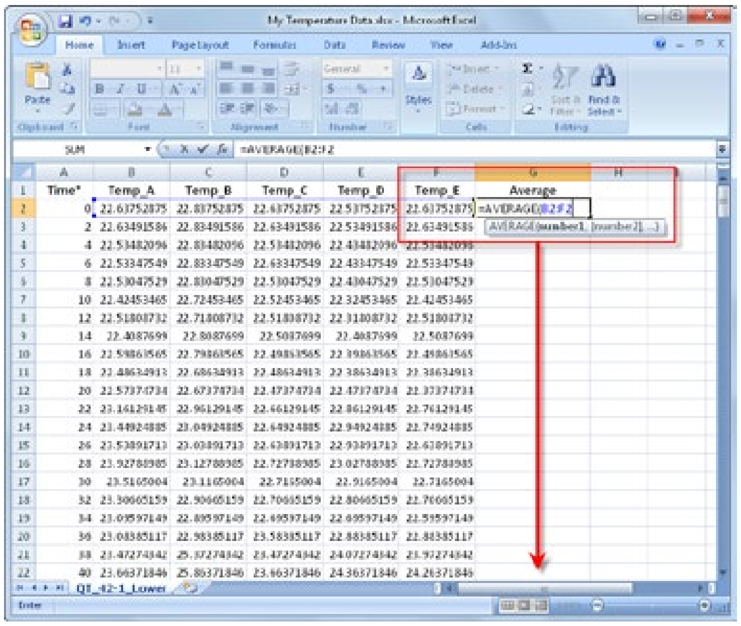
File Format

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

The Technical Data Management Streaming (TDMS) file format meets the specific needs and high demands of engineers and scientists. 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 offers ease of use, high-speed streaming, and exchangeability.

Traditional financial analysis tools that are often used for engineering reporting 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.



=AVERAGE(B2:F2)

AVERAGE(number1, [number2], ...)

	A	B	C	D	E	F	G	H
1	Time*	Temp_A	Temp_B	Temp_C	Temp_D	Temp_E	Average	
2	0	22.61752875	22.81752875	22.61752875	22.51752875	22.61752875	=AVERAGE(B2:F2)	
3	2	22.63491586	22.83491586	22.63491586	22.53491586	22.63491586		
4	4	22.53482096	22.83482096	22.53482096	22.43482096	22.53482096		
5	6	22.53347549	22.83347549	22.63347549	22.43347549	22.53347549		
6	8	22.53047529	22.83047529	22.53047529	22.43047529	22.53047529		
7	10	22.42453465	22.72453465	22.52453465	22.32453465	22.42453465		
8	12	22.51808732	22.71808732	22.51808732	22.31808732	22.51808732		
9	14	22.4087699	22.8087699	22.5087699	22.4087699	22.5087699		
10	16	22.59861565	22.79861565	22.49861565	22.39861565	22.49861565		
11	18	22.48634913	22.68634913	22.48634913	22.38634913	22.38634913		
12	20	22.57374734	22.67374734	22.47374734	22.47374734	22.37374734		
13	22	23.16129145	22.96129145	22.66129145	22.86129145	22.76129145		
14	24	23.44924885	23.04924885	22.64924885	22.94924885	22.74924885		
15	26	23.53891713	23.03891713	22.63891713	22.93891713	22.63891713		
16	28	23.92788985	23.12788985	22.72788985	23.02788985	22.72788985		
17	30	23.5165004	23.1165004	22.7165004	22.9165004	22.7165004		
18	32	23.30665159	22.90665159	22.70665159	22.80665159	22.70665159		
19	34	23.09597149	22.89597149	22.49597149	22.69597149	22.59597149		
20	36	23.08385117	22.98385117	23.58385117	22.88385117	22.88385117		
21	38	23.47274342	25.37274342	23.47274342	24.07274342	23.97274342		
22	40	23.66371846	25.86371846	23.66371846	24.36371846	24.26371846		

Figure 1. 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).

DAQ applications collecting dozens of data channels at megasample-per-second (MS/s) rates are also commonplace. In these applications, data manipulation and interaction are implemented on a signal – or channel – as a whole. When manipulating columns of individual cells, the unity of a signal can be lost.

Though entire columns can be manipulated at a time, this is cumbersome. Columns also often contain descriptive information such as a name or unit in addition to the raw numeric data. In this case, you must select a subset of the column (for example, the range A3:A999), which introduces overhead and the potential for inaccuracy or errors.

In Figure 1, Microsoft Excel is used to perform a simple but common engineering task: averaging five temperature channels stored in columns to create a resultant average channel. The averaging calculation must first be implemented with the building block of a cell and then copied (or filled) to all cells in the resultant column.

Data Volume

These days, the data streaming speeds of common applications often reach or exceed MS/s rates. In an application that collects one single channel of data at 1 MS/s, a total of 1,000,000 data points are collected in a one-second acquisition. In a matter of minutes, billions of data points can be saved to gigabytes of hard drive space.

When traditional reporting tools attempt to load a data file containing a large volume of data, they try to load every single data point into memory. Loading the entirety of a large data set into these tools can often take many minutes due to the sheer volume of data that needs to be loaded. 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 engineers to readjust their storage strategy by either choosing a new file format (possibly having to rearchitect their applications late in the game) or segmenting data into many small files just so that their reporting tools can open them.

When designing a DAQ system, you should be sure that your chosen reporting tool can handle your chosen 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 reporting tool(s) offer the visuals that I need?

When it comes to reporting, most engineers require, at a minimum, basic charting and graphing capabilities. Luckily, almost every reporting tool on the market can make simple charts and graphs. However, you should 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 foresee needing to graph different curves that have drastically different y-scales on the same chart, you need to ensure that your reporting tool has the capability to distinguish between these scales. Many tools have this capability, but they also have a maximum y-axis count that is limited.

In addition, you should consider your reporting needs that go beyond basic 2D graphing. For example, if you have a need to represent data using polar plots, or if your data would be best represented in the form of a 3D graph, then your reporting tool must support these capabilities.

Can I use templates to simplify repetitive reports?

Many times your reporting needs involve applying the same type of report to a series of raw data files. For example, if you run the same types of tests every week and have to report results in a standardized fashion, you wind up reusing the same report layout across multiple data sets. In traditional reporting tools, the report display is saved 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 winds up containing 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 have to edit every fi
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 should seek out a reporting tool that you can use to produce a report template and apply it to different raw data files.

Can I automate reporting to save some time?

Typically, a DAQ application uses one of two types of reporting scenarios: infrequent and repetitive reporting. Infrequent reporting is implemented on an occasional basis – usually in such a way that is interactive and customized to the needs of the moment. Alternatively, repetitive reporting is done in a frequent and usually standardized fashion and often uses templates.

If your reporting needs are repetitive, you should seek out a reporting tool that provides the capabilities to automate reporting. Even most traditional tools support the development of macros or scripts to make this easier. Many have recording modes so you can interactively record scripts that automate lengthy evaluations or calculations.

Does my reporting tool(s) export reports in the right format?

The final output of a reporting tool is usually some sort of easily exchangeable format that can be emailed, printed, or presented regardless of the original format of the raw data files. Most reporting tools support the export of reports to several formats, but you should ensure that your most commonly used report formats exist – for example, PDF files, PowerPoint slides, imagery, or HTML files.

Additionally, if your reporting needs are vast – for example, your reports often span dozens of pages – you should ensure that the reporting tool you choose can export your report in your desired format and 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 of choice can't create reports at your required length.

▷ Learn how NI DIAdem can help you manage and report on your measurement data