Data Acquisition Fundamentals

Introduction

Today, most scientists and engineers use personal computers (PCs) with PCI, PXI/CompactPCI, PCMCIA, USB, IEEE 1394, ISA, or parallel or serial ports for data acquisition in laboratory research, test and measurement, and industrial automation. Many applications use plug-in devices to acquire data and transfer it directly to computer memory. Others use data acquisition (DAQ) hardware remote from the PC that is coupled via parallel or serial port. Obtaining proper results from a PC-based DAQ system depends on each of the following system elements (see Figure 1):

- The PC
- Transducers
- Signal Conditioning
- DAQ Hardware
- Software

This document gives an overview of each element and explains the most important criteria of each element. The document also defines much of the terminology common to each element of a PC-based DAQ system.

Figure 1. The Typical PC-Based DAQ System
The Personal Computer

The computer you use for your DAQ system can drastically affect the maximum speeds at which you can continuously acquire data. Today's technology boasts Pentium and PowerPC class processors coupled with the higher performance PCI, PXI/CompactPCI, and IEEE 1394 (FireWire) bus architectures as well as the traditional ISA bus and USB. The PCI bus and a USB port are standard equipment on most of today's desktop computers, though the ISA bus has become less common. With the advent of PCMCIA, USB, and IEEE 1394, portable data acquisition offers a flexible alternative to desktop PC-based DAQ systems. For remote DAQ applications that use RS-232 or RS-485 serial communication, your data throughput is usually limited by the serial communication rates. When choosing a DAQ device and bus architecture, keep in mind the data transfer methods supported by your chosen device and bus.

The data transfer capabilities of your computer can significantly affect the performance of your DAQ system. All PCs are capable of programmed I/O and interrupt transfers. Direct memory access (DMA) transfers, implemented on almost all of today's personal computers, increase the system throughput by using dedicated hardware to transfer data directly into system memory. Using this method, the processor is not burdened with moving data and is therefore free to engage in more complex processing tasks. To reap the benefits of DMA or interrupt transfers, your DAQ device must be capable of these transfer types. For example, while PCI, ISA, and IEEE 1394 devices offer both DMA and interrupt-based transfers, PCMCIA and USB devices use only interrupt-based transfers. The chosen transfer method will affect the achievable throughput of your DAQ device.

The limiting factor for acquiring large amounts of data often is the hard drive. Disk access time and hard drive fragmentation can significantly reduce the maximum rate at which data can be acquired and streamed to disk. For systems that must acquire high-frequency signals, select a high-speed hard drive for your PC and ensure that there is enough contiguous (unfragmented) free disk space to hold the data. In addition, dedicate a hard drive to the acquisition and run the operating system (OS) on a separate disk when streaming data to disk.

Applications requiring real-time processing of high-frequency signals need a high-speed, 32-bit processor with its accompanying coprocessor or a dedicated plug-in processor such as a digital signal processing (DSP) board. If the application only acquires and scales a reading once or twice a second, however, a low-end PC can be satisfactory.

Determine which operating system and computer platform will yield the greatest long-term return on investment while still meeting your short-term goals. Factors that may influence your choice may include the experience and needs of both your developers and end users, other uses for the PC (now and in the future), cost constraints, and the availability of different computers with respect to your implementation time frame. Traditional platforms include Mac OS, which is known for its simple graphical user interface, and Windows 9x. In addition, Windows NT 4.0 and Windows 2000 offer a more robust 32-bit OS with the look and feel of Windows 9x. Windows 2000 is the next generation of the Windows NT OS that combines the best features of both Windows NT and Windows 9x. These features include native Plug and Play and power management.

Transducers and Signal Conditioning

Transducers sense physical phenomena and produce electrical signals that the DAQ system measures. For example, thermocouples, resistance temperature detectors (RTDs), thermistors, and IC sensors convert temperature into an analog signal that an analog-to-digital converter (ADC) can measure. Other examples include strain gauges, flow transducers, and pressure transducers, which measure force, rate of flow, and pressure, respectively. In each case, the electrical signals produced are proportional to the physical parameters they monitor.

The electrical signals generated by the transducers must be optimized for the input range of the DAQ device. Signal conditioning accessories amplify low-level signals and then isolate and filter them for more accurate measurements. In addition, some transducers use voltage or current excitation to generate a voltage output. Figure 2 depicts a typical DAQ system with National Instruments SCXI signal conditioning accessories.
Signal conditioning accessories can be used in a variety of important applications:

- **Amplification** – The most common type of conditioning is amplification. Low-level thermocouple signals, for example, should be amplified to increase the resolution and reduce noise. For the highest possible accuracy, the signal should be amplified so that the maximum voltage range of the conditioned signal equals the maximum input range of the ADC.

  For example, SCXI has several signal conditioning modules that amplify input signals. The gain is applied to the low-level signals within the SCXI chassis that are located near the transducers, so the module sends only high-level signals to the PC, minimizing the effects of noise on the readings.

- **Isolation** – Another common signal conditioning application is isolating the transducer signals from the computer for safety purposes. The system being monitored may contain high-voltage transients that could damage the computer without signal conditioning.

  An additional reason for isolation is ensuring that the readings from the plug-in DAQ device are unaffected by differences in ground potentials or common-mode voltages. When the DAQ device input and the signal being acquired are each referenced to “ground,” problems occur if there is a potential difference in the two grounds. This difference can lead to what is known as a ground loop, which may cause inaccurate representation of the acquired signal; or if the difference is too large, it may damage the measurement system. Using isolated signal conditioning modules eliminates ground loops and ensures that the signals are accurately acquired. For example, the SCXI-1120 and SCXI-1121 modules provide isolation up to 250 Vrms of common-mode voltage. The SCXI-1122 module provides up to 450 Vrms isolation.

- **Multiplexing** – A common technique for measuring several signals with a single measuring device is multiplexing. Signal conditioning hardware for analog signals often provides multiplexing for use with slowly changing signals like temperature. The ADC samples one channel, switches to the next channel, samples it, switches to the next channel, and so on. Because the same ADC samples many channels instead of one, the effective sampling rate of each individual channel is inversely proportional to the number of channels sampled.

  For example, a PCI-MIO-16E-1 sampling at 1 MS/s on 10 channels will effectively sample each individual channel at:

  \[
  \frac{1 \text{ MS/s}}{10 \text{ channels}} = 100 \text{ kS/s per channel}
  \]
SCXI modules for analog signals employ multiplexing so that as many as 3,072 signals can be measured with one DAQ device. With the AMUX-64T analog multiplexer, you can measure up to 256 signals with a single device. This feature is in addition to any built-in multiplexing on the DAQ device.

- **Filtering** – The purpose of a filter is to remove unwanted signals from the signal that you are trying to measure. A noise filter is used on DC-class signals, such as temperature, to attenuate higher frequency signals that can reduce your measurement accuracy. For example, many SCXI modules use 4 Hz and 10 kHz lowpass filters to eliminate noise before the signals are digitized by the DAQ device.

  AC-class signals, such as vibration, often require a different type of filter known as an antialiasing filter. Like the noise filter, the antialiasing filter is also a lowpass filter; however, it requires a very steep cutoff rate, so it almost completely removes all signal frequencies that are higher than the input bandwidth of the device. If the signals were not removed, they would erroneously appear as signals within the input bandwidth of the device. Devices designed specifically for AC-class signal measurement – the NI 455x, NI 445x, and NI 447x dynamic signal acquisition (DSA) devices, the NI 6115 simultaneous-sampling multifunction I/O devices, and the SCXI-1141 module have antialiasing filters built into them.

- **Excitation** – Signal conditioning also generates excitation for some transducers. Strain gauges, thermistors, and RTDs, for example, require external voltage or current excitation signals. Signal conditioning modules for these transducers usually provide these signals. RTD measurements are usually made with a current source that converts the variation in resistance to a measurable voltage. Strain gauges, which are very low-resistance devices, typically are used in a Wheatstone bridge configuration with a voltage excitation source. The SCXI-1121 and SCXI-1122 have onboard excitation sources, configurable as current or voltage, that you can use for strain gauges, thermistors, or RTDs.

- **Linearization** – Another common signal conditioning function is linearization. Many transducers, such as thermocouples, have a nonlinear response to changes in the phenomena being measured. NI makes NI-DAQ, LabVIEW, Measurement Studio, and VirtualBench, which are application software packages that include linearization routines for thermocouples, strain gauges, and RTDs.

You should understand the nature of your signal, the configuration that is being used to measure the signal, and the affects of the environment surrounding the system. Based on this information, you can determine whether signal conditioning will be a necessary part of your DAQ system.

## DAQ Hardware

### Analog Inputs

**Basic Considerations of Analog Inputs** – The analog input specifications give you information on both the capabilities and the accuracy of the DAQ product. Basic specifications, which are available on most DAQ products, tell you the number of channels, the sampling rate, the resolution, and the input range.

- **Number of Channels** – The number of analog channel inputs is specified for both single-ended and differential inputs for devices with both input types. Single-ended inputs are all referenced to a common ground reference. These inputs are typically used when the input signals are high level (greater than 1 V), the leads from the signal source to the analog input hardware are short (less than 15 ft.), and all input signals share a common ground reference. If the signals do not meet these criteria, you should use differential inputs. With differential inputs, each input has its own ground reference; noise errors are reduced because the common-mode noise picked up by the leads is canceled out.

- **Sampling Rate** – This parameter determines how often conversions can take place. A faster sampling rate acquires more data in a given time and can therefore often form a better representation of the original signal.

- **Multiplexing** – A common technique for measuring several signals with a single ADC is multiplexing. For more information on multiplexing, refer to the *Signal Conditioning* section of this document.
• **Resolution** – The number of bits that the ADC uses to represent the analog signal is the resolution. The higher the resolution, the larger the number of divisions the range is broken into, and therefore, the smaller the detectable voltage change. Figure 3 shows a sine wave and its corresponding digital image as obtained by an ideal 3-bit ADC. A 3-bit converter (which is actually seldom used but a convenient example) divides the analog range into 23, or 8 divisions.

Each division is represented by a binary code between 000 and 111. Clearly, the digital representation is not a good representation of the original analog signal because information has been lost in the conversion. By increasing the resolution to 16 bits, however, the number of codes from the ADC increases from 8 to 65,536, and you can therefore obtain an extremely accurate digital representation of the analog signal if the rest of the analog input circuitry is properly designed.

![Figure 3. Digitized Sine Wave with a Resolution of Three Bits](image)

• **Range** – Range refers to the minimum and maximum voltage levels that the ADC can quantize. NI multifunction DAQ devices offer selectable ranges so that the device is configurable to handle a variety of voltage levels. With this flexibility, you can match the signal range to that of the ADC to take advantage of the available measurement resolution.

• **Code Width** – The range, resolution, and gain available on a DAQ device determine the smallest detectable change in voltage. This change in voltage represents 1 least significant bit (LSB) of the digital value and is often called the code width. The ideal code width is found by dividing the voltage range by the gain times two raised to the order of bits in the resolution. For example, one of our 16-bit multifunction DAQ device, the NI 6030E, has a selectable range of 0 to 10 or −10 to 10 V and selectable gain of 1, 2, 5, 10, 20, 50, or 100. With a voltage range of 0 to 10 V, and a gain of 100, the ideal code width is defined by the following equation:

\[
\frac{10}{100 \times 2^{16}} = 1.5 \mu V
\]

• **Critical Considerations of Analog Inputs** – Although the basic specifications previously described may show that a DAQ device has a 16-bit resolution ADC and a 100 kS/s sampling rate, you may not sample at full speed on all 16 channels and get full 16-bit accuracy. For example, some products on the market today with 16-bit ADCs get less than 12 bits of useful data. To determine if the device that you are considering will give you the desired results, scrutinize specifications that go beyond the product resolution.

While evaluating DAQ products, also consider the differential nonlinearity (DNL), relative accuracy, settling time of the instrumentation amplifier, and noise.

• **DNL** – Ideally, as you increase the level of voltage applied to a DAQ device, the digital codes from the ADC should also increase linearly. If you were to plot the voltage versus the output code from an ideal ADC, the plot would be a straight line. Deviations from this ideal straight line are specified as the nonlinearity. DNL is a measure in LSB of the worst-case deviation of code widths from their ideal value of 1 LSB. An ideal DAQ device has a DNL of 0 LSB. Practically, a good DAQ device will have a DNL of ±0.5 LSB.
There is no upper limit on how wide a code can be. Codes are not smaller than 0 LSB, so the DNL is never worse than –1 LSB. A DAQ device with poor performance may have a code width equal to or very near zero, which indicates a missing code. No matter what voltage you input to a DAQ device with a missing code, the device will never quantize the voltage to the value represented by this code. Sometimes DNL is specified by stating that a DAQ device has no missing codes, which means that the DNL is bounded below by –1 LSB but does not make any specifications about the upper boundaries. All National Instruments E Series devices are guaranteed to have no missing codes, and the specifications clearly state the DNL specification so that you know the device linearity.

If the DAQ device in the previous example, which had a code width of 1.5 µV, had a missing code slightly above 500 µV, then increasing the voltage to 502 µV would not be detectable. Only when the voltage is increased another LSB, or in this example, beyond 503 µV, will the voltage change be detectable. Poor DNL reduces the resolution of the device.

- **Relative Accuracy** – Relative accuracy is a measure in LSBs of the worst-case deviation from the ideal DAQ device transfer function, a straight line. Relative accuracy is determined on a DAQ device by connecting a voltage at negative full scale, digitizing the voltage, increasing the voltage, and repeating the steps until the input range of the device has been covered. When the digitized points are plotted, the result will be an apparent straight line, as shown in Figure 4(a). However, you can subtract actual straight-line values from the digitized values and plot these resulting points, as shown in Figure 4(b). The maximum deviation from zero is the relative accuracy of the device.

![Figure 4](image)

**Figure 4.** Determining the relative accuracy of a DAQ device. Figure 4(a) shows the apparent straight-line plot generated by sweeping the input. Figure 4(b) shows, by subtracting out calculated straight-line values, that the plot is actually not straight.

The driver software for a DAQ device translates the binary code value of the ADC to voltage by multiplying it by a constant. Good relative accuracy is important for a DAQ device because it ensures that the translation from the binary code of the ADC to the voltage value is accurate. Obtaining good relative accuracy requires that both the ADC and the surrounding analog circuitry are properly designed.

- **Settling Time** – Settling time is the time required for an amplifier, relays, or other circuits to reach a stable mode of operation. The instrumentation amplifier is most likely not to settle when you are sampling several channels at high gains and high rates. Under such conditions, the instrumentation amplifier has difficulty tracking large voltage differences that can occur as the multiplexer switches between input signals. Typically, the higher the gain and the faster the channel switching time, the less likely the instrumentation amplifier is to settle. In fact, no off-the-shelf programmable-gain instrumentation amplifier can settle to 12-bit accuracy in less than 2 µs when amplifying at a gain of 100. NI developed the NI-PGIA specifically for DAQ applications, so devices that use the NI-PGIA can consistently settle at high gains and sampling rates.
• **Noise** – Any unwanted signal that appears in the digitized signal of the DAQ device is noise. Because the PC is a noisy digital environment, acquiring data on a plug-in device takes a careful layout on multiple-layer DAQ devices by skilled analog designers. Simply placing an ADC, instrumentation amplifier, and bus interface circuitry on a one or two-layer board will likely result in a very noisy device. Designers can use metal shielding on a DAQ device to help reduce noise. Proper shielding should not only be added around sensitive analog sections on a DAQ device, but must also be built into the layers of the device with ground planes.

Figure 5 shows a DC noise plot that was run with an input range of ±10 V and a gain of 10. Therefore, 1 LSB = 31 µV, so a noise level of 20 LSB is equivalent to 620 µV of noise. Figure 6 shows the DC noise plot of two DAQ products, both of which use the same ADC. Two qualities of the DAQ device can be determined from these noise plots – range of noise and the distribution. The plot in Figure 6a, a National Instruments AT-MIO-16XE-10, has a high distribution of samples at 0 and a very small number of points occurring at other codes. The distribution is Gaussian, which is what is expected from random noise. From the plot, the peak noise level is within ±3 LSB. The plot in Figure 6b, made with a very noisy DAQ product from a competitor, has a far different distribution. The device has noise greater than 20 LSB, with many samples occurring at points other than the expected value.

![Figure 5](image)

**Figure 5.** The input to an instrumentation amplifier that is multiplexing 40 DC signals appears to be a high-frequency AC signal.

![Figure 6](image)

**Figure 6.** Noise plots of two DAQ products with significantly different noise performance though they use the same 16-bit ADC. Figure 6(a) is the NI AT-MIO-16XE-10; Figure 6(b) is a competitor's DAQ product.
With sophisticated measurement hardware such as a plug-in DAQ device, you can get significantly different accuracies depending on the device. NI works hard to make extremely accurate devices, in many cases more accurate than stand-alone instruments. This accuracy is published in NI product specifications. Be leery of boards that are inadequately specified; the specification omitted may be the one that causes inaccurate measurements. By evaluating more analog input specifications than simply the resolution of the ADC converter, you ensure that you get a DAQ product that is accurate enough for your application.

**Analog Outputs**

Analog output circuitry is often required to provide stimuli for a DAQ system. Several specifications for the digital-to-analog converter (DAC) determine the quality of the output signal produced – settling time, slew rate, and output resolution.

- **Settling Time** – Settling time is the time required for the output to settle to the specified accuracy. The settling time is usually specified for a full-scale change in voltage. For more information on settling time, refer to the Analog Inputs section.

- **Slew Rate** – The slew rate is the maximum rate of change that the DAC can produce on the output signal. Settling time and slew rate work together in determining how quickly the DAC changes the output signal level. Therefore, a DAC with a small settling time and a high slew rate can generate high-frequency signals because little time is needed to accurately change the output to a new voltage level.

An example of an application that requires high performance in these parameters is the generation of audio signals. The DAC requires a high slew rate and small settling time to generate the high frequencies necessary to cover the audio range. In contrast, an example of an application that does not require fast D/A conversion is a voltage source that controls a heater. Because the heater cannot respond quickly to a voltage change, fast D/A conversion is unnecessary.

- **Output Resolution** – Output resolution is similar to input resolution; it is the number of bits in the digital code that generates the analog output. A larger number of bits reduces the magnitude of each output voltage increment, thereby making it possible to generate smoothly changing signals. Applications requiring a wide dynamic range with small incremental voltage changes in the analog output signal may need high-resolution voltage outputs.

**Triggers**

Many DAQ applications need to start or stop a DAQ operation based on an external event. Digital triggers synchronize the acquisition and voltage generation to an external digital pulse. Analog triggers, used primarily in analog input operations, start or stop the DAQ operation when an input signal reaches a specified analog voltage level and slope polarity.

**RTSI Bus**

NI developed the RTSI bus for DAQ products. The RTSI bus uses a custom gate array and a ribbon cable to route timing and trigger signals between multiple functions on one DAQ board or between two or more boards. With RTSI bus, you can synchronize A/D conversions, D/A conversions, digital inputs, digital outputs, and counter/timer operations. For example, with RTSI bus, two analog input boards can simultaneously capture data while a third device generates an output pattern synchronized to the sampling rate of the inputs.

**Digital I/O (DIO)**

DIO interfaces are often used on PC DAQ systems to control processes, generate patterns for testing, and communicate with peripheral equipment. In each case, the important parameters include the number of digital lines available, the rate at which you can accept and source digital data on these lines, and the drive capability of the lines. If the digital lines are used for controlling events such as turning on and off heaters, motors, or lights, a high data rate is usually not required because the equipment cannot respond very quickly. The number of digital lines, of course, should match the number of processes to be controlled. In each of these examples, the amount of current required to turn the devices on and off must be less than the available drive current from the device.
With the proper digital signal conditioning accessories, however, you can use the low-current TTL signals to/from the DAQ hardware to monitor/control high voltage and current signals from industrial hardware. For example, the voltage and current needed to open and close a large valve may be on the order of 100 VAC at 2 A. Because the output of a DIO device is 0 to 5 VDC at several milliamperes, an SSR Series, ER-8/16, SC-206x, or SCXI module is needed to switch the power signal to control the valve.

A common DIO application is to transfer data between a computer and equipment such as data loggers, data processors, and printers. Because this equipment usually transfers data in one byte (8-bit) increments, the digital lines on a plug-in DIO device are arranged in groups of eight. In addition, some boards with digital capabilities will have handshaking circuitry for communication-synchronization purposes. The number of channels, data rate, and handshaking capabilities are all important specifications that should be understood and matched to the application needs.

**Timing I/O**

Counter/timer circuitry is useful for many applications, including counting the occurrences of a digital event, digital pulse timing, and generating square waves and pulses. You can implement all these applications using three counter/timer signals – gate, source, and output.

- **Gate** – The gate is a digital input that is used to enable or disable the function of the counter.
- **Source** – The source is a digital input that causes the counter to increment each time it toggles, and therefore provides the timebase for the operation of the counter.
- **Output** – The output generates digital square waves and pulses at the output line.

The most significant specifications for operation of a counter/timer are the resolution and clock frequency. The resolution is the number of bits the counter uses. A higher resolution simply means that the counter can count higher. The clock frequency determines how fast you can toggle the digital source input. With higher frequency, the counter increments faster and therefore can detect higher frequency signals on the input and generate higher frequency pulses and square waves on the output. The DAQ-STC counter/timer used on our E Series DAQ devices, for example, has 16 and 24-bit counters with a clock frequency of 20 MHz. The NI-TIO counter/timer used on NI 660x counter/timer devices has eight 32-bit counters with a maximum clock frequency of 80 MHz.

The DAQ-STC is a NI custom application-specific integrated circuit (ASIC) designed specifically for DAQ applications. In comparison with the off-the-shelf counter/timer chips generally used on DAQ devices, the DAQ-STC stands alone. For example, the DAQ-STC is an up/down counter/timer, meaning that it can use additional external digital signals to count up or down, depending on whether the level is “high” or “low.” This type of counter/timer can measure positioning from rotary or linear encoders. Other special functions include buffered pulse-train generation, timing for equivalent time sampling, relative timestamping, and instantaneous changing of sampling rate.

The NI-TIO is also a custom ASIC designed specifically for timing applications. It incorporates all of the DAQ-STC counter/timer functionality as well as adding new features such as native encoder compatibility, debouncing filters, and two signal edge separation measurements.
Software

Software transforms the PC and the DAQ hardware into a complete data acquisition, analysis, and display system. DAQ hardware without software is useless – and DAQ hardware with poor software is almost useless. The majority of DAQ applications use driver software. Driver software is the layer of software that directly programs the registers of the DAQ hardware, managing its operation and its integration with the computer resources, such as processor interrupts, DMA, and memory. Driver software hides the low-level, complicated details of hardware programming, providing the user with an easy-to-understand interface.

For example, the code snippet that follows shows NI-DAQ function calls used in C to read and scale a voltage from an analog input channel of an MIO-16E-10.

```c
main()                 /* Program to read and scale an analog input */
{
    int    brd,     /* Which board to read analog value from */
            chan,    /* Analog input channel to read value from */
            gain,    /* Software-programmable gain to use on channel */
            reading; /* Binary result of A/D conversion */
    Double voltage; /* Voltage value at input channel after scaling */
    brd = 1;        /* Read from board 1, */
    chan = 3;       /* channel 3, */
    gain = 100;     /* with gain of 100 */
    AI_Read(brd, chan, gain, &reading);      /* Take a reading */
    AI_Scale(brd, gain, reading, &voltage);  /* Scale to voltage */
    printf("\nThe voltage is %lf volts", voltage);
}
```
The increasing sophistication of DAQ hardware, computers, and software continues to emphasize the importance and value of good driver software. Properly selected driver software can deliver an optimal combination of flexibility and performance, while significantly reducing the time required to develop the DAQ application.

While selecting driver software, there are several factors to consider.

**Which Functions Are Available?**

Driver functions for controlling DAQ hardware can be grouped into analog I/O, digital I/O, and timing I/O. Although most drivers will have this basic functionality, you will want to make sure that the driver can do more than simply get data to and from the device. Make sure the driver has the functionality to:

- Acquire data in the background while processing in the foreground
- Use programmed I/O, interrupts, and DMA to transfer data
- Stream data to and from disk
- Perform several functions simultaneously
- Integrate multiple DAQ devices
- Integrate seamlessly with signal conditioning equipment

These and other functions of the DAQ driver, which are included with NI-DAQ, can save the user a considerable amount of time.

**Which Operating Systems Can You Use with the Driver?**

Make sure that the driver software is compatible with the operating systems you plan to use now and in the future. The driver should also be designed to capitalize on the different features and capabilities of the OS. You may also need the flexibility to port your code easily between platforms, say from a Windows PC to a Macintosh. NI-DAQ is available for Windows 2000/NT/ME/9x and Mac OS.

NI-DAQ protects your software investment because you can switch between hardware products or operating systems with little or no modification to your application.

**Which Programming Languages Can You Use with the Driver?**

Ensure that the driver can be called from your favorite programming language and is designed to work well within that development environment. A programming language such as Visual Basic, for example, has an event-driven development environment that uses controls for developing the application. If you develop in the Visual Basic environment, be sure that the driver has custom controls, such as those in NI-DAQ, to match the methodology of the programming language.

**Are the Hardware Functions You Need Accessible in Software?**

A problem occurs when a developer purchases DAQ hardware, then attempts to use the hardware with software, only to find that a required hardware feature is not handled by the software. The problem occurs most frequently when the hardware and software are developed by different companies. NI-DAQ handles every function listed on the data sheets of our DAQ hardware.

**Does the Driver Limit Performance?**

Because the driver is an additional layer, it may cause some performance limitations. In addition, operating systems such as Windows 9x can have significant interrupt latencies. If dealt with improperly, these latencies can greatly reduce the performance of the DAQ system. NI-DAQ is a high-performance driver that has code written specifically to reduce the interrupt latencies of Windows and to provide acquisition rates up to 10 MS/s.
The answers to these questions will give you an indication of the effort that has gone into developing the driver software. Ideally, you want to get your driver software from a company that has as much expertise in the development of the DAQ software as they do in the development of DAQ hardware.

**Application Software**

An additional way to program DAQ hardware is to use application software. However, even with application software, it is important to know the answers to the previous questions because the application software also uses driver software to control the DAQ hardware. The advantage of application software is that it adds analysis and presentation capabilities to the driver software. Application software also integrates instrument control (GPIB, RS-232, and VXI) with data acquisition.

NI offers Measurement Studio, application software for the traditional C programmer, and LabVIEW, application software with graphical programming methodology, for developing complete instrumentation, acquisition, and control applications. Both products can be augmented with add-on toolkits for special functionality. Measurement Studio also contains tools that give complete instrumentation capabilities to Visual C++ and Visual Basic users. National Instruments VI Logger is an easy-to-use yet very flexible tool specifically designed for your data-logging applications.
Developing Your System

To develop a high-quality DAQ system for measurement and control or test and measurement, you must understand each component involved. Of all the DAQ system components, the element that should be examined most closely is the software. Because plug-in DAQ devices do not have displays, the software is the only interface you have to the system. The software is the component that relays all the information about the system, and it is the element that controls the system. The software integrates the transducers, signal conditioning, DAQ hardware, and analysis hardware into a complete, functional DAQ system.

Therefore, when developing a DAQ system, be sure to thoroughly evaluate the software. The hardware components can be selected by determining the requirements of your system and ensuring that the hardware specifications are compatible with your system and your needs. Carefully selecting the proper software – whether it be driver level or application software – can save you development time and money.