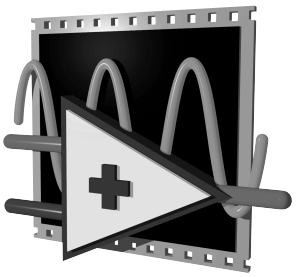


CHAPTER 8

Data Acquisition

The subject of this chapter is data acquisition (DAQ). Focusing on the use of the Easy I/O VIs, the basic notions associated with analog and digital I/O are presented. Some common terms and concepts associated with data acquisition are also discussed, including the components of a DAQ system, signal conditioning, and the types of signals encountered.

GOALS

1. Review some basic notions of signals and signal acquisition.
2. Introduce the organization of the DAQ VIs.
3. Understand the basics of analog and digital input and output using Easy I/O VIs.

8.1 COMPONENTS OF A DAQ SYSTEM

In this chapter, we introduce some of the basic notions associated with **data acquisition**. The focus is on the LabVIEW VIs that can be used in a DAQ system—analog and digital signal **input/output** VIs. The subject of data acquisition cannot be adequately covered in one chapter, although the fundamental ideas can be introduced and discussed in enough detail to generate enthusiasm for pursuing other sources of information.[1] The most effective way to learn about data acquisition is by doing it. Reading about it is not enough.

Your LabVIEW Student Edition CDRom contains a folder at the top level called Chapter 8*. This is where the latest DAQVIs are stored. Copy the files from that folder to the* Chapter 8 *folder in the* Learning *directory. If you do not have easy access to the CDRom, download the latest VIs from the website by using the* ***help*** *pull-down menu and selecting* ***Student Edition Web Resources****.*

At its most basic level, the task of a DAQ system is the measurement or generation of physical signals. Two options for constructing a DAQ system are illustrated in Figure 8.1. Some students think that having a plug-in DAQ board

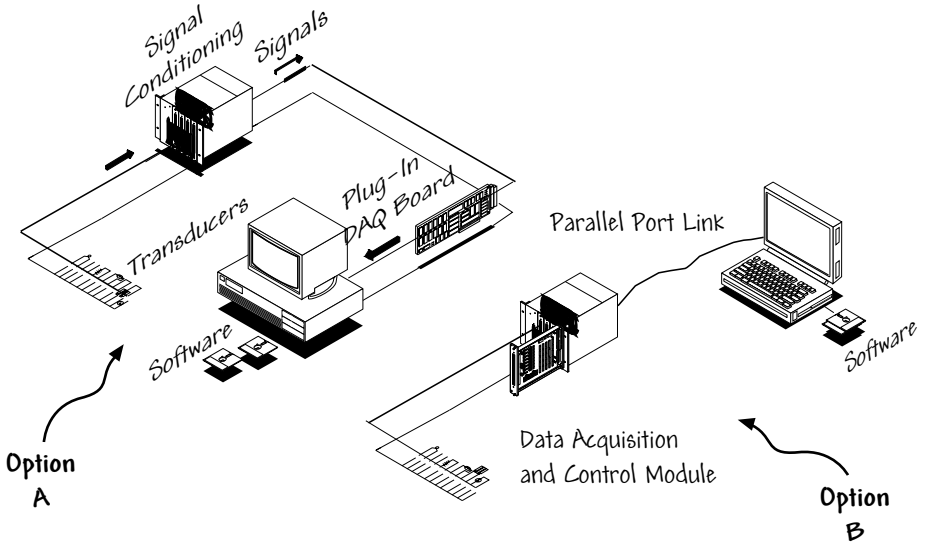


FIGURE 8.1
Two options for a DAQ system.

1. A good source of information for the beginner is the *Data Acquisition Basics Manual*, available from National Instruments, Inc. Part Number 320997C-01. For the advanced students, see Chapters 6 and 7 of *LabVIEW Graphical Programming, 2nd ed.,* by Gary W. Johnson (McGraw-Hill, New York, 1997).

properly installed in a personal computer is equivalent to having a complete DAQ system. This is not the case at all. In fact, the plug-in DAQ board is only one component of the system. A DAQ system generally has (in addition to the plug-in DAQ board) sensors and transducers, signal conditioning, and a suite of software for acquiring and manipulating the raw data, analyzing and displaying (and storing) the data.

In Option A (see Figure 8.1), the plug-in DAQ board resides in the computer. This computer can be a tower, desktop model, or a laptop with PCMCIA slots. In Option B, the DAQ board is external to the computer. With the latter approach you can build DAQ systems using computers without available plug-in slots (such as some laptops), and the computer and DAQ module communicate through various buses, such as the parallel port. Option B systems are usually more practical for remote data acquisition and control applications where you want to bring the DAQ system into the field.

Before a computer-based system can measure a physical signal, a sensor (or transducer) must convert the physical signal into an electrical signal (such as voltage or current). Generally, you cannot connect signals directly to a plug-in DAQ board in a computer, as you can with most stand-alone instruments. In many cases, the measured physical signal is very low-voltage and susceptible to noise. In these situations, the measured signal may need to be amplified and filtered before conversion to a digital format for use in the computer. A signal-conditioning accessory conditions measured signals before the plug-in DAQ board converts them to digital information. More will be said on signal conditioning later in the chapter. Software controls the DAQ system–acquiring the raw data, analyzing the data, and presenting the results.

8.2 TYPES OF SIGNALS

The concepts of signals and systems arise in a wide variety of fields, including science, engineering, and economics. In an effort to develop an analytic framework for studying certain natural phenomena, the notion of an input-output representation has arisen, and is illustrated in Figure 8.2. In the input-output

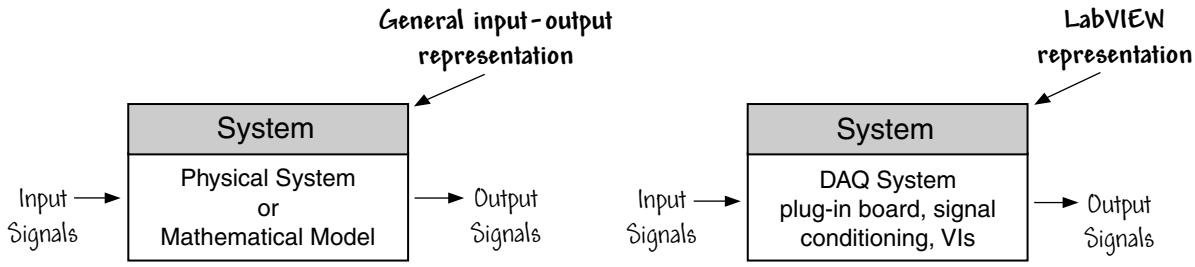


FIGURE 8.2
Modeling physical phenomena with input-output representations.

representation, the input signals are operated on by the system to produce the output signals. Signals are physical quantities that are functions of an independent variable (such as time) and contain information about a natural phenomena. For example, the input signals might be degraded and weak video signals received from a spacecraft approaching a distant asteroid. The system is the DAQ system, which includes the hardware and software to acquire, process and enhance the spacecraft signals (e.g., the spacecraft images of the asteroid can be enhanced to show features and colors more clearly), and the output signals would be the enhanced asteroid images. In general, physical signals are converted to electrical signals (such as voltage or current) before use by the signal-conditioning and DAQ hardware. This conversion is accomplished by some type of transducer. The list of common transducers includes video cameras, thermocouples, strain gauges, and thermistors. Once the physical signals are measured and converted to electrical signals, the information contained in them may be extracted and analyzed.

Five common classes of information that can be extracted from signals are illustrated in Figure 8.3. The signals evolve either continuously or only at discrete

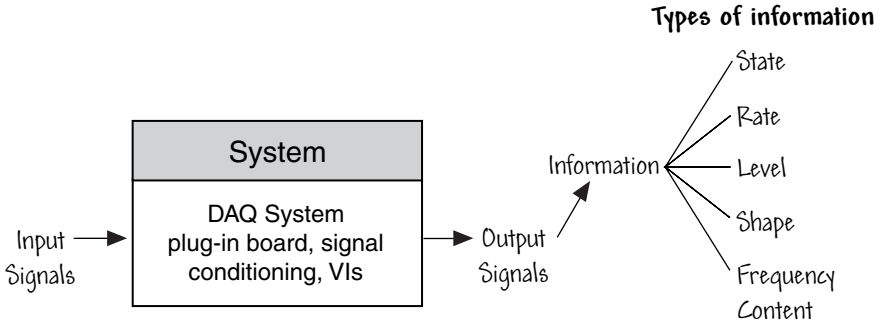


FIGURE 8.3
Measuring and analyzing signals provides information: state, rate, level, shape, and frequency content.

points in time and are known as continuous- or discrete-time signals, respectively. Room temperature is an example of a continuous-time signal (although we may discretize the temperature signal by recording the temperature only at specific sampling points). The end-of-day closing Dow-Jones stock index is an example of a discrete signal, taking on a new value once per working day.

For purposes of discussing data acquisition, we will use the following signal classifications:

- For digital signals, we have two types:
  - on-off
  - pulse train

- For analog signals, we have three types:
  - DC
  - time-domain
  - frequency-domain

The five signal types listed above are classified as analog or digital according to how they convey information. A digital signal has only two possible discrete levels: high level (on) or low level (off). An analog signal, on the other hand, contains information that varies continuously with respect to an independent variable (such as time).

A schematic with the various signal types is shown in Figure 8.4. Each signal type is unique in the information conveyed, and the five signal types closely parallel the five basic types of signal information: state, rate, level, shape, and frequency content.

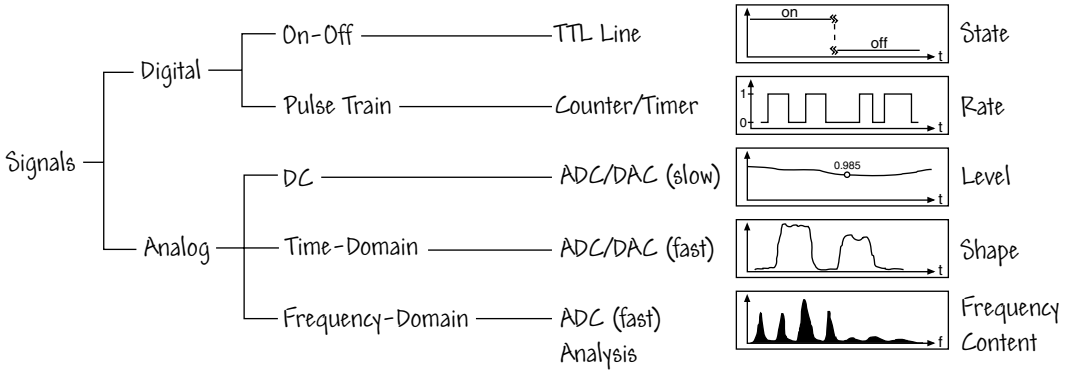


FIGURE 8.4
Signal types. (ADC → analog-to-digital converter, DAC → digital-to-analog converter, TTL → transistor-transistor logic)

8.2.1 Digital Signals

The two types of digital signals that we will consider here are the on-off switch and the pulse train signals. The on-off signal illustrated in Figure 8.5 conveys information concerning the immediate digital *state* of the signal. A simple digital state detector is used to measure this signal type. An example of a digital on-off signal is the output of a transistor-transistor logic (TTL) switch.

The second type of digital signal is the pulse train signal illustrated in Figure 8.6. This signal consists of a series of state transitions, and information is contained in the number of state transitions, the *rate* at which the transitions occur, and the time between one or more state transitions. The output signal of an optical encoder mounted on the shaft of a motor is an example of a digital pulse train signal.

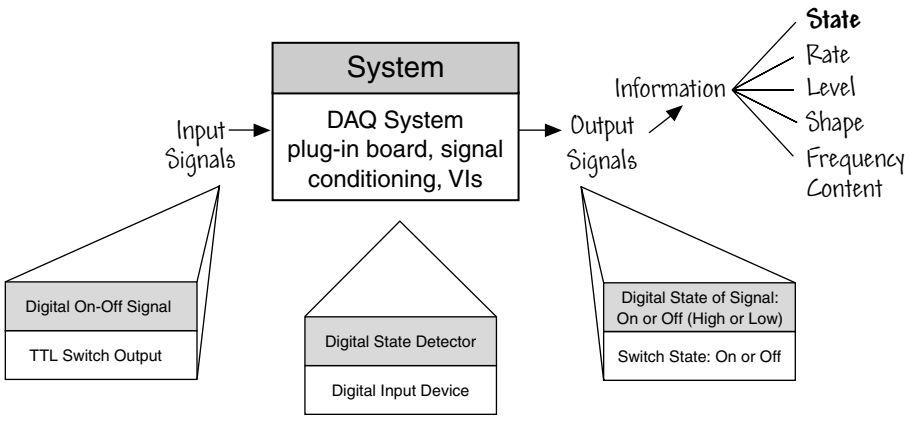


FIGURE 8.5
The on-off signal.

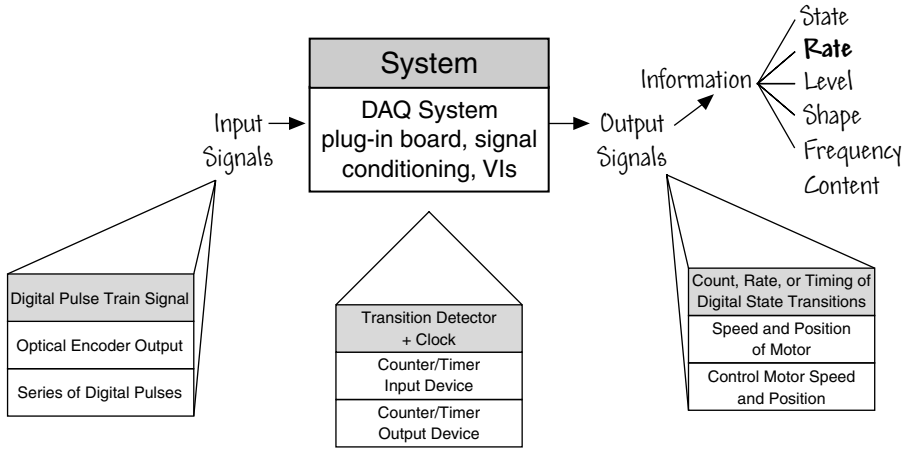


FIGURE 8.6
The pulse train signal.

8.2.2 Analog DC Signals

Analog DC signals are static (or slowly varying analog) signals that convey information in the *level* (or amplitude) of the signal at a given instant. The analog DC signal is depicted in Figure 8.7 in the context of the input-output representation. Since the analog DC signal is static or varies slowly, the accuracy of the measured level is of more concern than the time or rate at which you take the measurement. Common examples of DC signals include temperature, battery voltage, flow rate, pressure, strain-gauge output, and fluid level, as illustrated

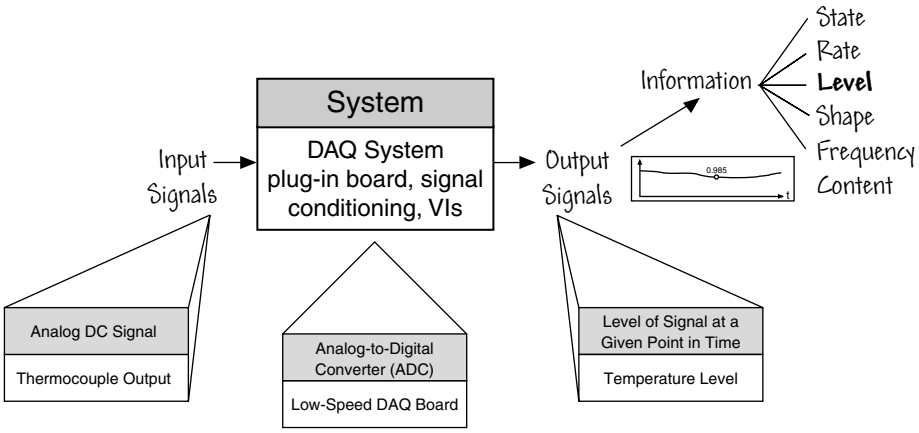


FIGURE 8.7
Analog DC signals are static or slowly varying analog signals.

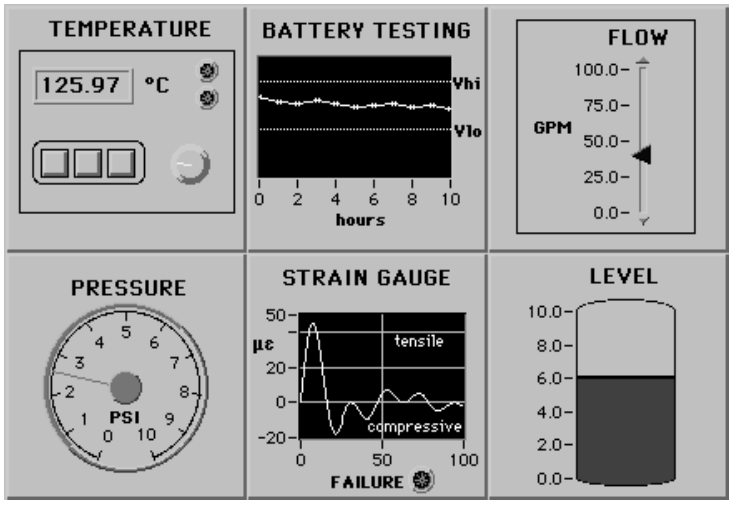


FIGURE 8.8
Common examples of DC signals include temperature, battery voltage, flow rate, pressure, strain-gauge output, and fluid level.

in Figure 8.8. The information contained in the signals is displayed on meters, gauges, strip charts, and numerical readouts.

The DAQ system will return a single value indicating the magnitude of the signal at the requested time. For the measurement to be an accurate representation of the signal, the DAQ system must possess adequate accuracy and resolution capability and adequate sampling rates (usually slow).

8.2.3 Analog Time-Domain Signals

Analog time-domain signals differ from other signals in that they convey useful information in signal level *and* the way this level varies with time. The information associated with a time-domain signal (also referred to as a waveform) includes such information as time to peak, peak magnitude, time to settle, slope, and shapes of peaks. An analog time-domain signal is illustrated in Figure 8.9.

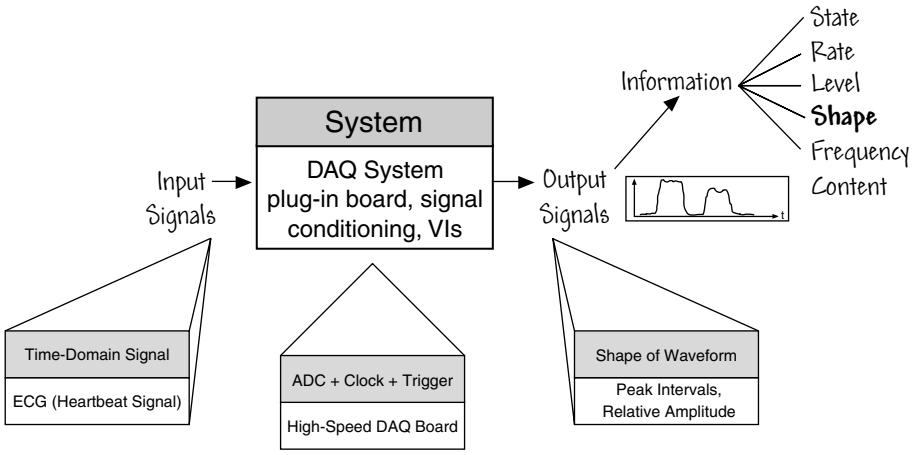


FIGURE 8.9
Analog time-domain signals convey information in the signal level and in the way this level varies with time.

You must take a precisely timed sequence of individual amplitude measurements to measure the shape of a time-domain signal. DAQ systems that are used to measure time-domain signals usually have an **analog-to-digital conversion** (ADC) component, a sample clock, and a trigger.

The physical signal must be sampled and measured at a rate that adequately represents the shape of the signal. Therefore, when acquiring analog time-domain signals, DAQ systems need a high-bandwidth ADC to sample the signal at high rates. The signal must also be sampled accurately without significant loss of precision.

Generally the signal measurements need to start at a specified time to guarantee that an interesting segment of the signal is acquired. The sample clock accurately times the occurrence of each analog-to-digital (A/D) conversion. In many situations, triggering is necessary to initiate the measurement process at a precise time. The trigger starts the measurement at the proper time based on some external condition specified by the user. There are an unlimited number of different time-domain signals, some of which are shown in Figure 8.10.

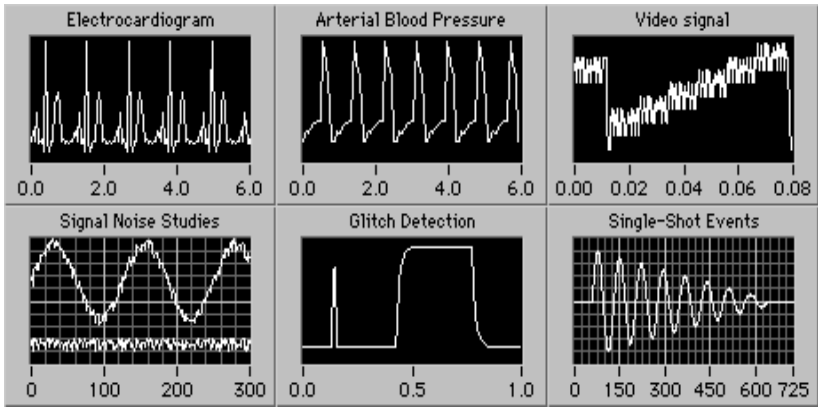


FIGURE 8.10
Six different time-domain signals.

8.2.4 Analog Frequency-Domain Signals

Analog frequency-domain signals are similar to time-domain signals in that they also convey information in the way the signals vary with time. However, the information extracted from a frequency-domain signal is based on the signal frequency content, as opposed to the shape of the signal. An analog frequency-domain signal is shown in Figure 8.11.

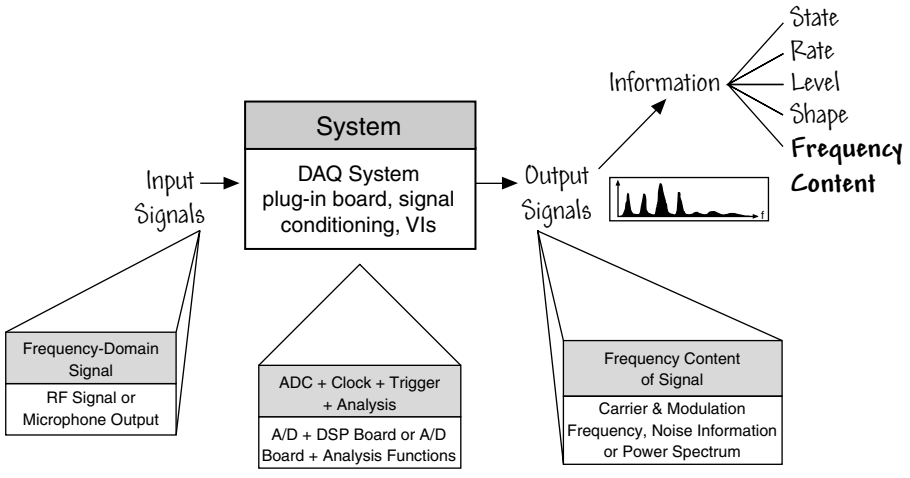


FIGURE 8.11
The information extracted from a frequency-domain signal is based on the signal frequency content.

As with DAQ systems that measure time-domain signals, a system used to measure a frequency-domain signal must also include an ADC, a sample clock, and a trigger to accurately capture the waveform. Additionally, the DAQ system must include the necessary analysis capabilities to extract frequency information from the signal. You can perform this type of digital signal processing (DSP) using application software or special DSP hardware designed to analyze the signal quickly and efficiently.

In short, a DAQ system that acquires analog frequency-domain signals should possess a high-bandwidth ADC capability to sample signals at high rates and an accurate clock to take measurements at precise intervals. Also, it is common to use triggers to initiate the measurement process precisely at prespecified times. Finally, a complete DAQ system should provide a library of analysis functions, including a function to convert time-domain information to frequency-domain information. Chapter 11 discusses some of the LabVIEW analysis functions, but time and space limitations make it impossible to cover all those available to VI developers.

Figure 8.12 shows some examples of frequency-domain signals. Each example in the figure includes a graph of the originally measured signal as it varies with respect to time, as well as a graph of the signal frequency spectrum. While you can analyze any signal in the frequency domain, certain signals and application areas lend themselves especially to this type of analysis. Among these areas are speech and acoustics analysis, geophysical signals, vibration, and studies of system transfer functions.

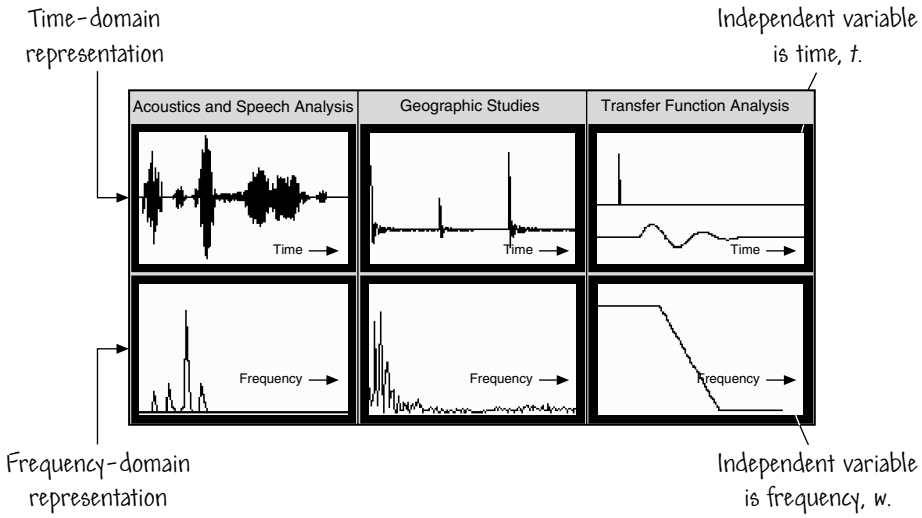


FIGURE 8.12
Three examples of frequency-domain signals: speech and acoustics analysis, geophysical signals, and transfer function frequency response.

8.2.5 One Signal—Five Measurement Perspectives

The five classifications of signals presented in the previous discussions are not mutually exclusive. A particular signal may convey more than one type of information, and you can classify a signal as more than one type and measure it in more than one way. In fact, you can use simpler measurement techniques with the digital on-off, pulse train, and DC signals, because they are just simpler cases of the analog time-domain signals.

The measurement technique you choose depends on the information you want to extract from the signal. In many cases you can measure the same signal with different types of systems, ranging from a simple digital input board to a sophisticated frequency analysis system. Figure 8.13 demonstrates how a series of voltage pulses can provide information for all five signal classes.

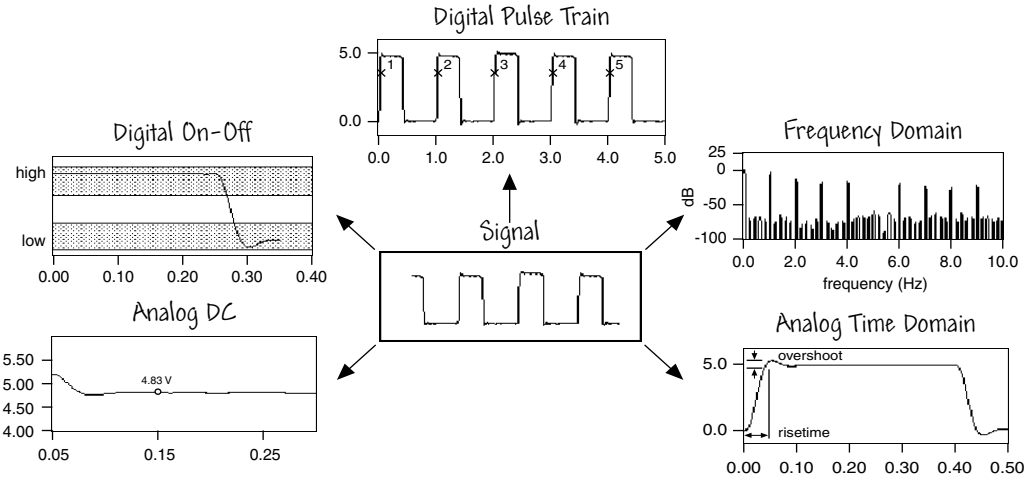


FIGURE 8.13
A series of voltage pulses can provide information for all five signal classes.

8.3 COMMON TRANSDUCERS AND SIGNAL CONDITIONING

When measuring a physical phenomena, a transducer must convert this phenomena (such as temperature or force) into a measurable electrical signal (such as voltage or current). Table 8.1 lists some common transducers used to convert physical phenomena into measurable quantities.

All transducers output an electrical signal that is often not well suited for a direct measurement by the DAQ system. For example, the output voltage of most thermocouples is very small and susceptible to noise, and often needs to be amplified and filtered before measuring. Figure 8.14 shows some common types of

TABLE 8.1 Phenomena and Transducers

Phenomenon	Transducer
Temperature	Thermocouples Resistance temperature detectors (RTDs) Thermistors Integrated circuit sensor
Light	Vacuum tube photosensors Photoconductive cells
Sound	Microphone
Force and pressure	Strain gauges Piezoelectric transducers Load cells
Position (displacement)	Potentiometers Linear voltage differential transformer (LVDT) Optical encoder
Fluid flow	Head meters Rotational flowmeters Ultrasonic flowmeters
pH	pH electrodes

transducer and signal pairs and the required signal conditioning. A highly expandable signal conditioning system—dubbed Signal Conditioning eXtensions for Instrumentation (**SCXI**) made by National Instruments, Inc.—conditions low-level signals in a noisy environment within an external chassis located near the sensors. The close proximity improves the signal-to-noise ratio of the signals reaching the DAQ boards. Of course, LabVIEW works with signal conditioning systems other than SCXI.

Some **signal conditioning** (such as linearization and scaling) can be performed in the software. LabVIEW provides several VIs in the **Data Acquisition** palette for such purposes. The remainder of this section is devoted to short descriptions of some of the basic ideas involved in signal conditioning. Further information on signal conditioning and SCXI hardware (i.e., setup procedures for SCXI hardware, hardware operating modes, and programming considerations for SCXI) can be found in other LabVIEW documentation (check the NI website).

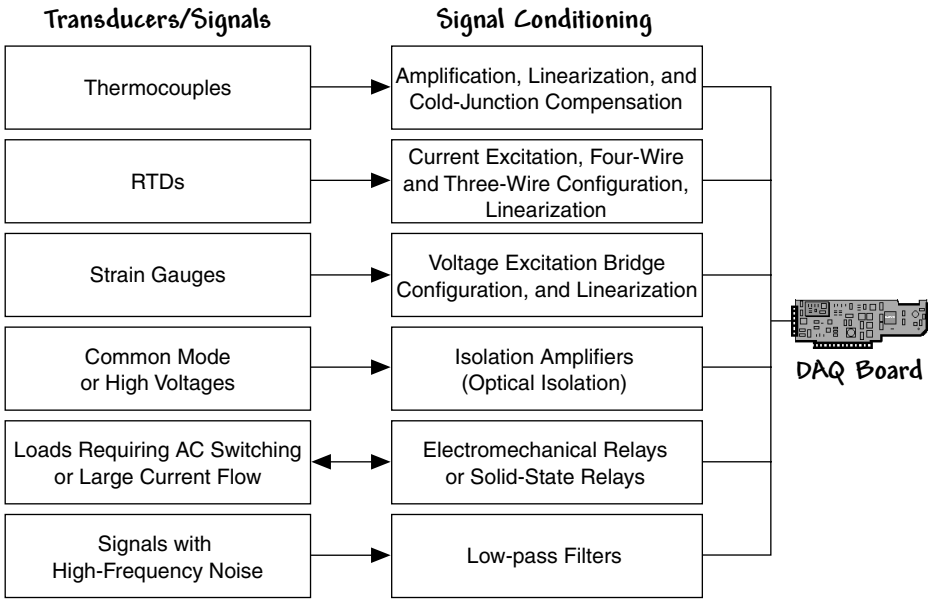
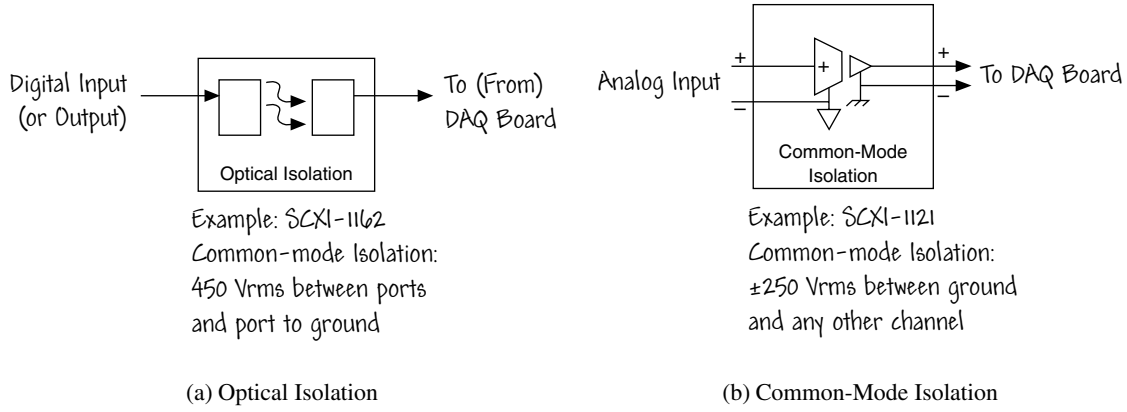


FIGURE 8.14
Common types of transducers/signals and the required signal conditioning.

Some common types of signal conditioning follow:

- Transducer excitation: Certain transducers (such as strain gauges) require external voltages or currents to excite their own circuitry in a process known as transducer excitation. The process is similar to a television needing power to receive and decode video and audio signals. The necessary excitation in a DAQ system can be provided by the plug-in DAQ boards and the signal-conditioning peripherals, and sometimes by external instruments.
- Linearization: Common transducers (such as strain gauges, thermistors, RTDs, and thermocouples) generate voltages that are nonlinear with respect to the phenomena they represent. The LabVIEW DAQ Utility VIs can perform software **linearization** to scale a transducer voltage to the correct units of strain or temperature.
- Isolation: Another common use for signal conditioning is to isolate the transducer signals from the computer. For example, when the signal being monitored contains large voltage spikes that could damage a computer or harm a person, you should not connect the signal directly to a DAQ board without some type of isolation. Figure 8.15 shows two common methods for isolating signals.
- Filtering: Another form of signal conditioning is the filtering of unwanted signals from the desired signal. A common filter reduces 60 Hz AC power-line noise present in many signals. Other well-known types of filters



(a) Optical Isolation (b) Common-Mode Isolation

FIGURE 8.15
Two common methods for isolating signals.

include low-pass, high-pass, and notch filters. Some DAQ boards and signal-conditioning devices have built-in filters.

- Amplification: This is the most common type of signal conditioning. Amplification maximizes the use of the available voltage range to increase the accuracy of the digitized signal and to increase the signal-to-noise ratio (SNR). Low-level signals should be amplified at the DAQ board or at an external signal-conditioning peripheral positioned near the source of the signal, as shown in Figure 8.16. One reason to amplify low-level signals close to the signal source, instead of at the DAQ board, is to increase the signal-to-noise ratio. Consider the case where you amplify the signal only on the DAQ board. Then the DAQ board will also measure and digitize any noise that enters the lead wires along the path as the signal travels from the source to the DAQ board. On the other hand, the ratio of signal voltage to noise voltage that enters the lead wires is larger if you amplify the signal close to the signal source. Table 8.2 shows how the SNR changes with the location of amplification.

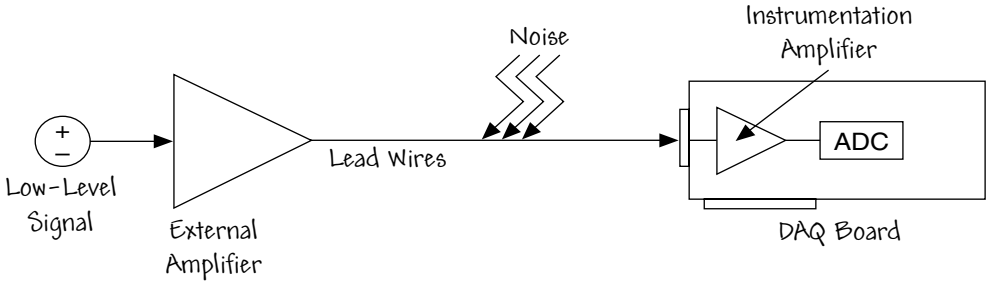


FIGURE 8.16
Low-level signals should be amplified at the DAQ board or at an external signal-conditioning peripheral positioned near the source of the signal.

TABLE 8.2 Effects of Amplification on Signal-to-Noise Ratio (S.C. → Signal-Conditioning Peripheral)

	Signal Voltage	S.C. Amplification	Noise in Lead Wires	DAQ Board Amplification	Digitized Voltage	SNR
Amplify only at DAQ board	0.01 V	None	0.001 V	×100	1.1 V	10
Amplify at S.C. and DAQ board	0.01 V	×10	0.001 V	×10	1.01 V	100
Amplify only at S.C.	0.01V	×100	0.001 V	None	1.001 V	1000

You can minimize the effects of external noise on the measured signal by using shielded or twisted-pair cables and by minimizing the cable length. Keeping cables away from AC power cables and computer monitors will also help minimize 50/60 Hz noise.

8.4 SIGNAL GROUNDING AND MEASUREMENTS

Up to this point, we have discussed three components of the DAQ system: signals, transducers, and signal conditioning. Now you might be tempted to think that all that remains is to wire the signal source to the DAQ board and begin acquiring data. However, a few important items must be considered:

- The nature of the signal source (grounded or floating)
- The grounding configuration of the amplifier on the signal conditioning hardware or DAQ board
- The cabling scheme to connect all the components together

A DAQ system is depicted in Figure 8.17 highlighting the signals and the cabling.

8.4.1 Signal Source Reference Configuration

Signal sources come in two forms: referenced and nonreferenced. Referenced sources are usually called **grounded signals**, and nonreferenced sources are called **floating signals**. A schematic of a grounded signal source is shown in Figure 8.18a.

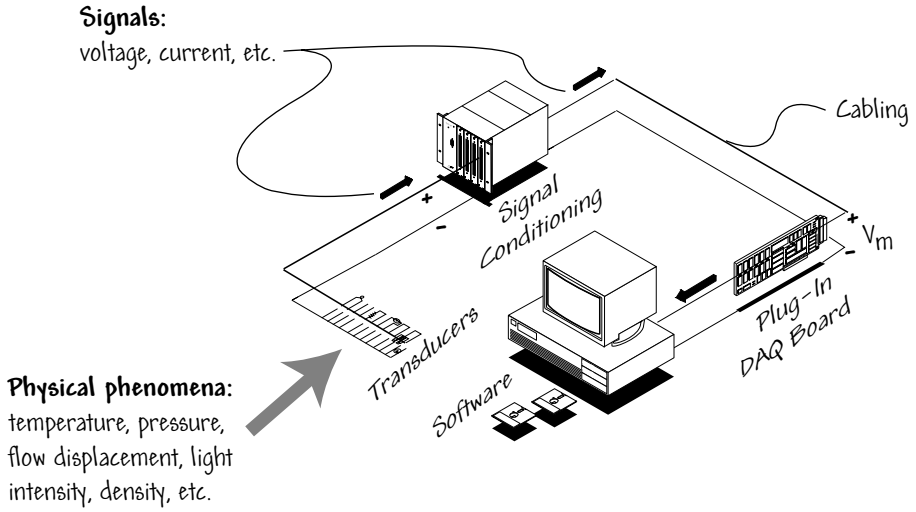
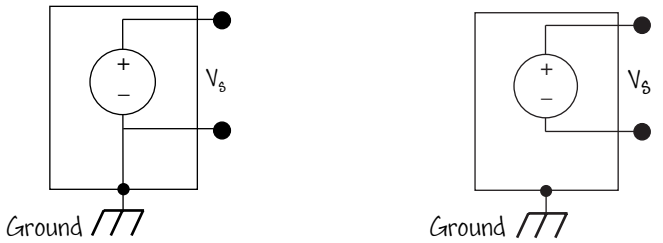


FIGURE 8.17
A DAQ system highlighting the signals and the cabling.



(a) Grounded signal source (b) Floating signal source

FIGURE 8.18
Grounded signal sources have voltage signals that are referenced to a system ground. Floating signal sources contain a signal that is not connected to an absolute reference.

Grounded signal sources have voltage signals that are referenced to a system ground, such as earth or a building ground. Devices that plug into a building ground through wall outlets, such as signal generators and power supplies, are the most common examples of grounded signal sources. Grounded signal sources share a common ground with the DAQ board.

Floating signal sources contain a signal (e.g., a voltage) that is not connected to an absolute reference, such as earth or a building ground. Some common examples of floating signals are batteries, battery-powered sources, thermocouples, transformers, isolation amplifiers, and any instrument that explicitly floats its output signal. As illustrated in Figure 8.18b, neither terminal of the floating source is connected to the electrical outlet ground.

8.4.2 Measurement System

A schematic of a measurement system is depicted in Figure 8.19. A measurement system can be placed in one of three categories:

- Differential
- Referenced single-ended (RSE)
- Nonreferenced single-ended (NRSE)

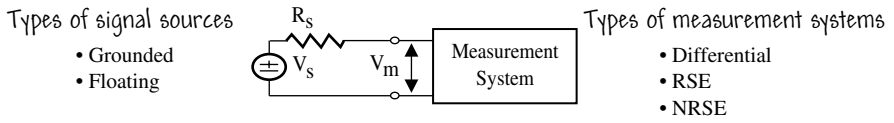


FIGURE 8.19
Types of signal sources and measurement systems.

In a **differential** measurement system, you do not need to connect either input to a fixed reference, such as earth or a building ground. DAQ devices with instrumentation amplifiers can be configured as differential measurement systems. Figure 8.20 depicts the 8-channel differential measurement system used in the MIO series devices. A **channel** is a pin or wire lead where analog or digital signals enter or leave a DAQ device. For this device, the analog input ground pin labeled AIGND is the measurement system ground. The analog multiplexers (labeled MUX in the figure) increase the number of available measurement channels while still using a single instrumentation amplifier.

An ideal differential measurement system, shown in Figure 8.21, reads only the potential difference between its two terminals—the (+) and (−) inputs. It completely rejects any voltage present at the instrumentation amplifier inputs with respect to the amplifier ground. In other words, an ideal differential measurement system completely rejects the common-mode voltage.

A **referenced single-ended** (RSE) measurement system measures a signal with respect to building ground and is sometimes called a grounded measurement system. Figure 8.22 depicts a 16-channel version of an RSE measurement system.

DAQ devices often use a **nonreferenced single-ended** (NRSE) measurement system, which is a variation of the RSE measurement system. In an NRSE measurement system, all measurements are made with respect to a common reference, because all of the input signals are already grounded. Figure 8.23 depicts an NRSE measurement system. AISENSE is the common reference for taking measurements and all signals in the system share this common reference. AIGND is the system ground.

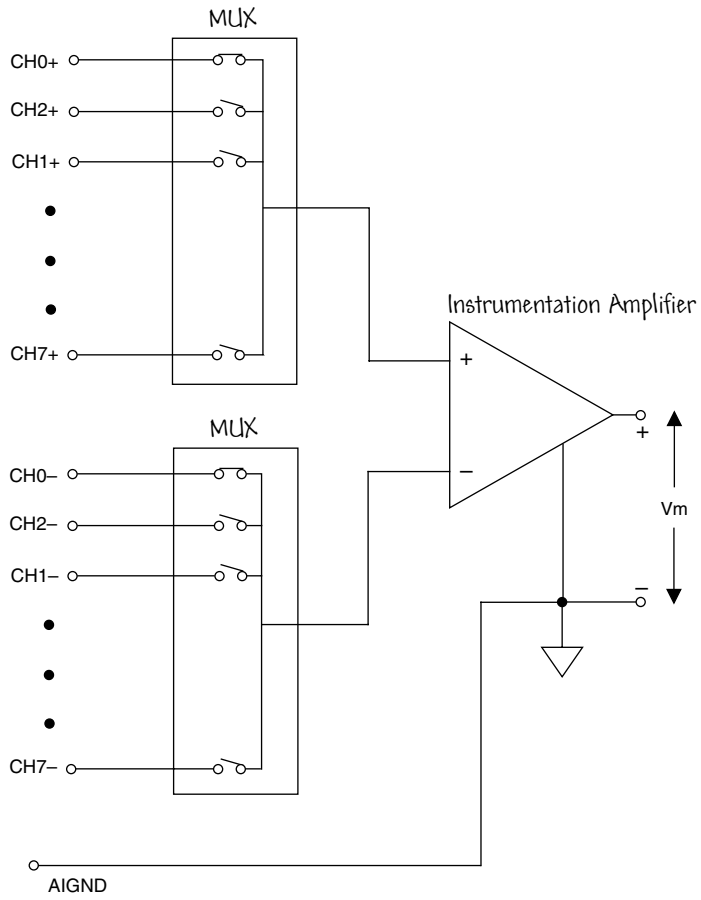


FIGURE 8.20
An 8-channel differential measurement system.

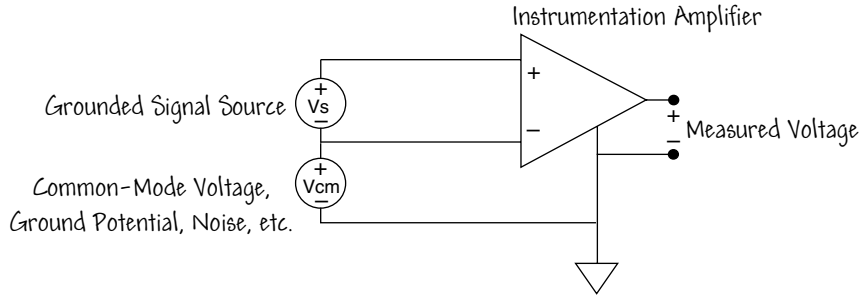


FIGURE 8.21
An ideal differential measurement system completely rejects common-mode voltage.

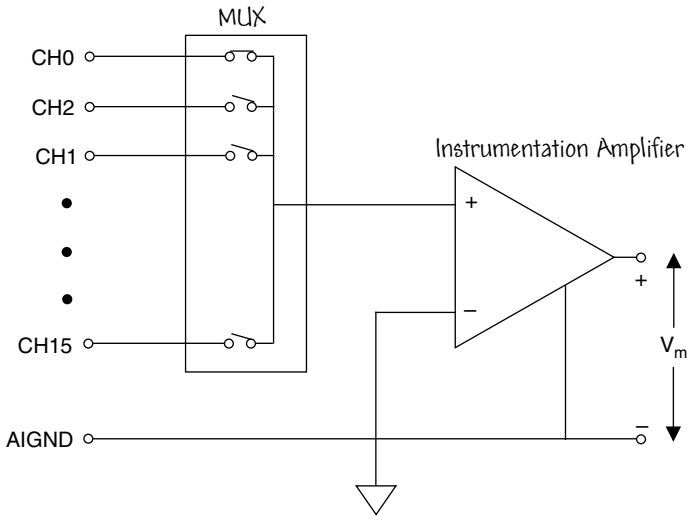


FIGURE 8.22
A 16-channel RSE measurement system.

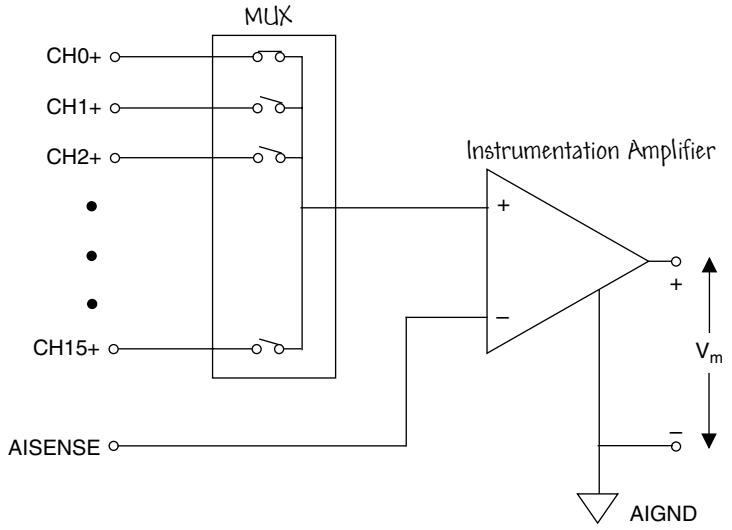


FIGURE 8.23
An NRSE measurement system.

A grounded signal source is best when using a differential or NRSE measurement system. An RSE measurement system can be used with a grounded signal source if the signal levels are high and the cabling has a low impedance. The measured signal voltage will be degraded with the RSE system, but the degradation is usually small relative to the signal voltage. Floating signal sources can be measured with differential, RSE, and NRSE measurement systems. In general, a differential measurement system is preferable because it rejects the ground loop-induced errors and reduces the noise picked up in the environment. On the

other hand, the single-ended configuration allows for twice as many measurement channels and is acceptable when the magnitude of the induced errors is smaller than the required accuracy of the data. You can use single-ended measurement systems when all input signals meet the following criteria:

1. High-level signals (normally, greater than 1 V).
2. Short or properly shielded cabling traveling through a noise-free environment (normally less than 15 ft).
3. All signals can share a common reference signal at the source.

A summary of analog input connections is given in Figure 8.24.

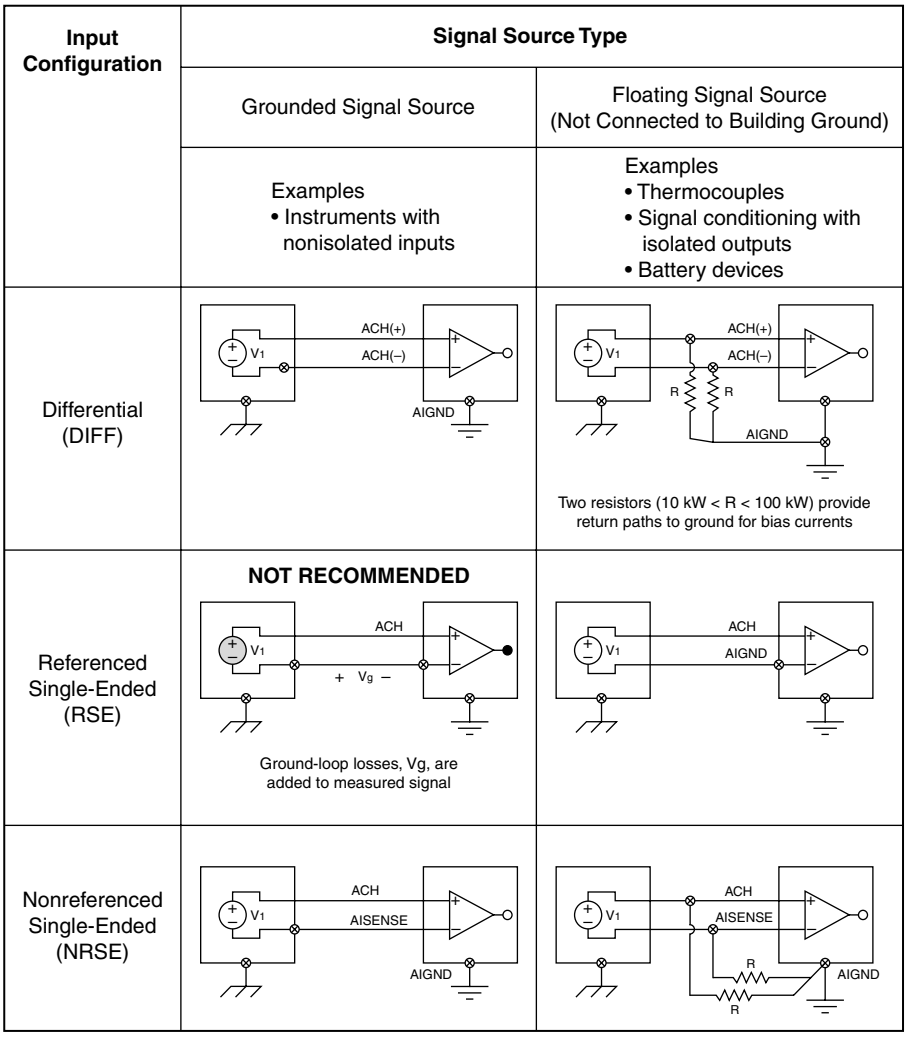


FIGURE 8.24
Summary of analog input connections.

8.5 ANALOG I/O CONSIDERATIONS

When preparing to configure a DAQ board you need to consider the quality of the analog-to-digital conversion. There are many questions that arise when making analog signal measurements with your DAQ system. For example, what are the signal magnitude limits? How fast does the signal vary with time? The latter question is important because the sample rate determines how often the A/D conversions take place. The four parameters of concern are

- Resolution
- Range
- Signal limit settings
- Sampling rate

Depending on the type of DAQ board you have, these four parameters will be set either in the hardware (using dip switches and jumpers on the board) or set using software (with the **Measurement and Automation Explorer** discussed later in this chapter).

The number of bits used to represent an analog signal determines the **ADC resolution** of the analog-to-digital conversion. You can compare the resolution on a DAQ device to the number of divisions on a ruler. For a fixed ruler length, the more divisions you have on the ruler, the more precise the measurements that you can make. A ruler marked off in millimeters can be read more accurately than a ruler marked off in centimeters. Similarly, the higher the ADC resolution, the higher the number of divisions of the ADC range and, therefore, the more accurately the analog signal can be represented. For example, a 3-bit ADC divides the signal range into 2^3 divisions, with each division represented by a binary or digital code between 000 and 111. The ADC then translates measurements of the analog signal to one of the digital divisions. Figure 8.25 shows a sine wave digital image obtained by a 3-bit ADC. Clearly, the digital signal does not represent the original signal adequately because there are too few divisions to represent the varying voltages of the analog signal. By increasing the resolution to 16 bits, however, the ADC's number of divisions increases from 2^3 to 2^{16}. The ADC can now provide an adequate representation of the analog signal.

The **range** refers to the minimum and maximum analog signal levels that the ADC can handle. You should attempt to match the range to that of the analog input signal to take best advantage of the available resolution. Fortunately, many DAQ devices feature selectable ranges. Consider the example shown in Figure 8.26a, where the 3-bit ADC has eight digital divisions in the range from 0 to 10 volts. If you select a range of -10.00 to $+10.00$ volts, as shown in Figure 8.26b, the same ADC now separates a 20-volt range into eight divisions. The

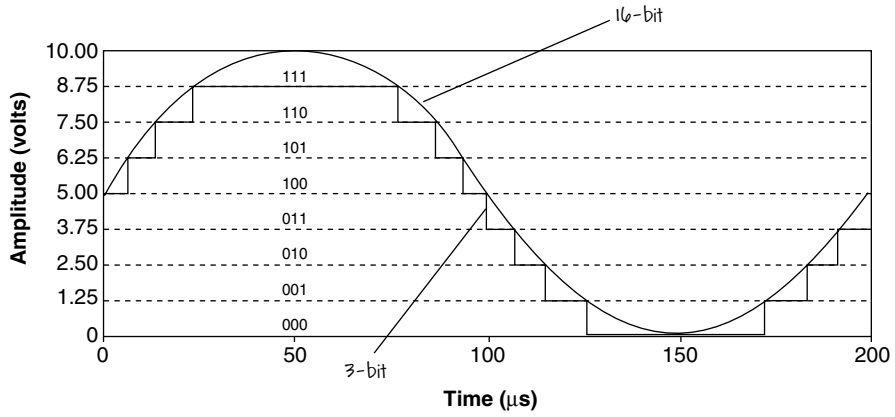
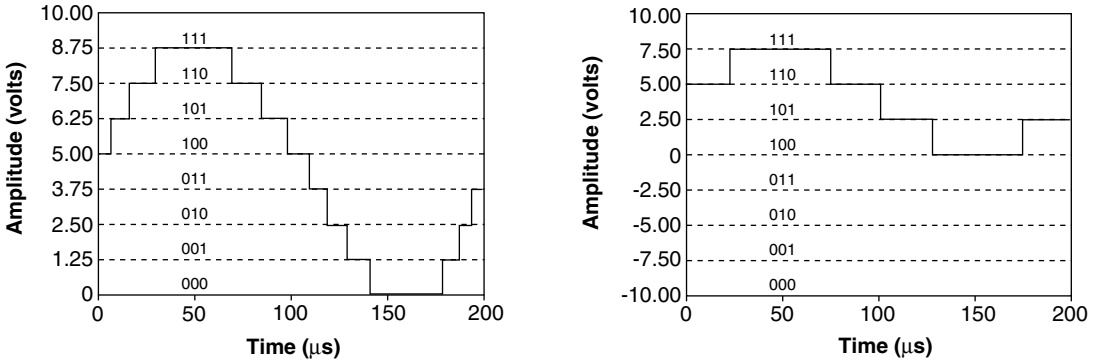


FIGURE 8.25
16-bit versus 3-bit resolution (5 kHz sine wave).



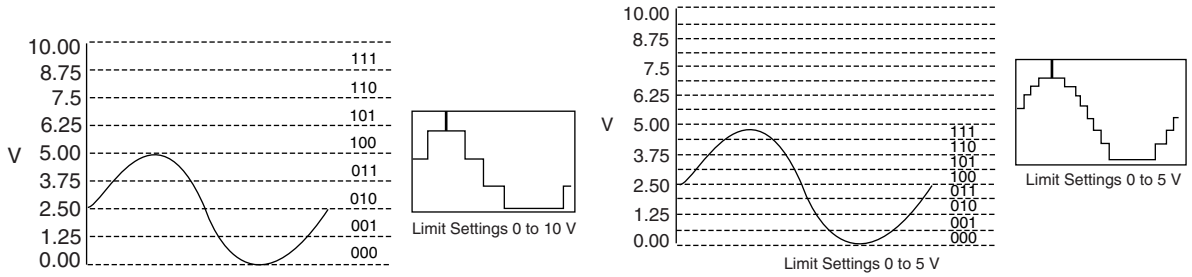
(a) Range = 0 V to 10 V

(b) Range = −10 V to 10 V

FIGURE 8.26
The 3-bit ADC in (a) has eight digital divisions in the range from 0 to 10 volts—if you select a range of −10.00 to 10.00 volts as in (b), the same ADC now separates a 20-volt range into eight divisions.

smallest detectable voltage increases from 1.25 to 2.50 volts, and you now have a much less accurate representation of the signal.

The **limit settings** are the maximum and minimum values of the signal you are measuring. The closer the limit setting is to the incoming analog signal maximum and minimum, the more digital divisions will be available to the ADC to represent the signal. Using a 3-bit ADC and a range setting of 0.00 to 10.00 volts, we see in Figure 8.27 the effects of a limit setting between 0 and 5 volts and 0 and 10 volts. With a limit setting of 0 to 10 volts, the ADC uses only four of the eight divisions in the conversion; with a limit setting of 0 to 5 volts, the ADC now has



(a) Limit settings: 0 V to 10 V

(b) Limit settings: 0 V to 5 V

FIGURE 8.27
Precise limit setting allows the ADC to use more digital divisions to represent the signal.

access to all eight digital divisions. This makes the digital representation of the signal more accurate.

The resolution and range of a DAQ board and the limit settings determine the smallest detectable change in the input voltage. This change in voltage represents 1 **least significant bit** (LSB) of the digital value and is often called the **code width.** The smallest code width is calculated with the following formula:

$$V_{cw} = \frac{range}{2^{resolution}},$$

where the resolution is given in bits. For example, a 12-bit DAQ board with a 0 to 10-V range detects a 2.4-mV change. This is calculated as follows:

$$V_{cw} = \frac{range}{2^{resolution}} = \frac{10}{2^{12}} = 2.4 \text{ mV},$$

while the same board with a −10 to 10-V range detects only a change of 4.8 mV:

$$V_{cw} = \frac{range}{2^{resolution}} = \frac{20}{2^{12}} = 4.8 \text{ mV}.$$

A high-resolution ADC provides a smaller code width for a given range. For example, consider the two preceding examples, except that the resolution is now 16-bit. A 16-bit DAQ board with a 0 to 10-V range detects a 0.15-mV change:

$$V_{cw} = \frac{range}{2^{resolution}} = \frac{10}{2^{16}} = 0.15 \text{ mV},$$

while the same board with a −10 to 10-V range detects only a change of 0.3 mV:

$$V_{cw} = \frac{range}{2^{resolution}} = \frac{20}{2^{16}} = 0.3 \text{ mV}.$$

You also need to determine whether your signal is unipolar or bipolar, as this affects the code width. **Unipolar** signals range from 0 V to a positive value (e.g., 0 V to 10 V). **Bipolar** signals range from a negative value to a positive value (e.g., −5 V to 5 V). A smaller code width is obtained by specifying the range to be unipolar, if indeed the signal is unipolar; and conversely, specifying the range as bipolar if the signal is bipolar. If the maximum and minimum variation of the analog signal is smaller than the value of the range, you will need to set the limit settings to more accurately reflect the analog signal variation.

Some confusion may exist about selecting the limit settings rather than selecting the gain. *When you set the signal limit settings, you are effectively selecting the gain for that signal. Limit settings automatically magnify the magnitude of the signal to create more precise analog-to-digital conversions.*

The **sampling rate** is the rate at which the DAQ board samples an incoming analog signal. Figure 8.28 shows an adequately sampled signal as well as the effects of undersampling. The sampling rate determines how often an analog-to-digital (A/D) conversion takes place. Computing the proper sampling rate requires knowledge of the maximum frequency of the incoming signal and the accuracy required of the digital representation of the analog signal. It also requires some knowledge of the noise affecting the incoming signal and the capabilities of your hardware. A fast sampling rate acquires more points in a given time, allowing, in general, a better representation of the original signal than a slow sampling rate would allow. In fact, sampling too slowly may result in a misrepresentation of the incoming analog signal.

The effect of undersampling is that the signal appears as if it has a different frequency than it truly does. This misrepresentation is called an *alias*. To prevent

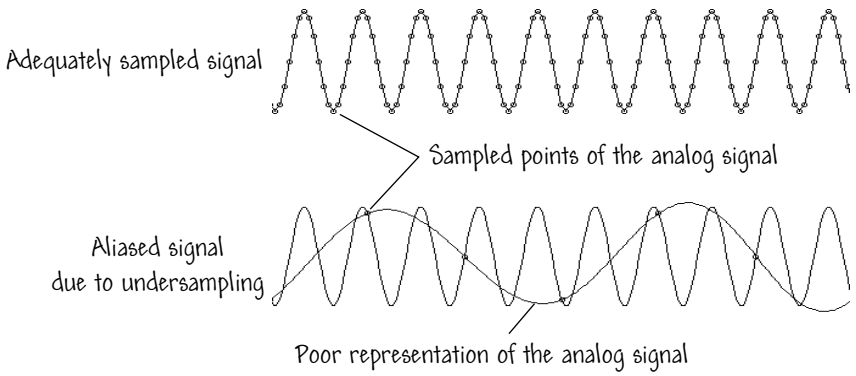


FIGURE 8.28
Sampling too slowly may result in a poor representation of the analog signal.

undersampling, you must sample, at a minimum, twice the rate of the maximum frequency component of the incoming signal. One way to deal with aliasing is to use low-pass filters that attenuate any frequency components in the incoming signal above the Nyquist frequency (defined to be one-half the sampling frequency). These filters are known as *antialiasing filters*.

Signal-Limit Selection

The kind of calculations required to select the signal limits for a DAQ application are illustrated in this example. Keep in mind that the objective is to minimize the code width while making sure that the entire signal fits within the allowable device range.

Assume your transducer output is a sine wave with an amplitude of ±30 mV and an offset of 10 mV. Your board has limit settings of 0 to +10 V, ±10 V, and ±5 V. What limit setting would you select for maximum precision if you use a DAQ board with 12-bit resolution?

Table 8.3 shows how the code width of a hypothetical 12-bit DAQ device varies with range and limit settings. The values in Table 8.3 depend on the hardware, and you should consult your DAQ hardware documentation to determine the available device voltage range and limit settings for your device.

Step 1: Select a range.

Since the signal input is bipolar (the sine wave contains both positive and negative voltages), we must choose one of the bipolar ADC ranges. For our board, there are two, so to start we choose ±5 V, for a range of 10 V.

Step 2: Determine the code width.

Using the formula for determing the code width, we compute

$$V_{cw} = \frac{range}{2^{resolution}} = \frac{10}{2^{12}} = 2.4 \text{ mV}.$$

Step 3: Choose a limit setting.

Since our signal has a maximum magnitude of +30 mV, we must choose input settings that allow us to read ±30 mV. Referring to Table 8.3, we find that (for a range of −5 to 5 V) a limit setting of −50 mV to 50 mV will cover the signal variation of ±30 mV. Choosing this limit setting yields a code width (or precision) of 24.4 μV.

Step 4: Repeat steps 1 through 3 for range of ±10 V.

We must also try the other range of ±10 V to see if it yields a smaller code width. Repeating steps 1 through 3 gives us a limit setting of −0.1 V to 0.1 V and a code width of 48.8 μV. This does not improve our accuracy. Thus, we choose the settings as follows:

- Range = −5 to 5 V
- Limit setting = −50 mV to 50 mV

TABLE 8.3 Code Width for Various Ranges and Limit Settings

Voltage Range	Limit Settings	Code Width
0 to 10 V	0 to 10 V	2.44 mV
	0 to 5 V	1.22 mV
	0 to 2.5 V	610 μV
	0 to 1.25 V	305 μV
	0 to 1 V	244 μV
	0 to 0.1 V	24.4 μV
	0 to 20 mV	4.88 μV
−5 to 5 V	−5 to 5 V	2.44 mV
	−2.5 to 2.5 V	1.22 mV
	−1.25 to 1.25 V	610 μV
	−0.625 to 0.625 V	305 μV
	−0.5 to 0.5 V	244 μV
	−50 to 50 mV	24.4 μV
	−105 to 10 mV	4.88 μV
−10 to 10 V	−10 to 10 V	4.88 mV
	−5 to 5 V	2.44 mV
	−2.5 to 2.5 V	1.22 mV
	−1.25 to 1.25 V	610 μV
	−1 to 1 V	488 μV
	−0.1 to 0.1 V	48.8 μV
	−20 to 20 mV	9.76 μV

8.6 DAQ VI ORGANIZATION

The LabVIEW DAQ VIs are organized into palettes corresponding to the type of operation involved—analog input, analog output, or digital I/O, as illustrated in Figure 8.29. You access the palette by clicking on **Data Acquisition** in the **Functions** palette. The DAQ VIs are organized into six palettes:

- **Analog Input**
- **Analog Output**

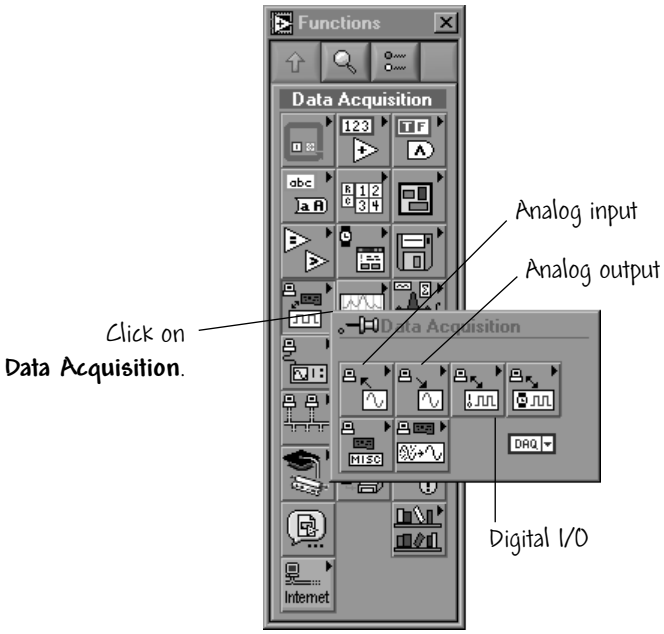


FIGURE 8.29
The **DAQ Acquisition** functions palette.

- **Digital I/O**
- **Counter**
- **Calibration and Configuration**
- **Signal Conditioning**

In this book, we will focus on the first three palettes: **Analog Input**, **Analog Output**, and **Digital I/O**.

Each palette contains VIs or palettes of VIs organized as Easy I/O, Intermediate, Utility, and Advanced VIs. The **Analog Input** palette shown in Figure 8.30 illustrates the organization. The top tier of VIs contains Easy I/O VIs, and the bottom tier contains Intermediate VIs. There are also two subpalettes: one for the **Analog Input Utility** VIs and one for the **Advanced Analog Input** VIs.

The Easy I/O VIs consist of high-level VIs that perform basic analog input, analog output, and digital I/O. They are ideal for getting started with DAQ in LabVIEW. The Easy I/O VIs include a simplified error-handling method so that when a DAQ error occurs in your VI, a dialog box shows error information. When the box appears, you will have the option to halt execution of the VI or to ignore the error.

Compared to the Easy I/O VIs, the Intermediate VIs have more hardware functionality and flexibility. They feature capabilities that the Easy I/O VIs lack, such as external timing and triggering. As you become more familiar with

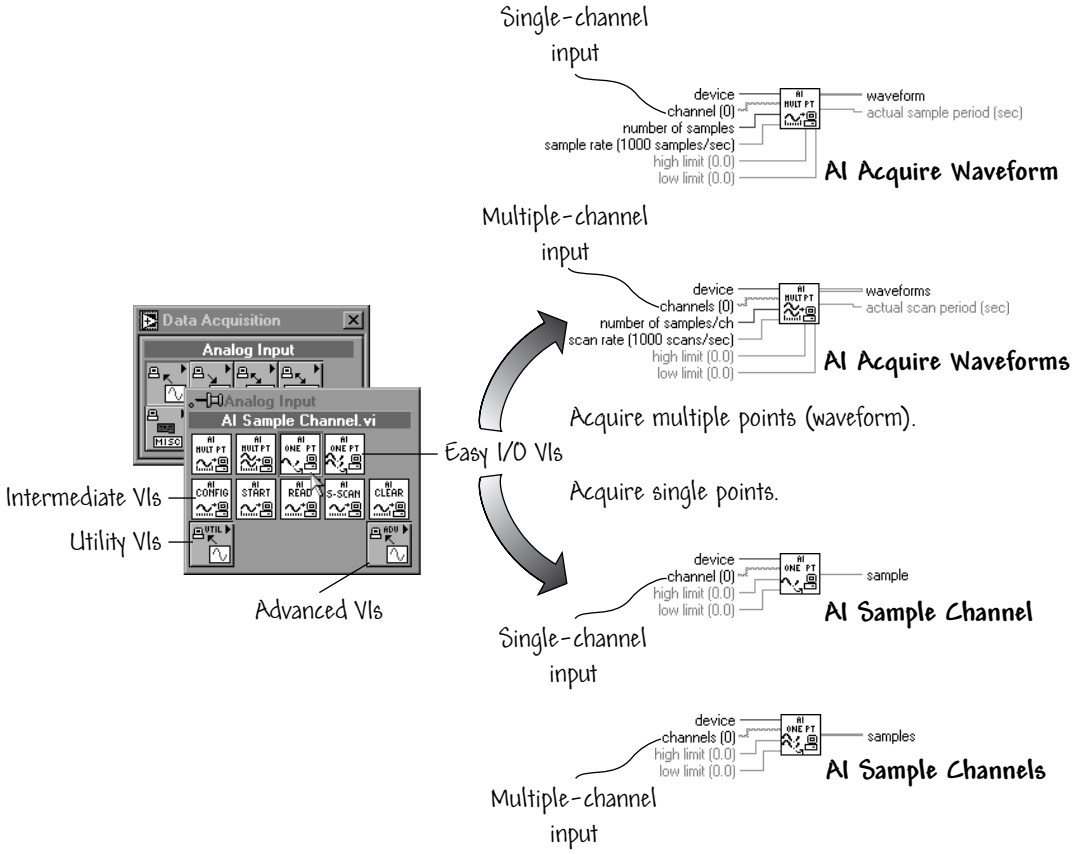


FIGURE 8.30
The **Analog Input** palette.

LabVIEW, you will discover that the Intermediate VIs are better suited for most of your applications. The Intermediate VIs feature more flexible error handling than the Easy I/O VIs in that you can pass error status information to other VIs and handle errors programmatically. The Utility VIs consist of convenient groupings of the Intermediate VIs. They will be helpful in situations where you need more control than the Easy I/O VIs provide.

The Advanced VIs are the lowest-level VIs and are required by few applications. In general, the Easy I/O and Intermediate VIs will suffice for most DAQ applications. However, with the Advanced VIs you have more control over the DAQ board operation, and you have access to more status information from the software driver.

The **Analog Output** palette is shown in Figure 8.31. The organization of the palette is similar to the **Analog Input** palette: the top tier of VIs contains Easy I/O VIs, and the bottom tier contains Intermediate VIs. There are also two subpalettes: one for the **Analog Output Utility** VIs and one for the **Advanced Analog Output** VIs.

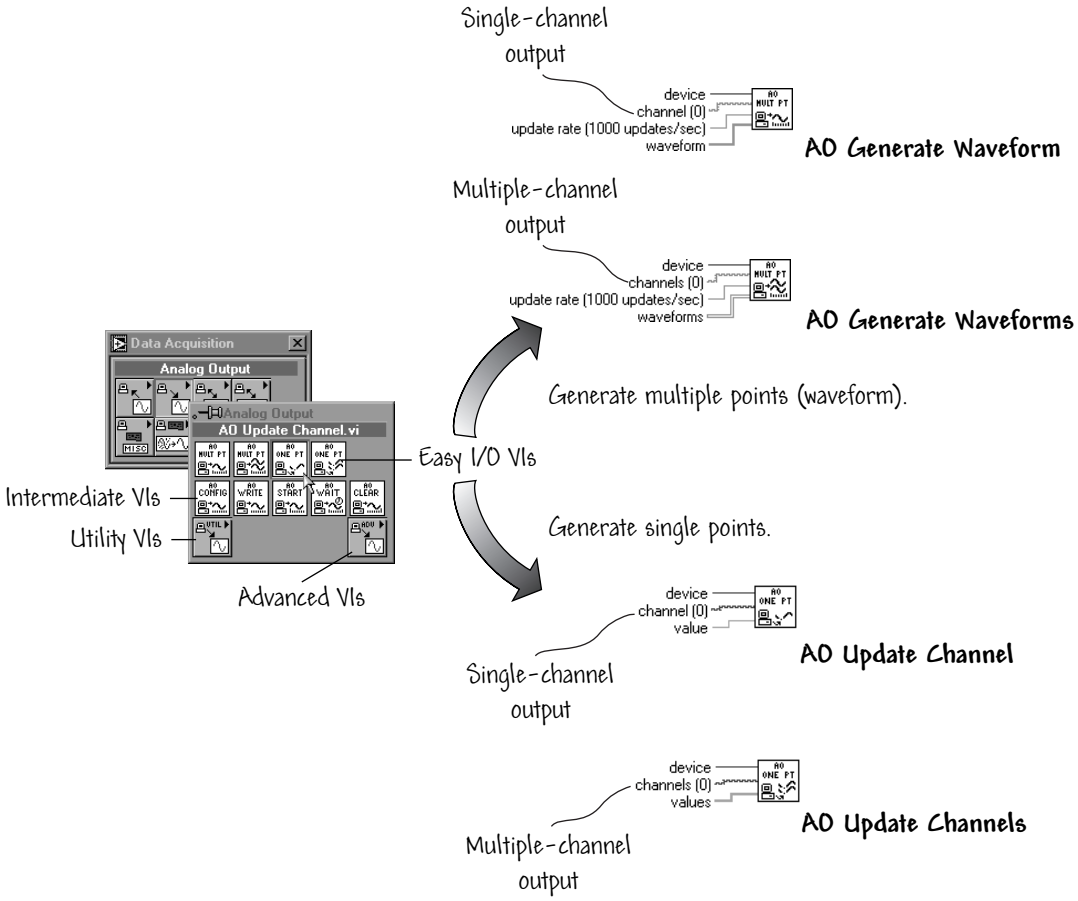


FIGURE 8.31
The **Analog Output** palette.

The **Digital I/O** palette is shown in Figure 8.32. The top tier of VIs contains Easy I/O VIs, and the bottom tiers contain Intermediate and Advanced VIs.

8.7 DAQ HARDWARE CONFIGURATION

LabVIEW provides utilities designed to help you define which signals are connected to which channels on your data acquisition board. In previous years, significant amounts of time were spent defining the signal types, connections, and transducer equations—and all this before beginning development of the actual DAQ system! For example, if you are using thermocouples, you must perform cold-junction compensation (CJC) calculations and apply appropriate scaling factors to convert raw measured voltages into actual temperature readings. This process is now one of entering the necessary information in dialog boxes to define

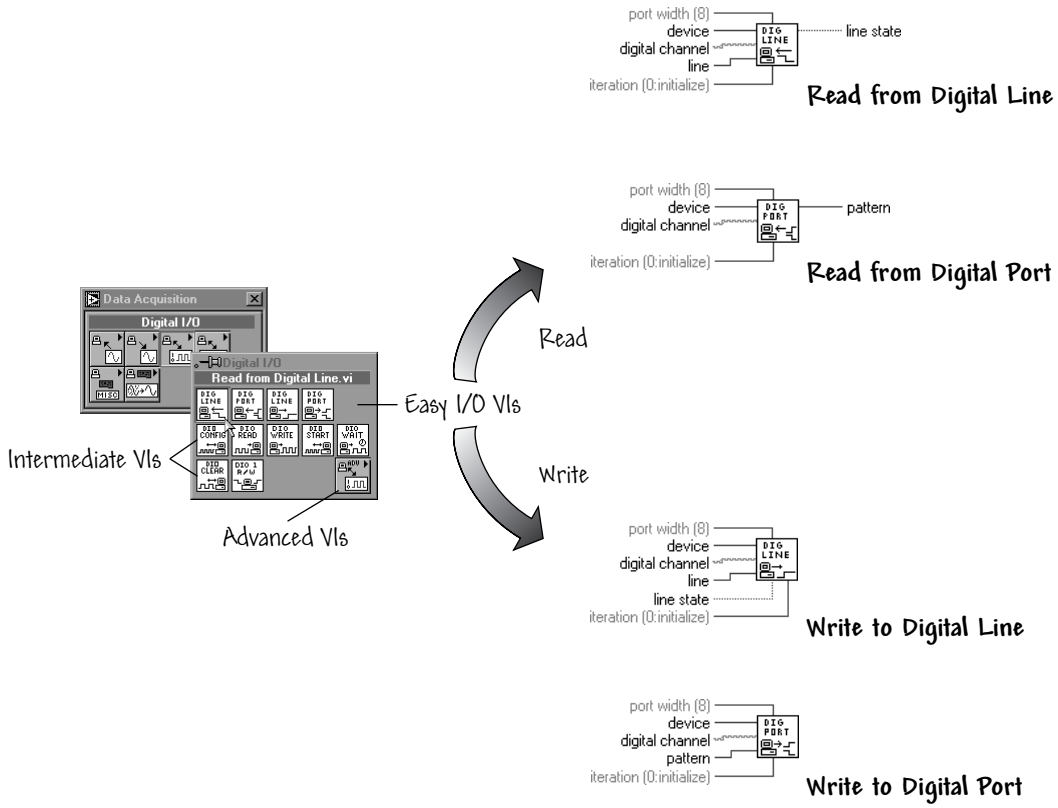


FIGURE 8.32
The **Digital I/O** palette.

an input signal, the type of transducer being used, any scaling factors required, CJC values, and the conversion factors. You can then reference the channel name (that you assign) for the input signal and the conversion from voltage to physical units is performed automatically (and transparently). You can save different configuration files for different settings or systems. Not only can you assign **channel names**, sensor types, engineering units, and scaling information to each channel using the LabVIEW utility, but you can also define the physical quantities you are measuring on each DAQ hardware channel. Once the software has been properly configured, the hardware will be configured correctly to make the measurement for each channel in terms of the physical quantity.

8.7.1 Windows

LabVIEW for Windows installs a configuration utility for establishing all board and channel configuration parameters. This utility is known as the **Measurement & Automation Explorer**, or MAX for short. After installing a DAQ board in

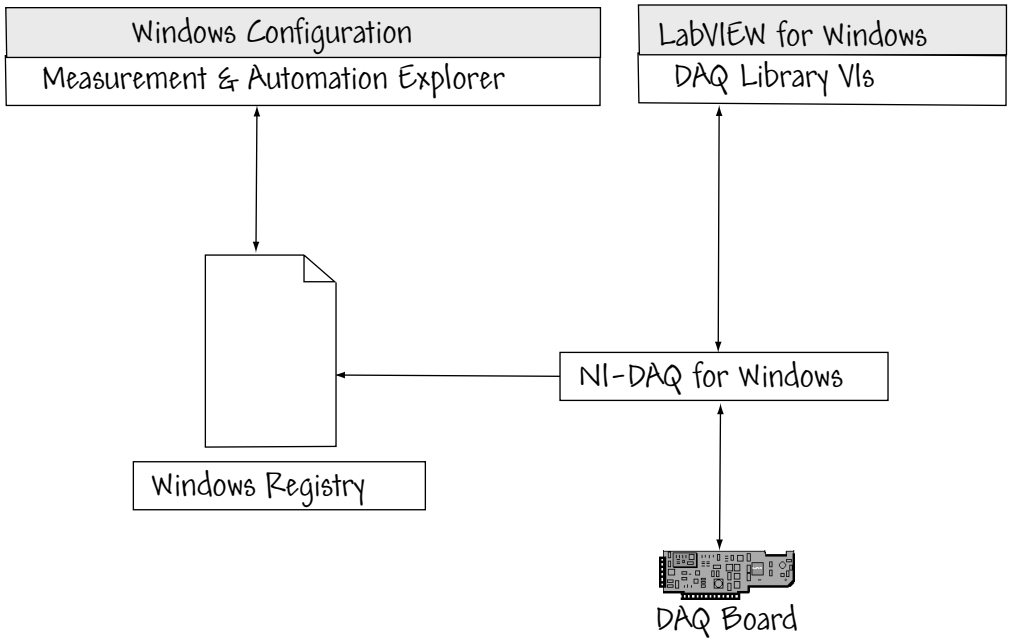


FIGURE 8.33
Windows configuration management.

your computer, you must run this configuration utility. The MAX utility reads the information the Device Manager records in the Windows registry and assigns a logical device number to each DAQ board. You use the device number to refer to the board in LabVIEW. Figure 8.33 shows the relationship between the DAQ board and MAX. The Windows Configuration Manager keeps track of all the hardware installed in your system, including National Instruments DAQ boards.

If you have a Plug & Play (PnP) board, the Windows Configuration Manager automatically detects and configures the board. If you have a non-PnP board (known as a Legacy device) you must configure the board manually using the ***Add New Hardware*** *option under the Control Panel.*

You can check the Windows Configuration by accessing the Device Manager (Start≫Settings≫Control Panel≫System≫Device Manager). You will find **Data Acquisition Devices**, which lists all DAQ boards installed in your computer as shown in Figure 8.34. Highlight a DAQ board and select **Properties** or double-click on the board, and you see a dialog window with tabbed pages. **General** displays overall information regarding the board. You use **Resources** to specify the system resources to the board such as interrupt levels, DMA, and base address for software configurable boards. **NI-DAQ Information** specifies

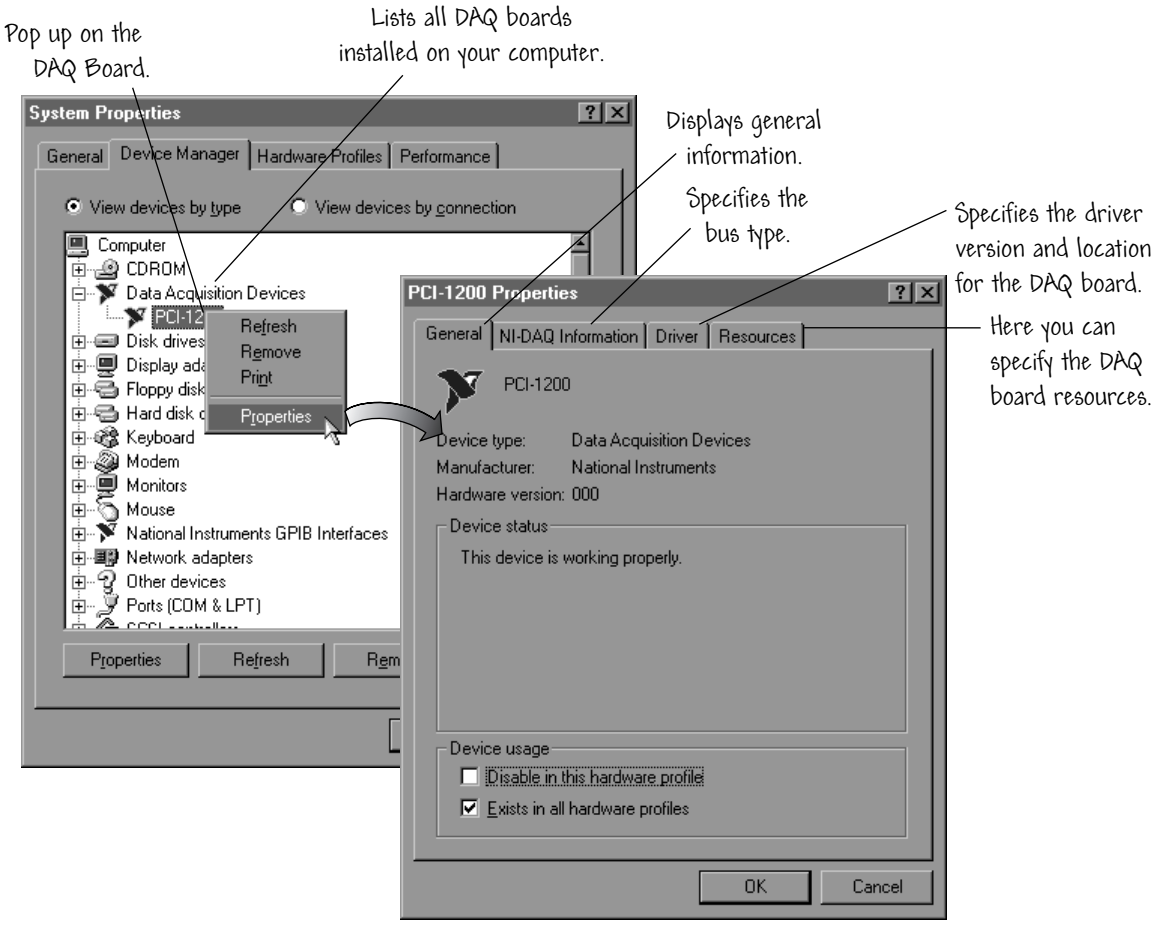


FIGURE 8.34
Checking the Windows Configuration by Accessing the Device Manager.

the bus type of your DAQ board. **Driver** specifies the driver version and location for the DAQ board.

LabVIEW for Windows DAQ VIs access the National Instruments standard NI-DAQ for Windows 32-bit dynamic link library (DLL). The LabVIEW setup program installs the NI-DAQ DLL in the WINDOWS\SYSTEM directory. NI-DAQ for Windows supports all National Instruments DAQ boards and SCXI.

The nidaq32.dll file, the high-level interface to your board, is loaded into the Windows\System directory. The nidaq32.dll file then interfaces with the Windows Registry to obtain the configuration parameters defined by the MAX. You access MAX either by double-clicking on its icon on the desktop or selecting **Measurement & Automation Explorer** from the **Tools** menu in LabVIEW, as illustrated in Figure 8.35.

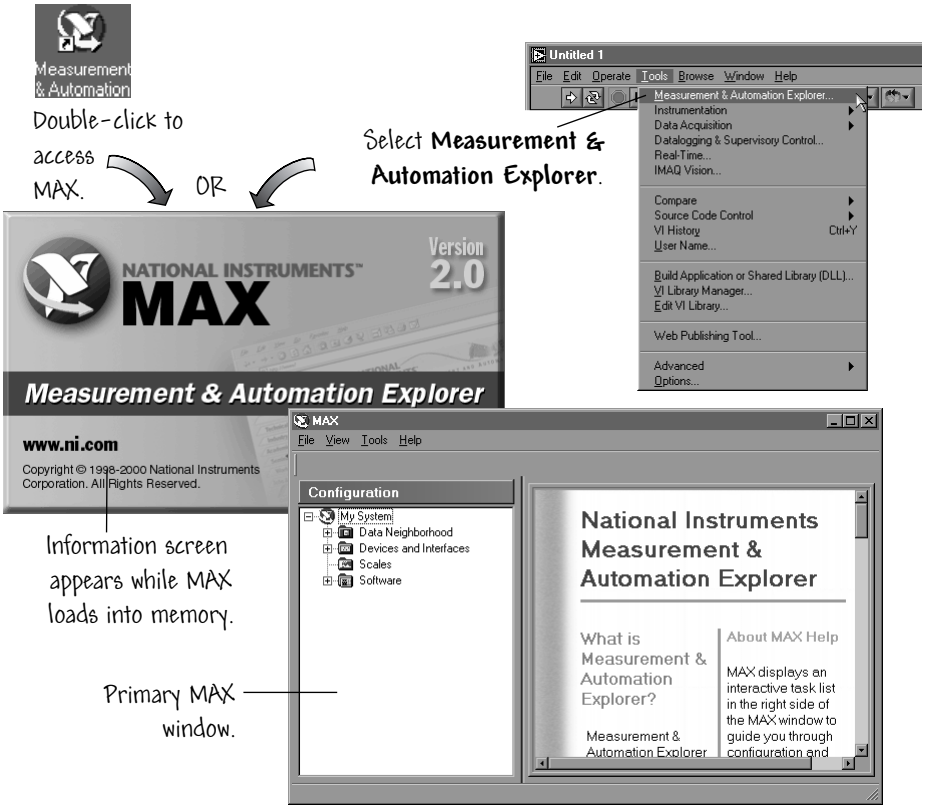


FIGURE 8.35
Accessing the Measurement & Automation Explorer (MAX).

8.7.2 Macintosh

The LabVIEW installation program installs the NI-DAQ for Macintosh software drivers necessary to communicate with National Instruments DAQ boards. You use the NI-DAQ Configuration utility to configure your DAQ board and accessories. Figure 8.36 shows the relationship between the DAQ boards and the NI-DAQ Configuration utility.

When you install NI-DAQ for Macintosh, install version 4.9 if you have an NB or a Lab Series board. Otherwise, install NI-DAQ version 6.6 or later for the PCI and DAQCard boards.

Using the MAX (Windows)

The objective of this exercise is to use the MAX to examine the configuration for the DAQ board in you computer and to configure one virtual channel.

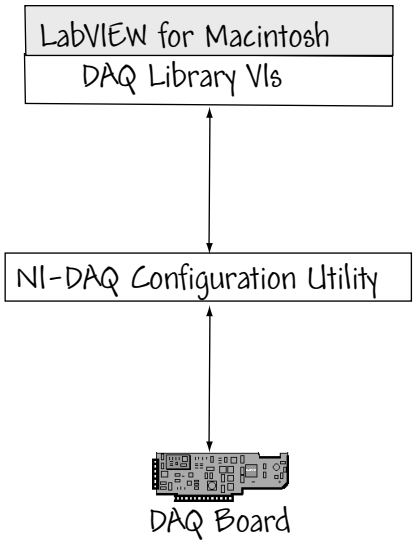


FIGURE 8.36
Macintosh configuration management.

Start MAX by double-clicking on the desktop icon or by selecting **Measurement & Automation Explorer** from the **Tools** menu in LabVIEW, as illustrated in Figure 8.35.

Depending on your system, MAX may be installed in a different location. On the Macintosh, the NI-DAQ configuration utility will be in a separate folder on your hard drive.

Open the Devices and Interfaces section as seen in Figure 8.37. The figure shows what MAX looks like for a PCI-1200 and a PCI-GPIB. The MAX window shows the National Instruments boards and software in your system. Note the Device number indicated in the parantheses after the DAQ board. The LabVIEW DAQ VIs use this Device number to determine which board performs DAQ operations. In Figure 8.37, we see that the DAQ board PCI-1200 is Device 1. You may have a different board installed and some of the options shown may be different. You can get more information about the board's configuration by examining its properties. With the DAQ board highlighted, select the **Properties** button just below the menu. A configuration window for the multiple input/output (MIO) board is shown in Figure 8.37.

The simulated DAQ is not configurable! To use the MAX you must have real hardware installed. If you are using simulated DAQ you will not see entries in the MAX for a simulated board or a data neighborhood. There are predefined

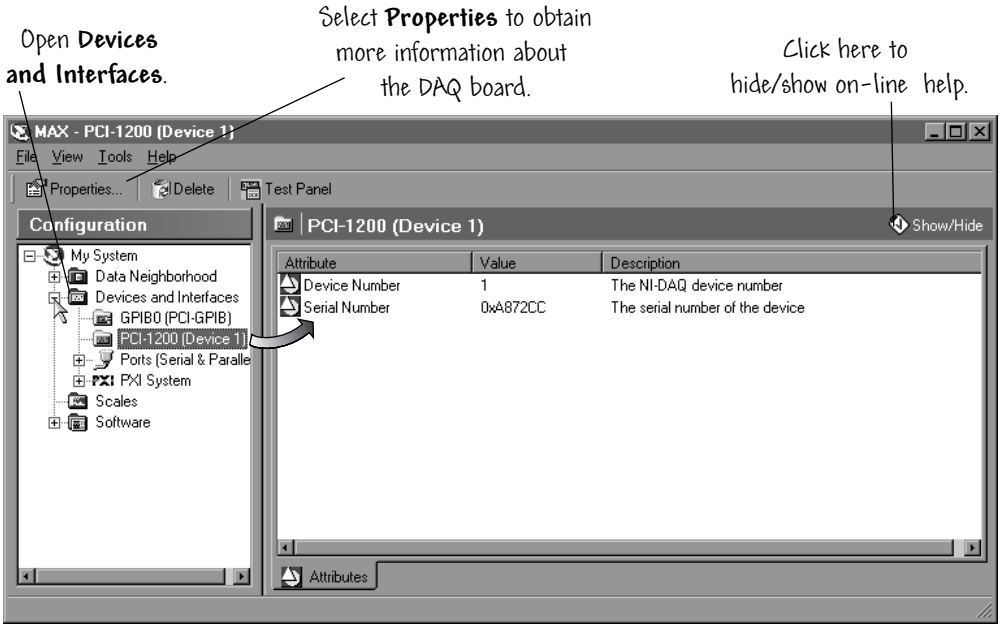


FIGURE 8.37
Checking the properties of the DAQ boards.

virtual channels which you can use in DAQ VIs, however these virtual channels cannot be modified.

*Press the **Show/Hide** button in the top right corner of the MAX window to hide the online help and show the DAQ board information.*

The configuration window depicted in Figure 8.38 contains several tabs. **System** reports the system resources assigned to the board through the Windows registry. The remaining tabs are used to configure the various analog input, output, and accessory parameters for the DAQ board. The **Test Resources** button tests the system resources assigned to the board according to the Windows Device Manager. Close the **Configuring Device 1:PCI-1200** window and refer the MAX window. Next, we want to configure a virtual channel. A virtual channel is a shortcut to configuring a channel which allows you to give descriptive names to physical channels. Right-click on the Data Neighborhood icon and choose **Create New...**. Select **Virtual Channel** and press the **Finish** button, as depicted in Figure 8.39. You will now configure a channel to take a reading from the temperature sensor (Analog Input Channel 0). After pressing the **Finish** button, a window for configuring a new channel will appear, and you can enter the virtual channel name as shown in Figure 8.40. Name the channel "Temperature" and add a short description as seen in Figure 8.41. Each new page of the setup is accessed by clicking on the **Next** button, and you can always go back to a previous page using the **Back** button. Once you have named the virtual channel, select **Next** and

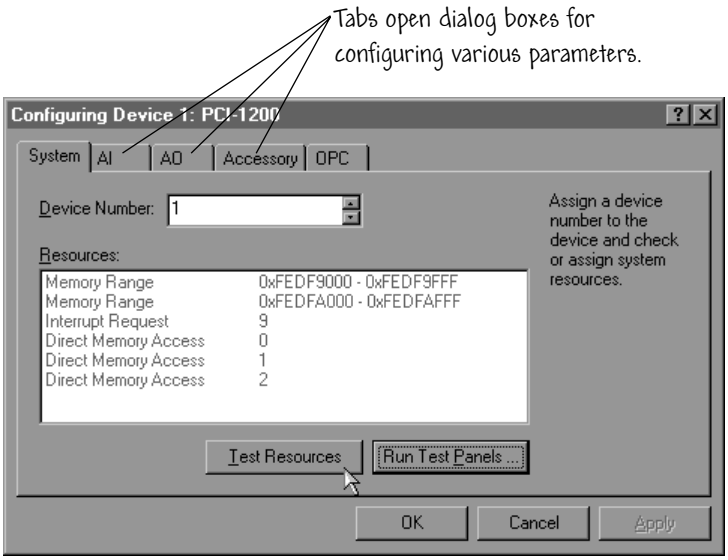


FIGURE 8.38
DAQ board properties.

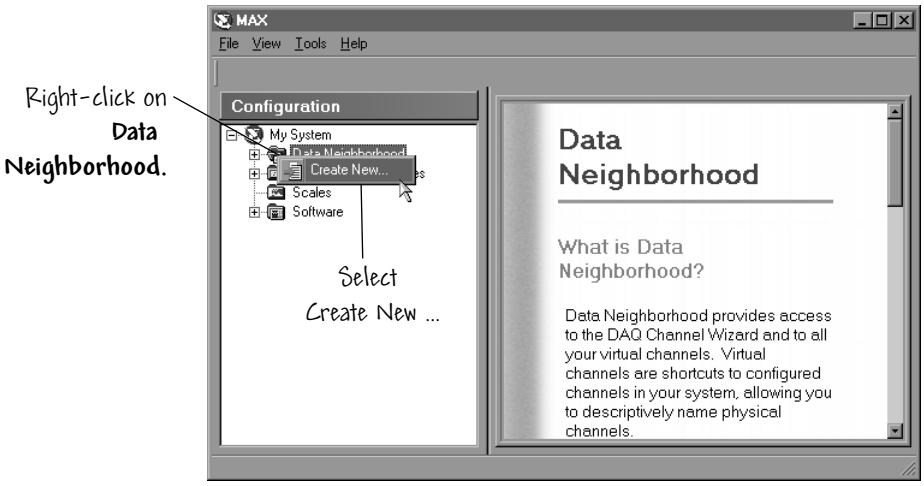


FIGURE 8.39
Configuring a new virtual channel.

move to the sensor type, as illustrated in Figure 8.42. You have many choices of sensor types, including voltage, current, resistance, frequency, thermocouple, RTD and accelerometer. In our example, we choose voltage. Once you have defined the physical quantity of the temperature sensor—click on the **Next** button. If your temperature sensor generates a voltage that is scaled to Deg C, you input the unit of measurement as Deg C, as shown in Figure 8.43. In this exercise use Deg C as the unit of measurement. The temperature range of 0 to 100° C may

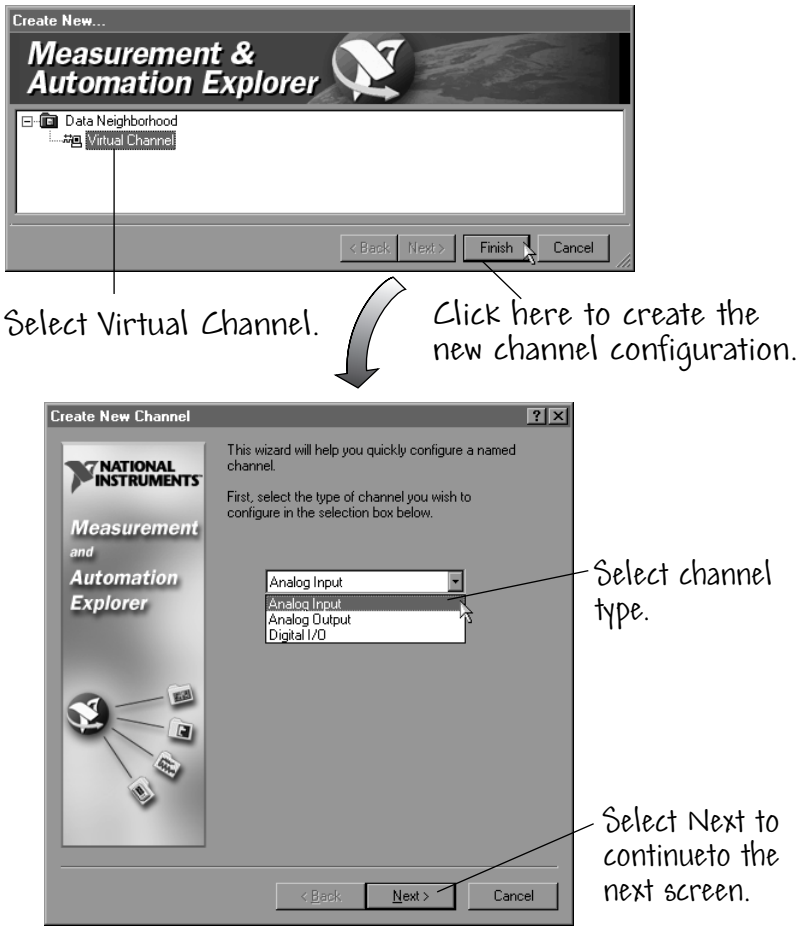


FIGURE 8.40
Creating a new virtual channel configuration.

be a wide range for measuring temperature. In general, you may need to expand (or reduce) the range depending on your application.

Click on the **Next** button to configure the scaling values for the temperature sensor. Select **Map Ranges** as the signal scale, as illustrated in Figure 8.44. This implies that you want to set the scaling range. Set the sensor range to 0 V to 1 V, and when finished with the input, click on **Next**.

You should now have arrived at the window that defines the hardware connection for the signal. The settings should match those shown in Figure 8.45, with the exception that the DAQ hardware should reflect the actual hardware that you have in your computer. For example, if you have a Lab-PC+ board, you should select that hardware in place of the PCI-1200 shown in Figure 8.45. You should select channel 0 in item 2 if the temperature sensor is connected to channel 0 of your DAQ board; otherwise select the channel number that you have wired to

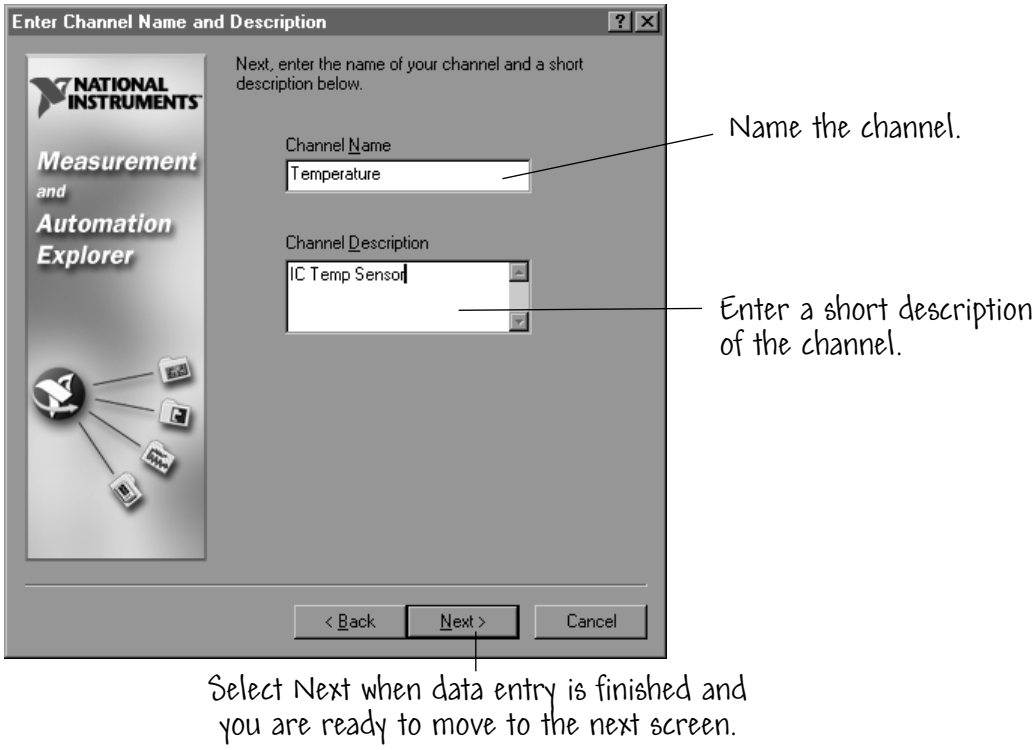


FIGURE 8.41
Naming the virtual channel and adding a short description.

your DAQ hardware. The analog input mode is also selected on this page. Refer to the previous discussion on differential, RSE, and NRSE modes for more information.

Click on the **Finish** button to view the channel you just defined. When you are finished defining the channel, MAX entries should appear as shown in Figure 8.46. In future sessions, you can edit the configurations through the MAX by highlighting the desired configuration and right-clicking on **Temperature**, as illustrated in Figure 8.47. Choose **Properties** to access the editing window. An editing window will appear and you can modify any of the configuration parameters. When you are finished configuring the virtual channel close the MAX window to exit.

◆

8.7.3 DAQ Channel Name Control

The DAQ Channel Name control is a LabVIEW data type used by the DAQ VIs for communicating to the DAQ boards. You can find the DAQ Channel Name control on the **I/O** subpalette of the **Controls** palette as shown in Figure 8.48.

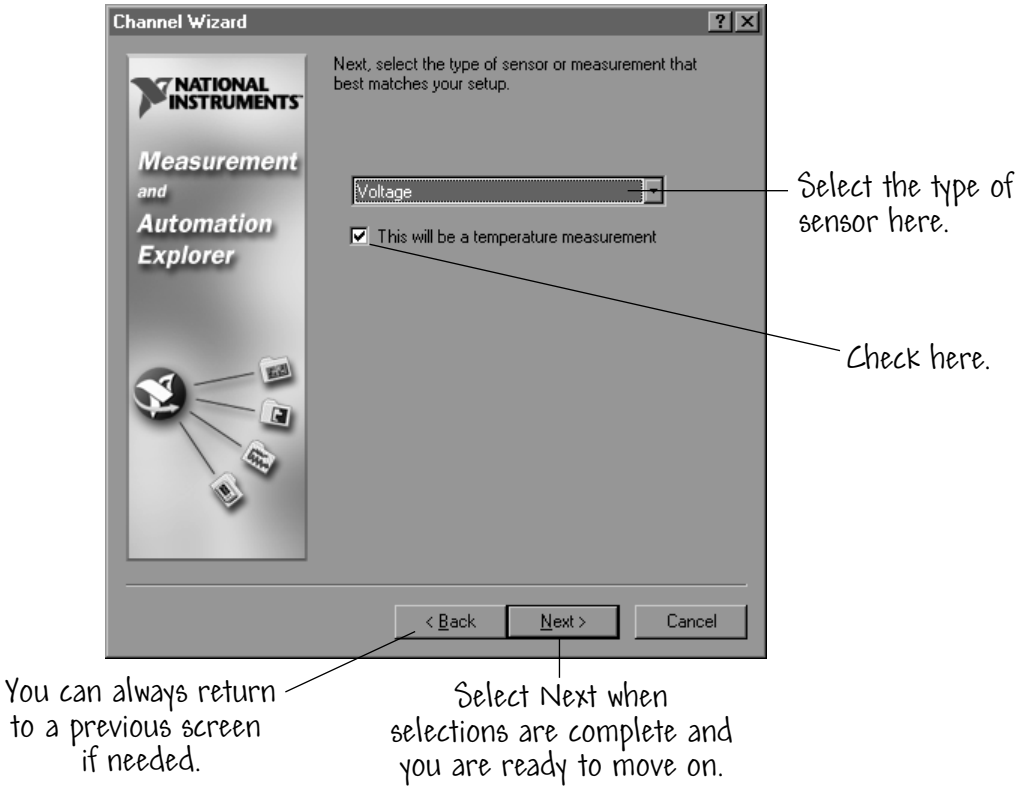


FIGURE 8.42
Selecting the sensor type.

The channel names can be entered into the control in two different ways. You can use the Operating tool to click on the DAQ Channel Name control and choose the channel name as defined in the MAX utility as shown in Figure 8.49. Another way to enter the DAQ channel name is to pop-up on the DAQ Channel Name control and select **Allow Undefined Names**. Then you can use the Labeling tool to enter the name or the channel number. The name will be one of the names of the virtual channels configured using the MAX, as discussed in the previous section.

8.7.4 The DAQ Wizards

The **DAQ Solution Wizard** helps you develop your LabVIEW applications by allowing you to choose among current data acquisition examples or to design a custom DAQ application. It works with analog input and output, digital I/O, and counter/timers. The DAQ Solution Wizard is an interactive utility that uses a series of windows that ask you about your applicaiton and creates an example

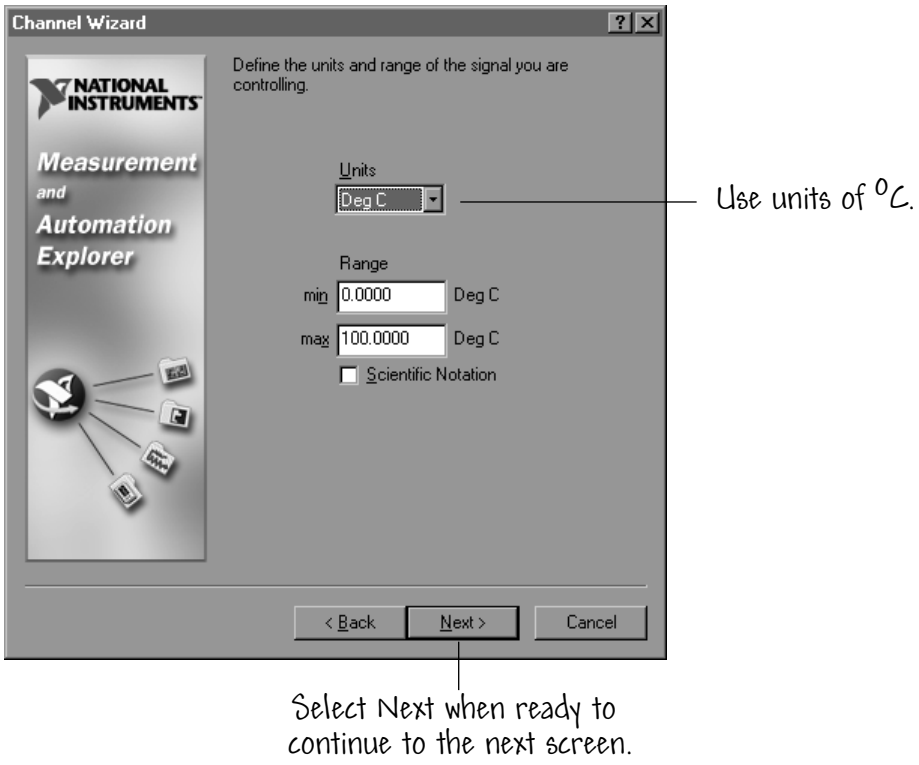


FIGURE 8.43
Defining the sensor units and scaling.

VI that you can save to a new location. The DAQ Solution Wizard is launched by selecting **DAQ Solution Wizard** from the Tools≫Data Acquisition menu in LabVIEW, as shown in Figure 8.50. You can see the virtual channels defined in the MAX by clicking on the **View Current Wizard Configuration** button. You can look at the channel definitions in more detail by clicking on the **Go to DAQ Channel Wizard** button to launch. The **DAQ Channel Wizard** is used to define which signals are connected to which channels on your DAQ board. The MAX utility opens when you press the **Go to DAQ Channel Wizard** button. You can then modify or add new virtual channels and scales for your data acquisition application. Subsequently, you can reference the channel names throughout the application, and all of the conversion processes are performed transparently.

In the open DAQ Solution Wizard window, make sure that the option **Use channel names specified in DAQ Channel Wizard** is selected. Selecting the **Next** button provides the option of either making a custom application of viewing the VIs in the Common Solutions Gallery. Select **Solutions Gallery** and press the **Next** button yielding the window shown in Figure 8.51.

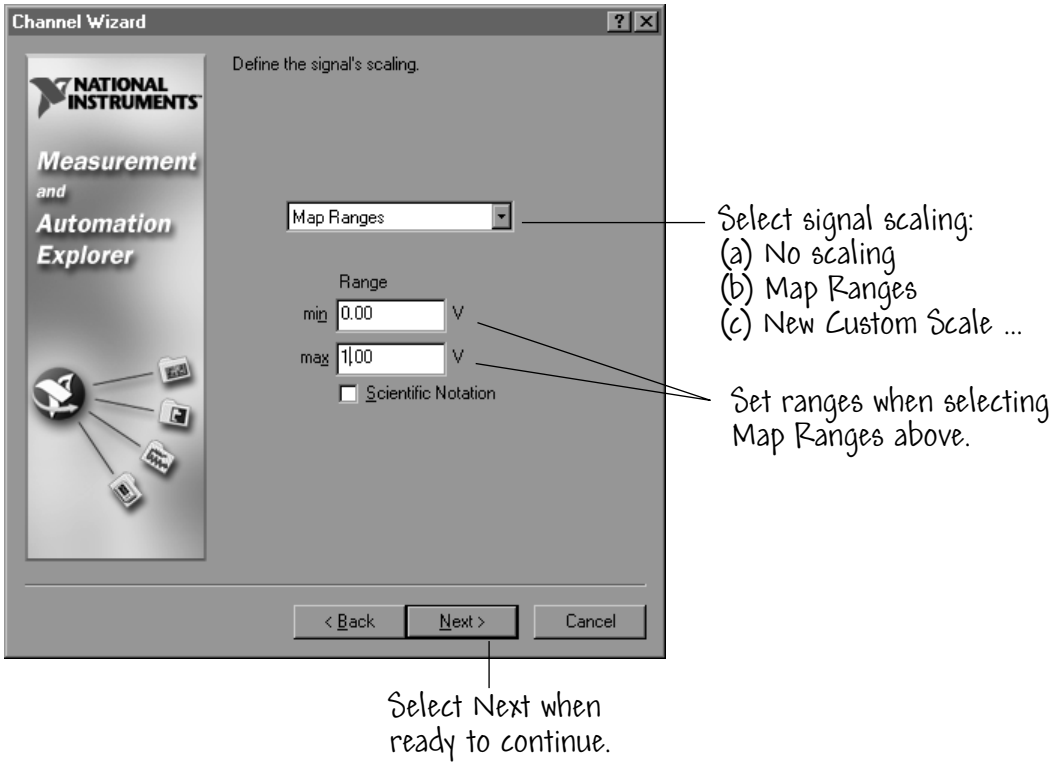


FIGURE 8.44
Selecting the signal scaling.

Select **Voltage & Current Measurement** from the Gallery Categories section and **Single-Point Voltage Measurement** from Common Solutions. When you press the **Next** button, you will see the available channels. Select the analog input channels by holding down <Shift> and clicking on the choice as illustrated in Figure 8.52. Clicking on the button will yield the VI shown in Figure 8.53.

Notice that the channels are already defined. Open and examine the block diagram. It uses the **AI One PT** VI to acquire the data. You can modify this VI just as you might any other VI. You will need to save the modified VI to hard disk if you want to utilize it in future sessions.

8.8 ANALOG INPUT

There are four classes of Analog Input VIs found in the **Analog Input** palette:

1. Easy Analog Input VIs
2. Intermediate Analog Input VIs

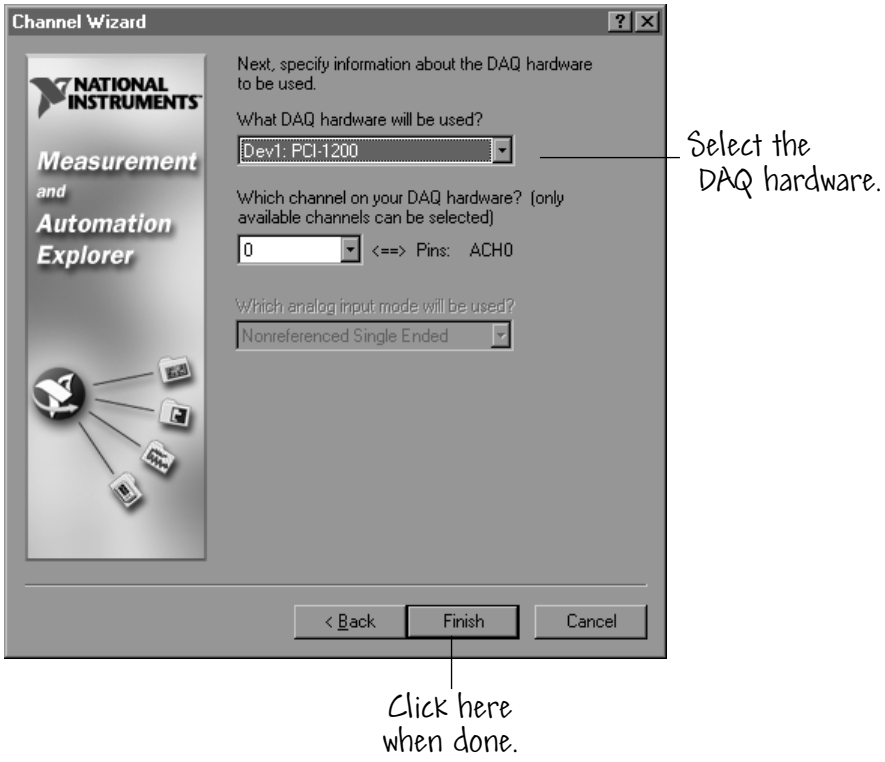


FIGURE 8.45
Selecting hardware.

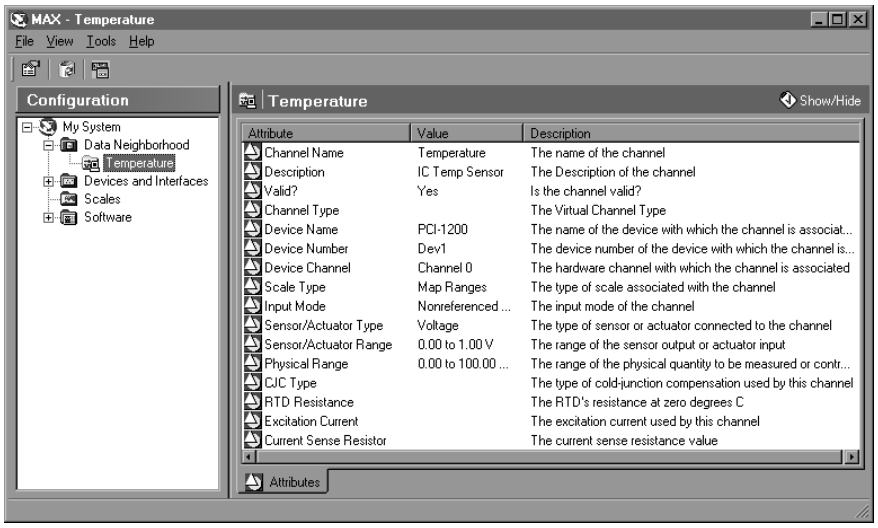


FIGURE 8.46
The virtual channel final configuration.

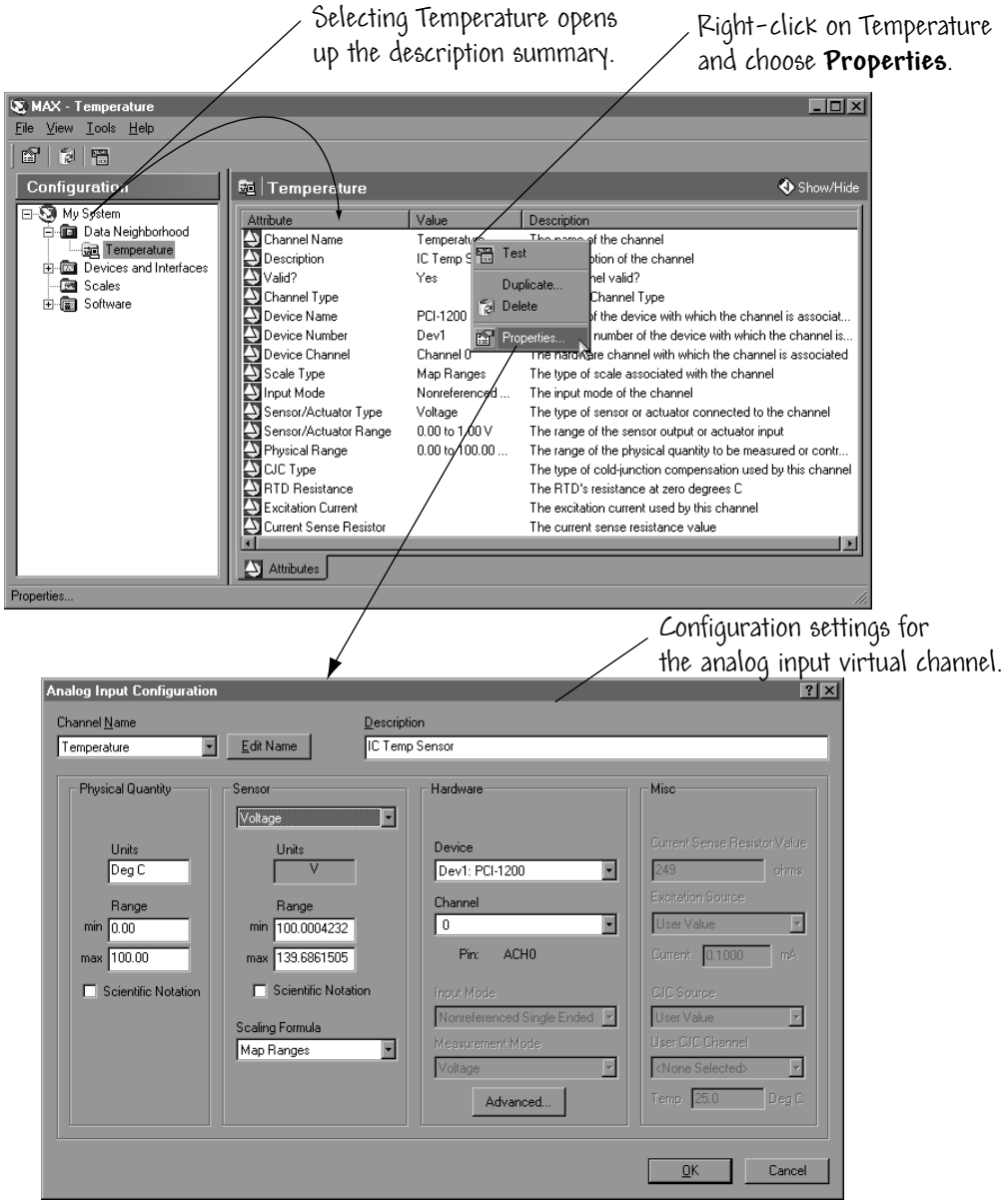


FIGURE 8.47
Editing the configuration.

3. Analog Input Utility VIs, and
4. Advanced Analog Input VIs

In *Learning with LabVIEW*, we consider only the use of the Easy Analog Input VIs, which perform simple analog input operations. You can run these VIs from the front panel or use them as subVIs in basic applications.

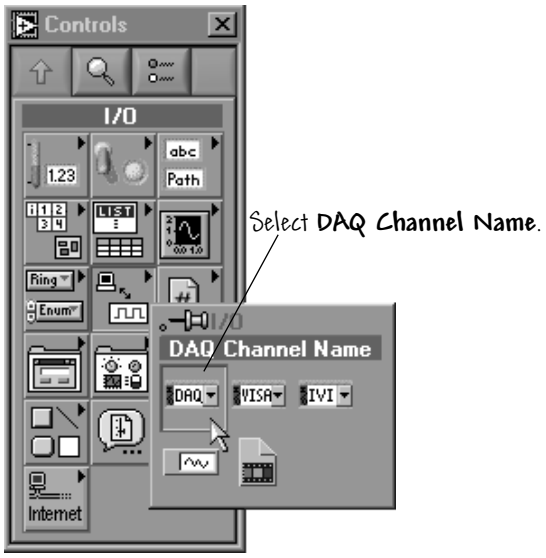


FIGURE 8.48
DAQ channel name control data type.

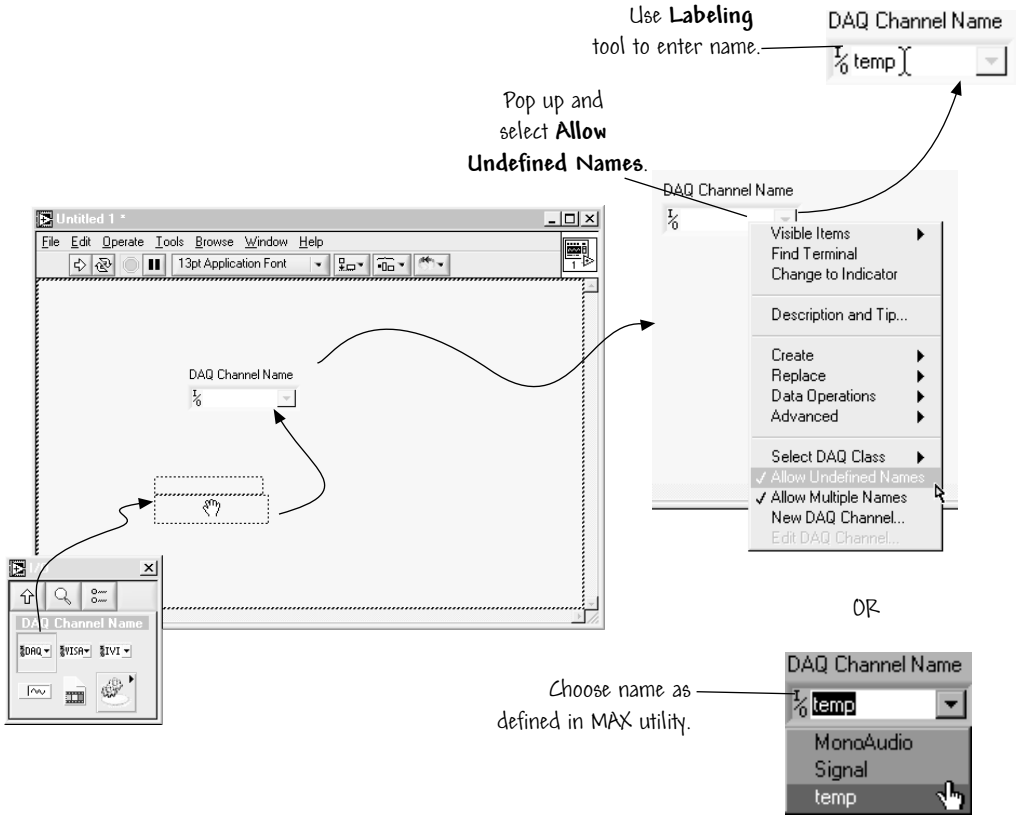


FIGURE 8.49
Entering DAQ channel names.

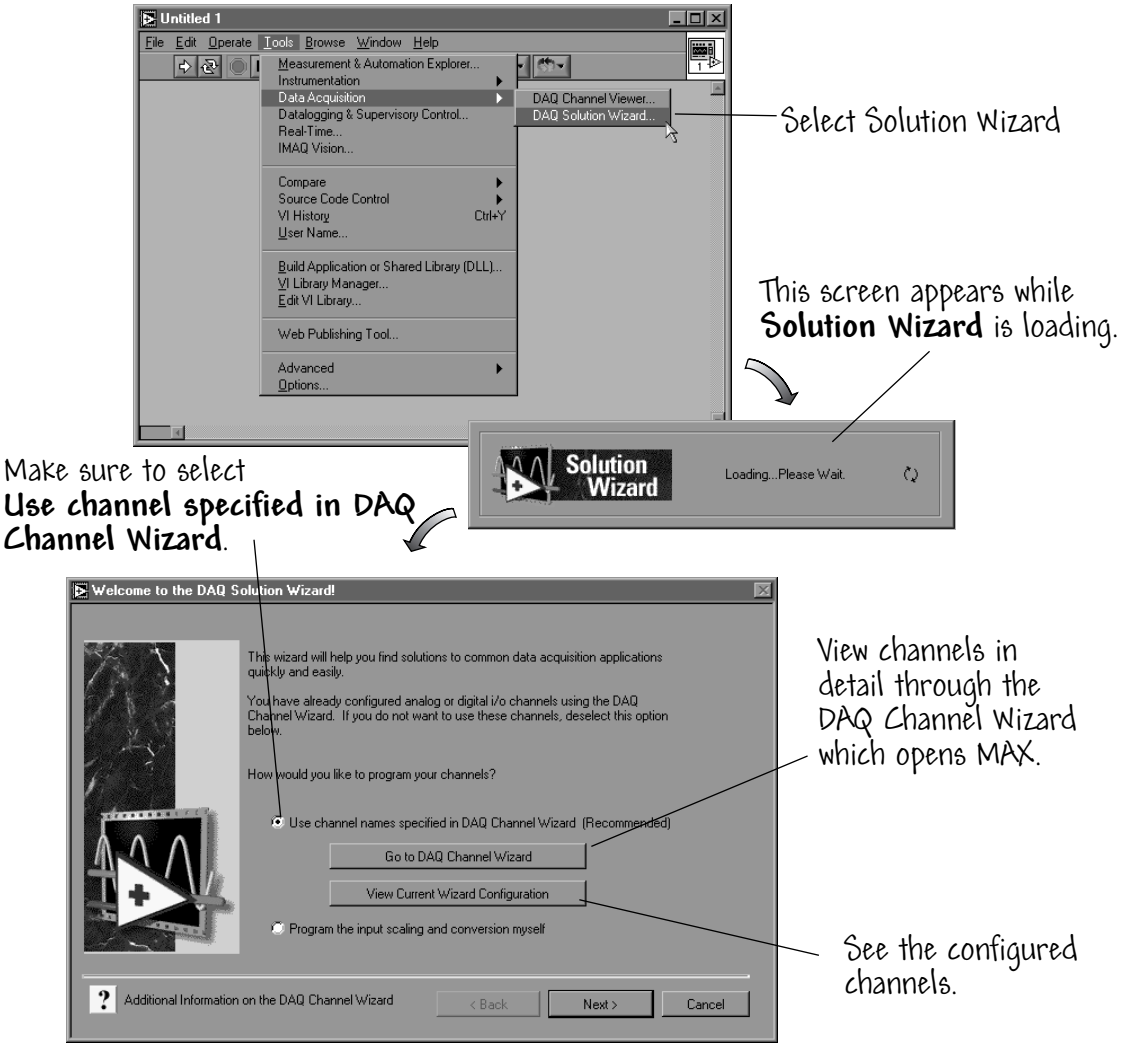


FIGURE 8.50
Accessing the DAQ Solution Wizard.

In this chapter, the exercises show real DAQ VIs with channels that have been configured using the Measurement and Automation Explorer. If you installed the simulation DAQ VIs, you need to wire values to the device and channel inputs, and you do not need to run the Measurement and Automation Explorer. For exercises that use one channel, enter 1 for device and 0 for channel. For exercises that use two channels, enter 1 for device and 0, 1 for channels. When you move your DAQ application to a machine with real VIs, you can continue to use numeric values for device and channel inputs, or you can run MAX and assign

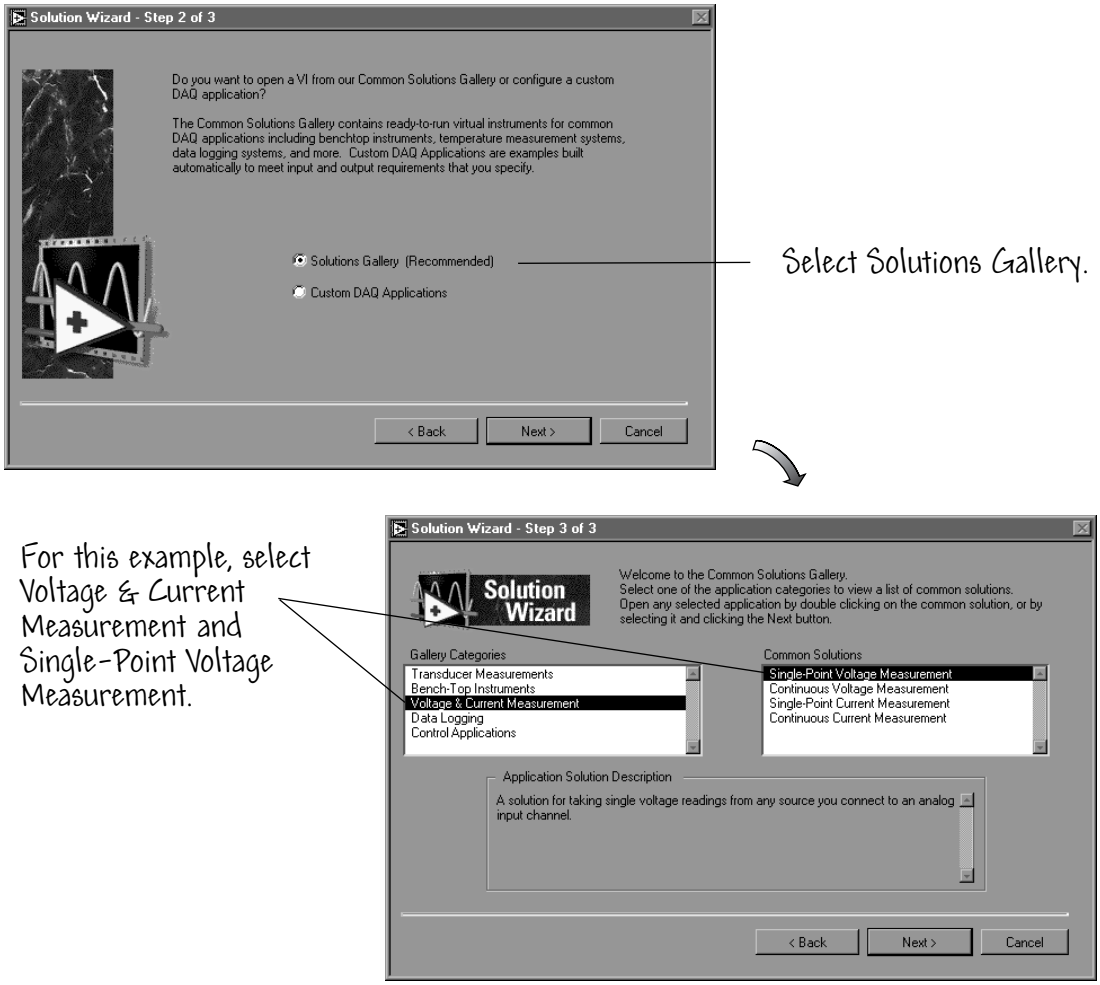


FIGURE 8.51
Accessing the Solutions Galley.

names to channels. When you use channel names configured with the MAX, you do not need a device value.

8.8.1 Single-Point Acquisition

A single-point analog input reads one value from the assigned input channel and immediately returns the value to the VI. This type of data acquisition is useful when you want to determine the magnitude of an acquired analog DC signal. An example of this would be if you want to periodically monitor the temperature in a room. You can connect the temperature transducer that produces a voltage (representing the temperature) to a single channel on your DAQ device and initiate

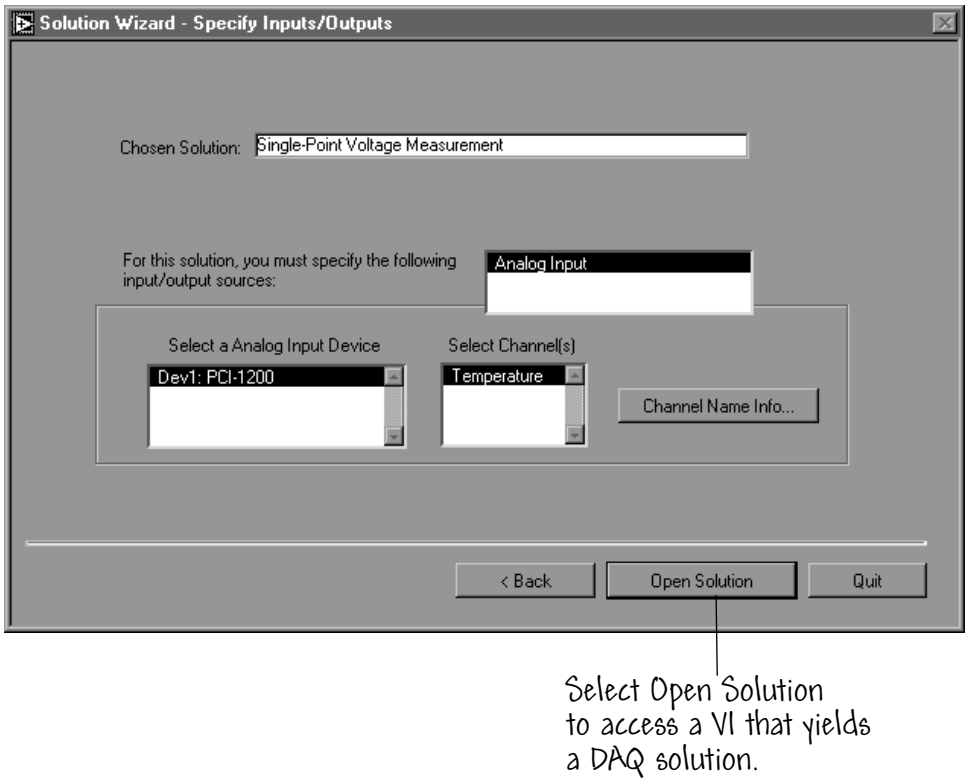


FIGURE 8.52
Selecting the inputs and outputs for the DAQ solution galley.

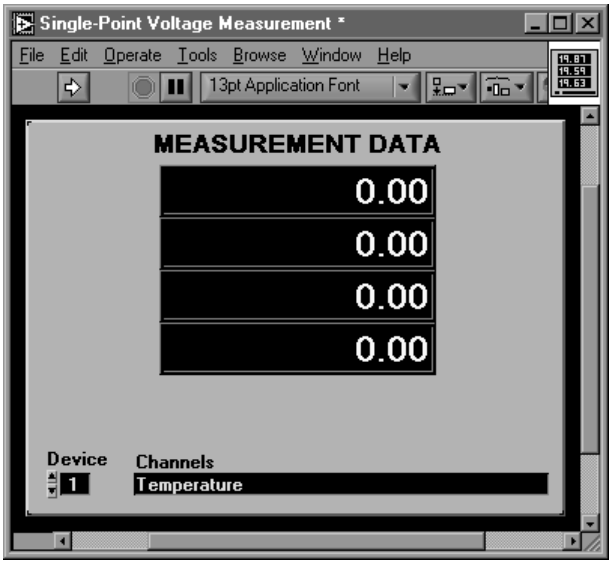


FIGURE 8.53
The DAQ Solution Galley VI.

(through the software) a single-point analog input data acquisition whenever you want to know the room temperature.

The **Analog Input** palette found on the **Data Acquisition** palette contains the VIs that perform the single-point acquisitions and other analog-to-digital (A/D) conversions (see Figure 8.30). The AI Sample Channel.vi is used for single-point acquisition; that is, it takes a single sample of the analog signal attached to a specified channel and returns the measured voltage. The two inputs for the VI are

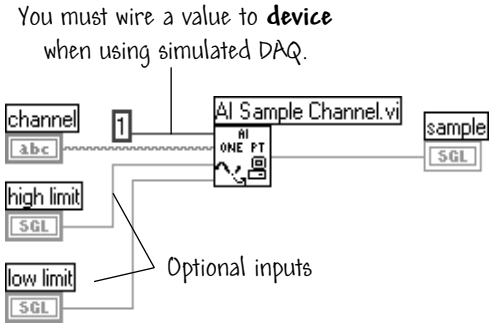
- **Device**—the device number of the DAQ board. When you use a channel name configured with the MAX you do not need a device value. If you are using the simulation DAQ VIs, you must wire values to **device**.
- **Channel** specifies the analog input channel number or a channel name configured with the MAX. The default value is 0.

The other optional inputs are

- High limit and low limit—specify the limit settings of the input signal. The default values for high and low limit are +10 V and −10 V, respectively.

Do not wire values to these inputs if you are using the MAX, because this information has already been configured. Figure 8.54a illustrates how to wire AI Sample Channel.vi.

(a) Don't wire the input **device** when using real DAQ VIs and the DAQ Channel Wizard



(b) Wire the input **device** when using simulated DAQ VIs

FIGURE 8.54
The AI Sample Channel.vi measures the signal attached to the channel you specify.

The output sample shown in Figure 8.54 is a scaled version of the input analog signal. If an error occurs during the operation of AI Sample Channel.vi, a

dialog box displays the error code, and you have the option to abort the operation or to continue execution.

It is possible to acquire single-point data from several analog channels on a DAQ board. This is accomplished using AI Sample Channels.vi, which measures the signals attached to multiple channels and returns those measured values in an array. The VI inputs are the same as for AI Sample Channel.vi, except that in this case, the input channels is a string that specifies the list of analog input channels. The output of the VI is an array of measurements. The order of the measurements in the output array matches the order listed in the input channels string. For example, if channels is "1, 2, 4", samples[0] would be from CH 1, samples[1] from CH 2, and samples[2] from CH 4.

You do not need to enter the device or input limits if you set up your channels using the MAX. Instead, enter a channel name in the channel input, and the value returned is relative to the physical units you specified for that channel in the MAX.

Practice with Single-Point Acquisition

The objective of this example is to acquire an analog signal using a DAQ board. You will build a VI that measures the voltage of a temperature sensor. The temperature sensor outputs a voltage proportional to the temperature and is hardwired to Channel 0 of the DAQ board.

Open a new VI and construct a front panel similar to the one depicted in Figure 8.55. The input variable **channel** is a DAQ Channel Name data type (as discussed in the previous section). Use the **Labeling** tool to input the text **Temperature**. Switch to the block diagram window and place the AI Sample Channel VI on the block diagram. Wire the block diagram using Figure 8.55 as a guide. In this exercise, the VI reads analog input Channel 0 and returns the measured voltage. The DAQ channels should have been configured using the MAX as discussed in the previous section (see Figure 8.46).

When the VI is ready to run, return to the front panel and select **Run**. The thermometer on the front panel will display the measured temperature in deg C. If an error occurs during the data acquisition process, the Easy I/O VIs automatically display a dialog box showing the error code and a description of the error. When you have finished experimenting with your VI, save it as AI Single Point.vi in the Users Stuff directory.

If you are having problems constructing the VI described above, you can find a working version of the VI shown in Figure 8.55 by opening the VI titled AI Single Point.vi located in the Chapter 8 folder in the Learning directory. An example of a VI that acquires single-point data from two analog channels is called AI

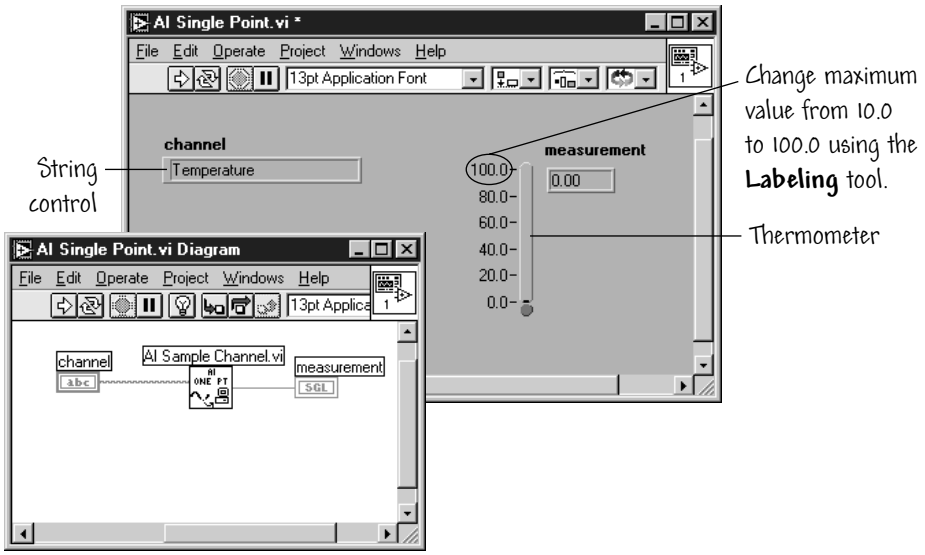


FIGURE 8.55
A VI designed to measure an analog temperature signal.

Single Pt Mult Ch.vi and can be found in the Chapter 8 folder in the Learning directory.

◆

8.8.2 Waveform Data Types

The waveform datatype is a LabVIEW datatype that combines the data read from the DAQ board with time information. The DAQ VIs return waveform data. You can place a waveform on the panel by selecting it from the **I/O** subpalette of the **Controls** palette as shown in Figure 8.56. When you wire the waveform output terminal of a DAQ VI directly to the waveform datatype, you will receive the starting time the data was acquired, the delta time for each data point, and an array of the data values. The waveform datatype can be wired directly to a waveform graph, and it will properly scale the X axis with the time data.

8.8.3 Waveform Acquisition

In some applications, acquiring one point at a time may not be fast enough. Also, with single-point acquisition it is difficult to obtain a constant sample interval between each point because the interval depends on a number of factors (e.g., loop execution speed and software overhead) that are not easily controlled. The AI Acquire Waveform.vi acquires a specified number of samples at a desired sample rate from a single input channel. The VI is shown in Figure 8.57.

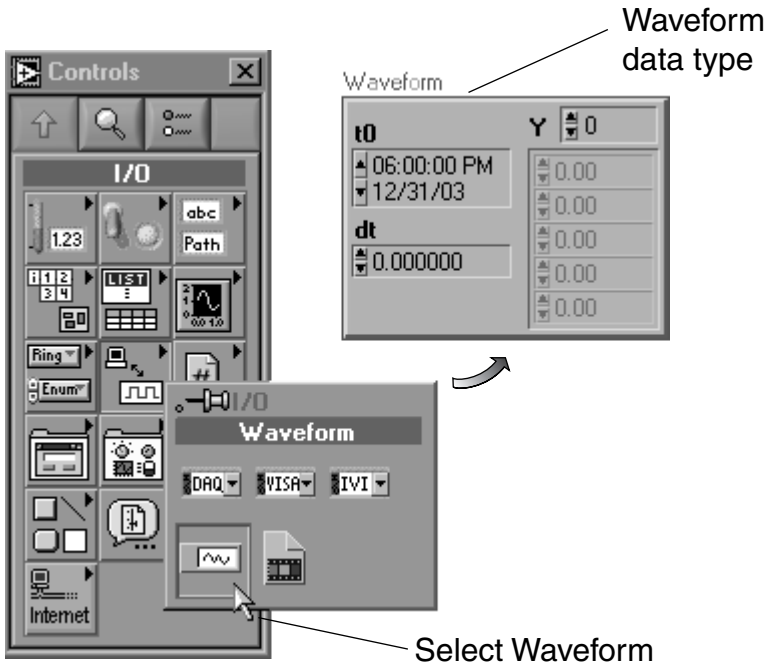
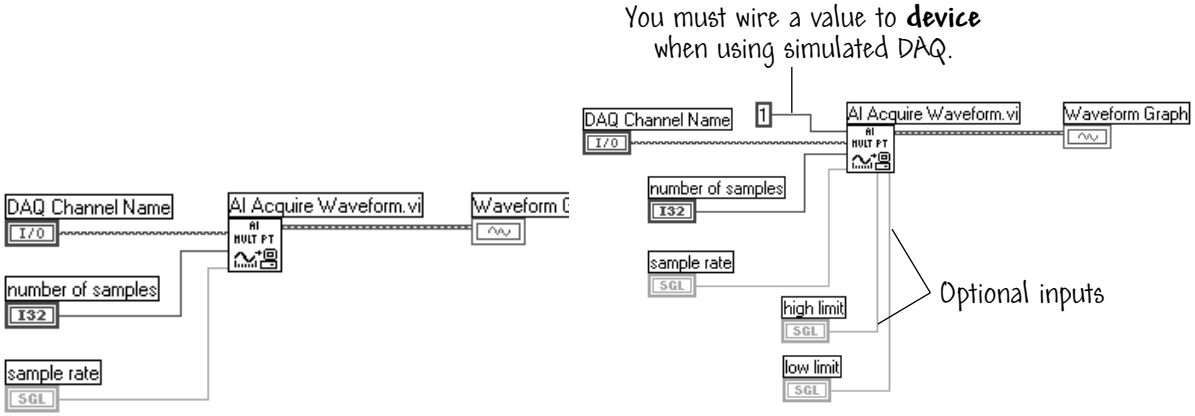


FIGURE 8.56
Waveform data type.



(a) Don't wire the input **device** when using real DAQ VIs and the DAQ Channel Wizard

(b) Wire the input **device** when using simulated DAQ VIs

FIGURE 8.57
The AI Acquire Waveform.vi acquires the specified number of samples at the specified sample rate from a single input channel and returns the acquired data.

Since this is an Easy I/O VI, it has the minimal number of inputs needed to acquire a waveform from a single channel. These nominal inputs are

- **Device**—the DAQ board device number. When you use channel names configured with the MAX, you do not need a device value. If you are using the simulation DAQ VIs, you must wire a value to **device**.
- **Channel** specifies the analog input channel number or a channel name configured with the MAX.
- **Number of samples**—the number of samples to acquire.
- **Sample rate**—the number of samples to acquire per second.

Figure 8.57 illustrates how to wire AI Acquire Waveform.vi. AI Acquire Waveform is a polymorphic VI that can be configured to output a **scaled array** or **waveform**. The outputs of the **AI Acquire Waveform.vi** are

- **Waveform**: contains the scaled analog input data, for both scaled array and waveform outputs.
- **Actual scan period**: the inverse of the actual sampling rate. Depending on the capabilities of your hardware, the actual sample rate may differ slightly from the requested sample rate. This applies to scaled array output only.

Practice with Single-Waveform Acquisition

The objective of this exercise is to acquire and display an analog waveform. The MAX will be used to configure the system foran input called **MonoAudio**. Your VI will use the DAQ VIs to acquire a signal and plot it on a graph. For this exercise, you will need to have an analog input signal available for assignment to channel 1.

Open a new VI and build the front panel shown in Figure 8.58. The input variable **channel** is labeled MonoAudio. The input **number of samples** control specifies the number of points to sample. The input **sample rate** specifies the sampling rate. When the VI is up and running, you can vary these input parameters and observe the effects on the waveform graph. After building the front panel, switch to the block diagram and wire the AI Acquire Waveform VI using Figure 8.58 as a guide.

Now you'll need to configure the DAQ system in preparation for acquiring the analog signal. Open the MAX, as shown in Figure 8.59. We will follow essentially the same steps through the MAX as we did previously in the example "Using the MAX." When the MAX opens up, it should resemble the one shown in Figure 8.46. The goal is to add to the channel configuration another analog input called **MonoAudio** assigned to channel 1. When the MAX screen appears, right-click on **Data Neighborhood** and select **Create New...**. to initiate the process of

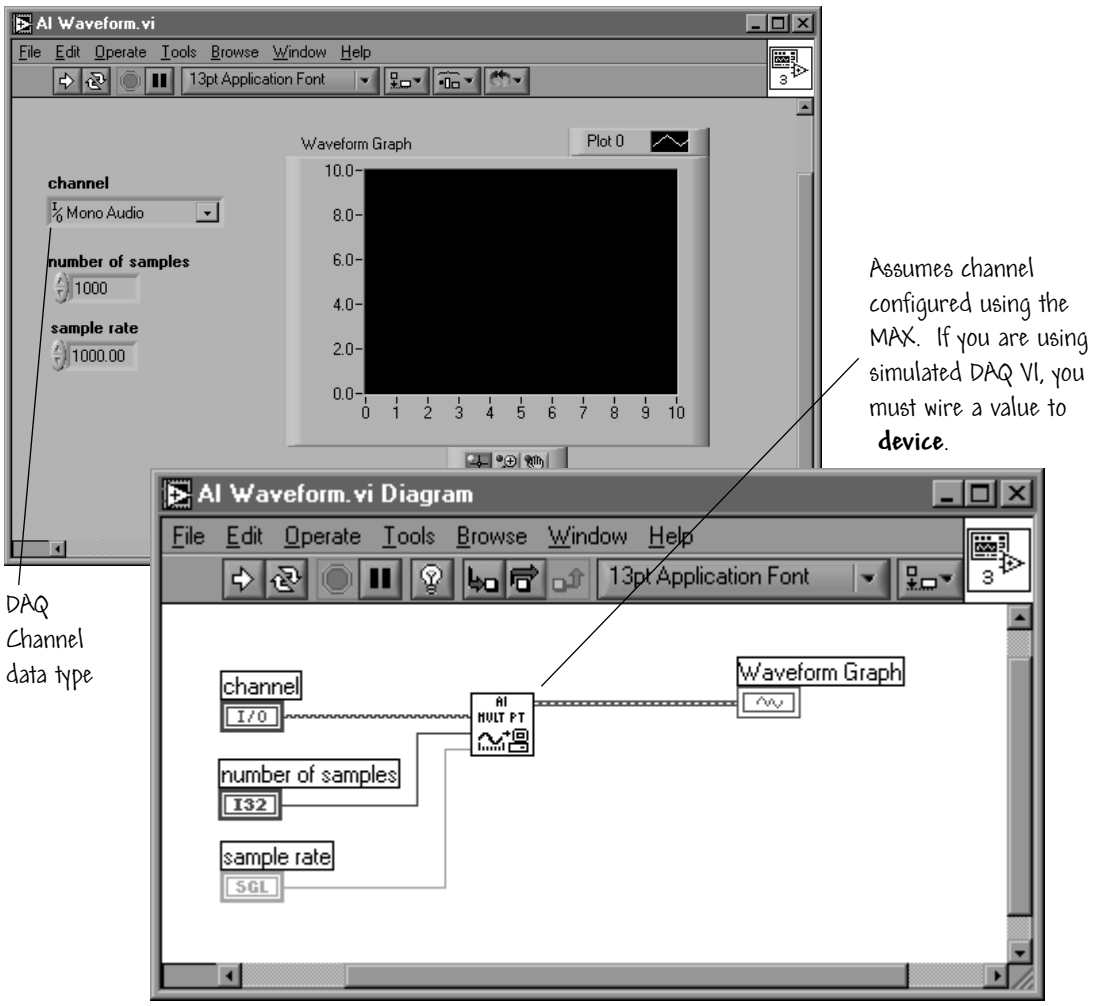


FIGURE 8.58
A VI designed to measure a waveform signal.

adding a new channel configuration, and remember that you are configuring an analog input.

In the screens that the MAX presents, fill in each item using Figure 8.60 as the guide. Some of the items that appear in MAX will already have default values that are acceptable. Otherwise, respond to the questions by entering the appropriate information. When you are finished entering the data, click on the **Finish** button located in the lower right corner of the fifth screen. The updated DAQ channel configuration should now include the **MonoAudio** on channel 1, as illustrated in Figure 8.61.

When you are finished, save the VI as AI Waveform.vi in Users Stuff in the Learning directory.

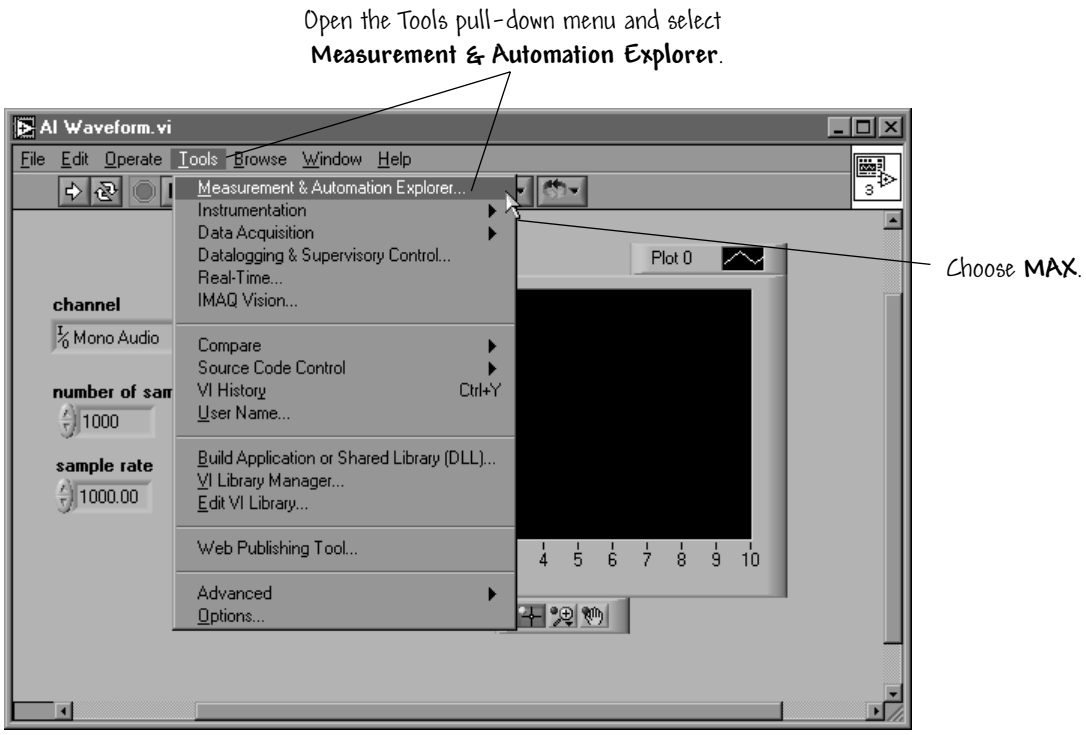


FIGURE 8.59
Opening MAX to configure the DAQ channel for acquiring waveform data.

Return to the front panel, enter the appropriate values for the controls, and run the VI. The graph plots the analog waveform. Try different values for the sampling rate and the number of samples.

If you are having problems with your VI, you can find a working version of the VI shown in Figure 8.58 in Chapter 8 *of the* Learning *directory. The VI is called* AI Waveform.vi*. An example of a VI that acquires multiple waveforms from two analog channels is called* AI Waveforms.vi *and can be found in the* Chapter 8 *folder in the* Learning *directory. If you use this VI, you will need to configure another analog input channel in the MAX.*

◆

8.9 ANALOG OUTPUT

There are two general methods for analog output—a single point at a time and many points at a time. Similiar to the situation with analog input, with analog output you can perform single updates on analog output channels (using no

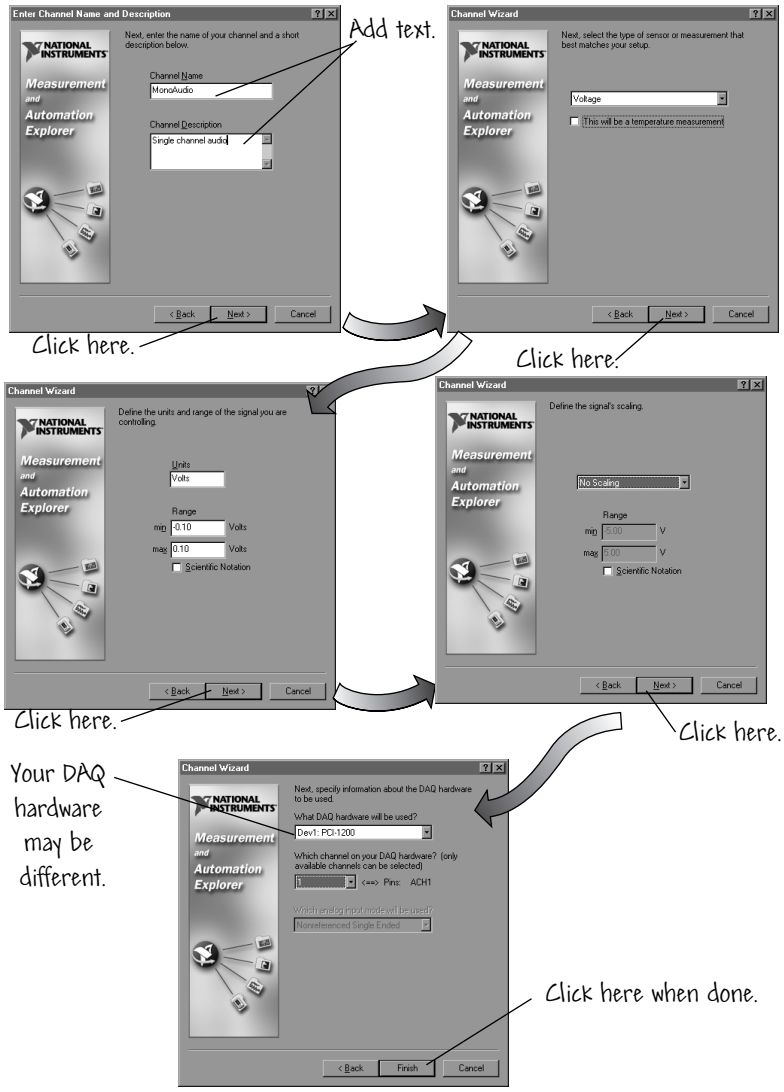


FIGURE 8.60
Configuring the MonoAudio analog channel using the MAX.

hardware clocks). You can update channels at a significantly faster and more precise rate using a buffer and an onboard counter to control how fast the updates occur.

In this chapter we discuss analog output and the role of VIs in the process. The VIs that control analog output operations are

- Easy Analog Output VIs: These VIs are appropriate for simple analog output applications, yet still possess built-in error-handling capability. The same limit settings are used for all output channels.

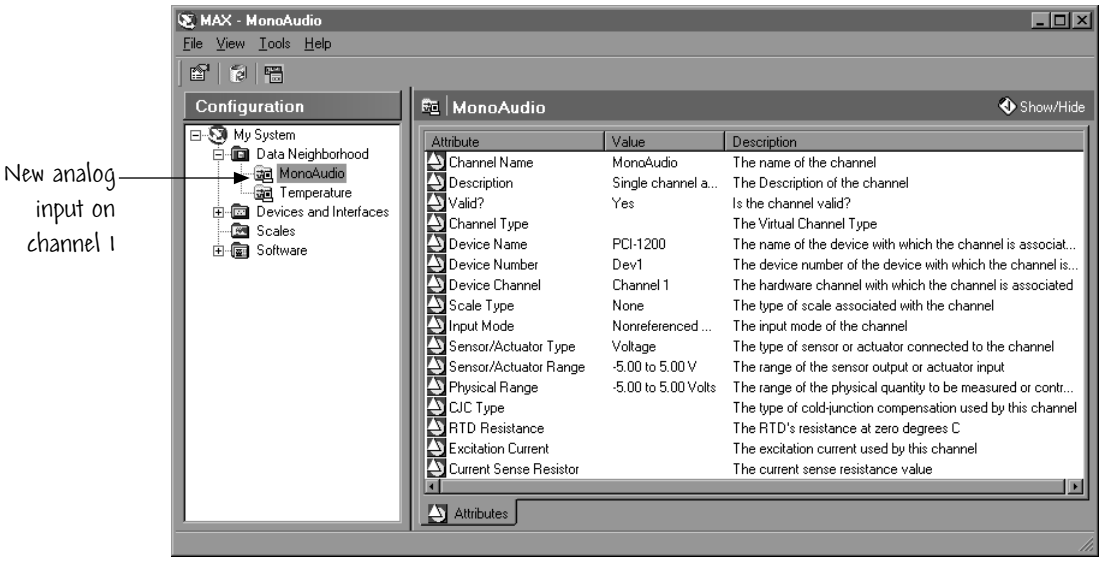


FIGURE 8.61
The MAX final configurations showing the **MonoAudio** input.

- Intermediate Analog Output VIs: These VIs are appropriate for most analog output applications that require special timing, user-defined error-handling routines, separate limit settings for each channel, or continuous data output.
- Analog Output Utility VIs: The Utility VIs are groupings of the intermediate VIs that perform single-point, waveform, and continuous analog output. These VIs combine the flexibility of user-defined error handling that come with using Intermediate VIs but with the convenience of calling just one VI per operation.
- Advanced Analog Output VIs: These are the building blocks for the other Analog Output VIs and are rarely needed since the Intermediate VIs provide most of the same functionality in a simpler format.

As is the case of analog input VIs, in *Learning with LabVIEW*, we consider only the use of the Easy Analog Output VIs that perform analog output operations. Much of the discussion that follows has the same flavor as the previous discussions on analog input—the process of configuring the channels and the functions of the associated VIs is similar. As before, the MAX is used to configure the channels, except that the configuration process begins by choosing **Analog Output**, as illustrated in Figure 8.62.

8.9.1 Single-Point Generation

The Analog Output library contains VIs that perform **digital-to-analog (D/A)** conversions. The AO Update Channel VI writes a specified value to an analog

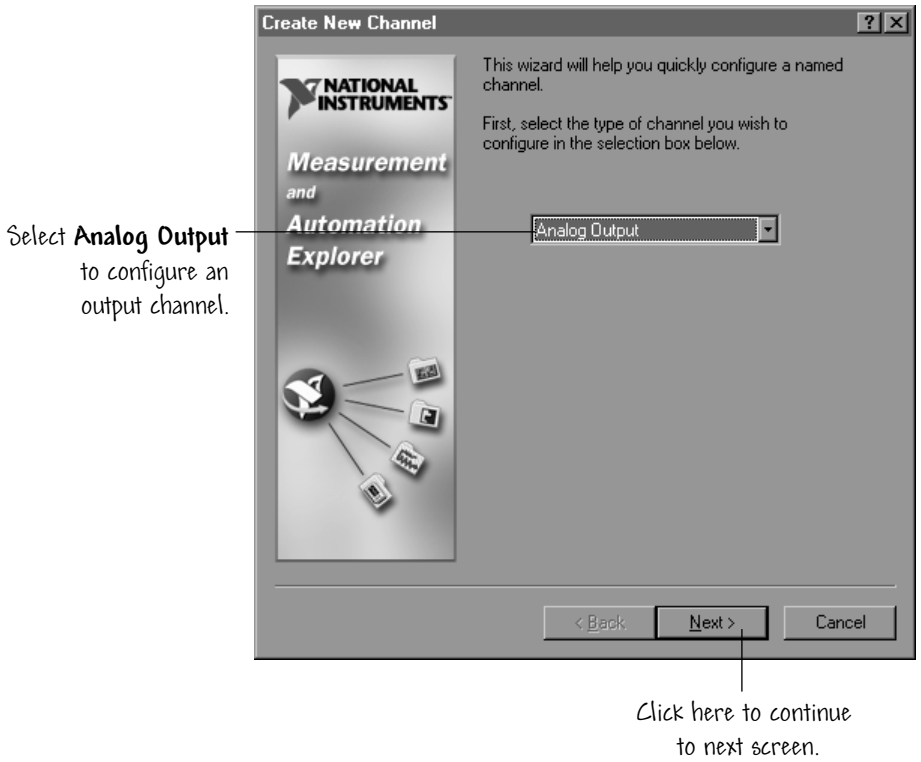


FIGURE 8.62
The MAX can configure the analog output channels—select **Analog Output** on the first MAX screen.

output channel. The VI has three inputs:

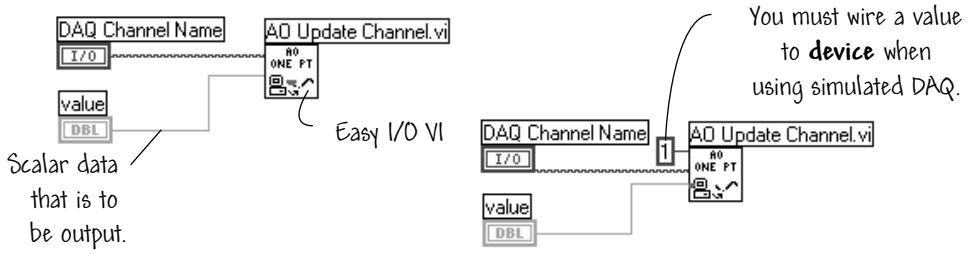
- **Device**: the device number of the DAQ board. You do not need a device value when you use channel names configured with the MAX. If you are using the simulation DAQ VIs, you must wire a value to **device**.
- **DAQ Channel Name**: specifies the analog output channel number.
- **Value**: the data to be output.

Figure 8.63 illustrates how to wire AO Update Channel.vi. If an error occurs during the operation of AO Update Channel, a dialog box displays the error code, and you have the option to abort the operation or continue execution.

Practice with Single-Point Generation

The objective of this exercise is to produce an analog output and to use the MAX to configure the system for a **Motor Speed** output.

Open a new VI and build the front panel shown in Figure 8.64. The input variable **channel** is a DAQ channel name control. The input **value** is determined by the user input on the front panel and is the data value that is output. Since Motor Speed is an analog output channel, after configuring the channel with the



(a) Don't wire the input **device** when using real VIs and the DAQ Channel Wizard

(b) Wire the input **device** when using simulated DAQ VIs

FIGURE 8.63
The AO Update Channel.vi writes a single value to an analog output channel.

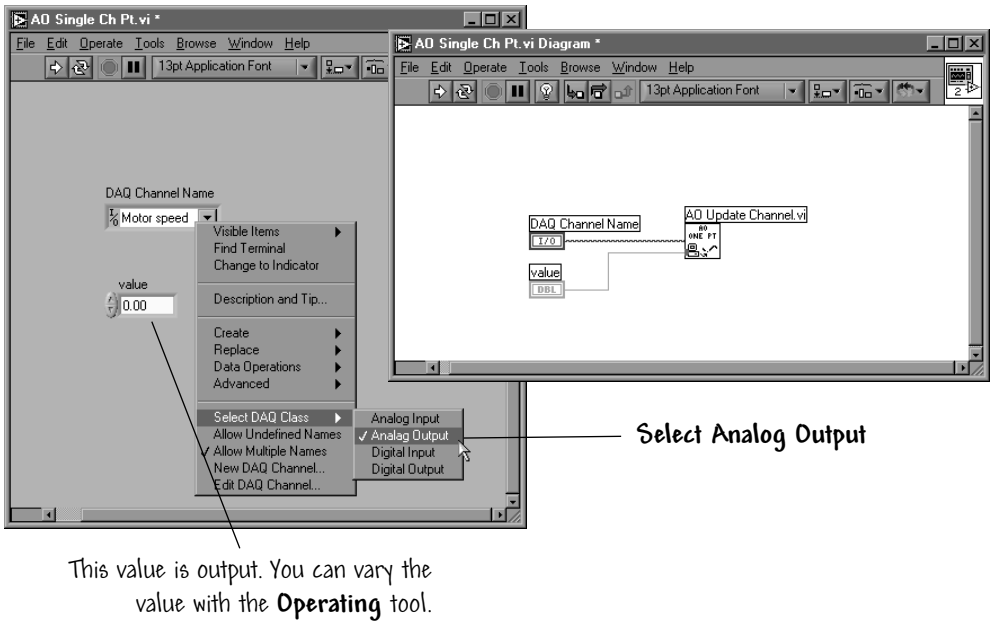


FIGURE 8.64
A VI designed to output a single point.

MAX, you will need to pop-up on the DAQ Channel Name control and choose **Select DAQ Class**. Then select **Analog Output** as shown in Figure 8.64. The result is that the analog output channel will be available for selection on the DAQ Channel Name menu. When the front panel is ready, switch to the block diagram and wire the VI AO Update Channel, using Figure 8.64 as a guide.

Open the MAX and follow essentially the same steps as with the analog input examples. The goal is to add to the channel configuration an analog output called

Motor Speed assigned to channel 0. Initiate the process of adding a new channel configuration, and remember that you are configuring an analog output channel. Right-click on **Data Neighborhood** in the MAX window and select **Create New...**. Then, in the next window select **Virtual Channel** and click on **Finish**.

At this point in the chapter, you should have configured two analog input channels **Temperature** and **MonoAudio**, and after this exercise is done, one analog output channel **Motor Speed**. In the screens that the MAX presents, fill in each item, using Figure 8.65 as the guide. Whe you are finished entering the

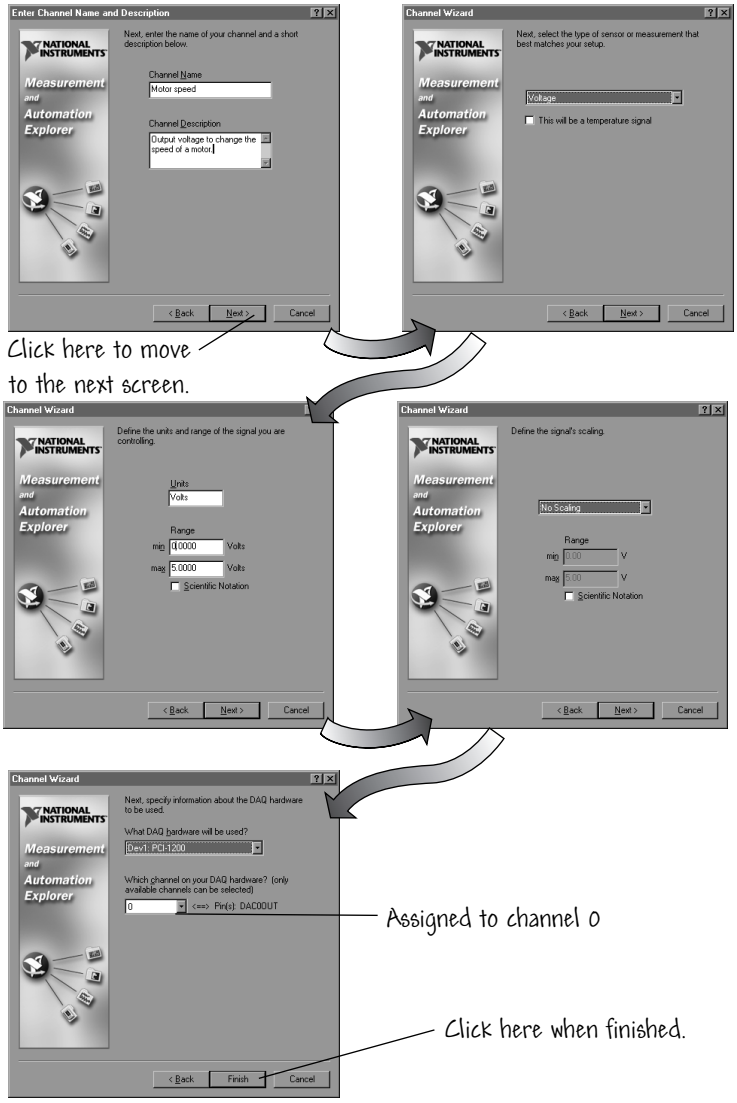


FIGURE 8.65
Configuring the analog output channel using the MAX.

data, click on **Finish**. You can access the **DAQ Channel Viewer** in Tools≫Data Acquisition≫DAQ Channel Viewer to see the current channel configuration, as shown in Figure 8.66.

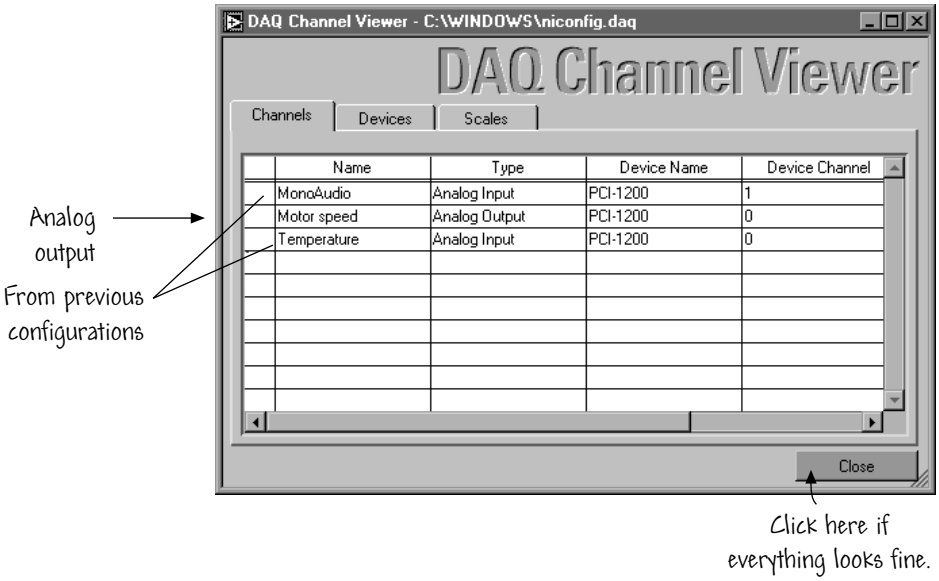


FIGURE 8.66
The MAX final configuration, including the **Motor Speed** output.

When you are finished constructing the VI, save it as AO Single Ch Pt.vi in the Users Stuff folder in the Learning directory. Modifying the VI to change from single-point output to a single channel to single-point outputs to multiple channels is an easy procedure. The AO Update Channels VI is used in place of the AO Update Channel VI, and of course, the number of assigned channels must be consistent with the desired number of output channels. Using the MAX makes the process of configuring the output channels a straightforward one.

If you are having problems with your VI, you can find a working version of the VI shown in Figure 8.64 in Chapter 8 of the Learning directory. The VI is called AO Single Ch Pt.vi. An example VI illustrating single-point outputs to multiple channels is AO Single Pt Mult Ch.vi located in the Chapter 8 folder.

◆

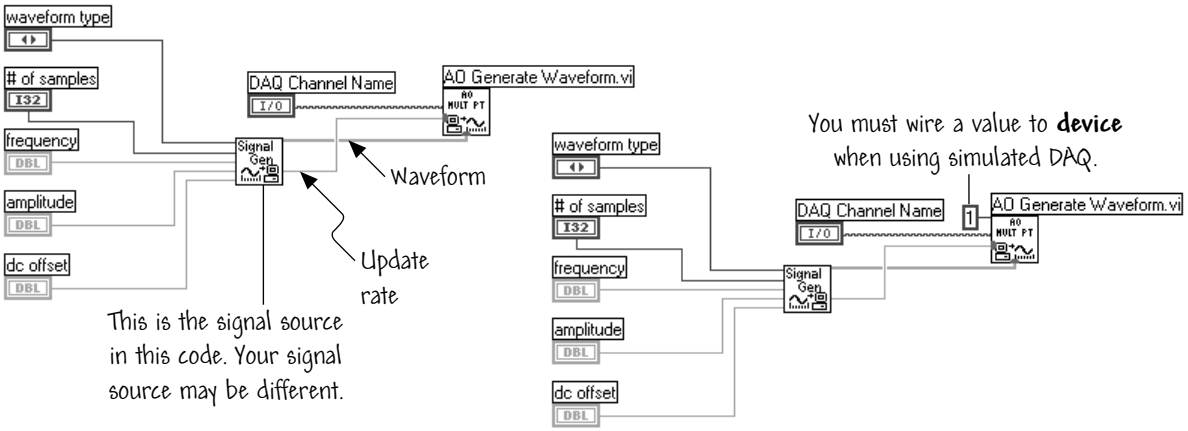
8.9.2 Waveform Generation

Sometimes when you generate output signals you are concerned with the rate of output. For example, you might want your DAQ system to act as a signal generator, and you want to control the rate of output and also the sample interval

between points. Generating one point at a time may not be fast enough, and it is difficult to obtain a constant sample interval between each point because the interval depends on a number of factors over which you have limited control. With the AO Generate Waveform VI, you can generate multiple points at rates greater than the AO Update Channel VI can achieve, and you specify the sampling rates. The AO Generate Waveform VI has four inputs:

- **Device**: the device number of the DAQ board. When you use channel names configured with the MAX, you do not need a device value. If you are using the simulation DAQ VIs, you must wire a value to **device**.
- **Channel**: specifies the analog output channel number.
- **Update rate**: the number of voltage updates to generate per second.
- **Waveform**: a 1D array that contains data to be written to the analog output channel.

Figure 8.67 illustrates how to wire AO Generate Waveform.vi.



(a) Don't wire the input **device** when using real DAQ VIs and the DAQ Channel Wizard

(b) Wire the input **device** when using simulated DAQ VIs

FIGURE 8.67
The AO Generate Waveform.vi writes multiple values to an analog output channel.

Practice with Waveform Generation

The objective of this example is to produce an analog waveform output using AO Generate Waveform.vi. You will need to again use the MAX to configure the channels.

Open a new VI and build the front panel shown in Figure 8.68. The input variable DAQ Channel Name specifies the output analog channel configured using the MAX. When the front panel is ready, switch to the block diagram and wire the AO Generate Waveform VI, using Figure 8.68 as a guide.

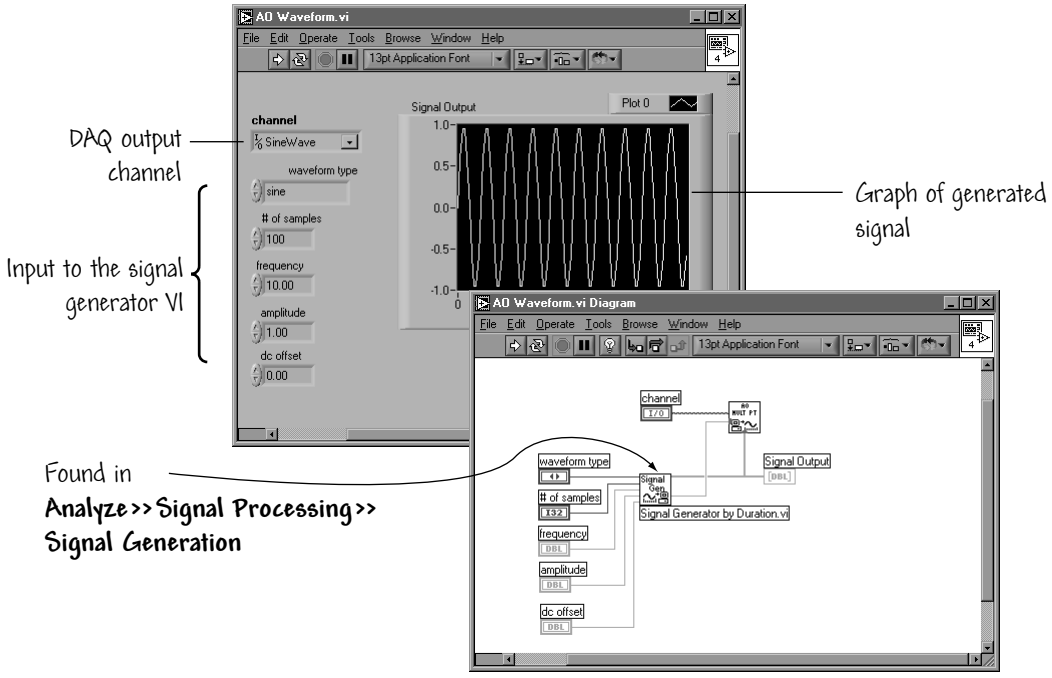


FIGURE 8.68
A VI designed to output multiple points—a single waveform.

To generate a waveform signal within the code, a VI called Signal Generation by Duration.vi is used. This VI can be found in the **Analyze** palette, as seen in Figure 8.68. Figure 8.69 illustrates the path to locating the signal-generation VI.

Open the MAX. The DAQ channel configuration needs to be modified to include the **Sine Wave** on channel 1, as illustrated in the DAQ Channel Viewer in Figure 8.70. To accomplish this task, you can follow the steps shown in Figure 8.71. Initiate the process of adding a new channel configuration, and remember to configure an analog output channel. Use Figure 8.71 as a guide to complete the channel configuration. When you are finished entering the data, click on the **Finish** button located in the lower right corner of the fifth screen.

When you are finished wiring the VI, save it as AO Waveform.vi in User's Stuff in the Learning directory.

If you are having problems with your VI, you can find a working version of the VI shown in Figure 8.68 in Chapter 8 of the Learning directory. The VI is called AO Waveform.vi.

A more advanced version of AO Waveform.vi would allow the DAQ system to output multiple-waveforms. A VI that performs multiple-waveform output is

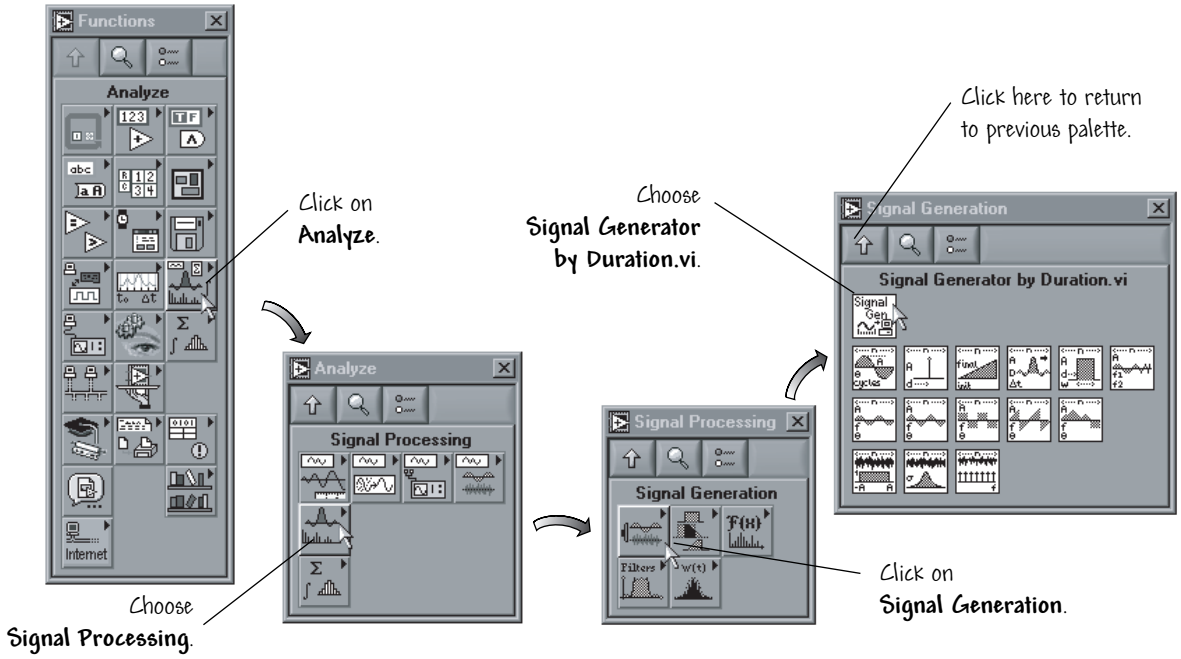


FIGURE 8.69
Using a VI to generate a sine wave signal (and other signals as well).

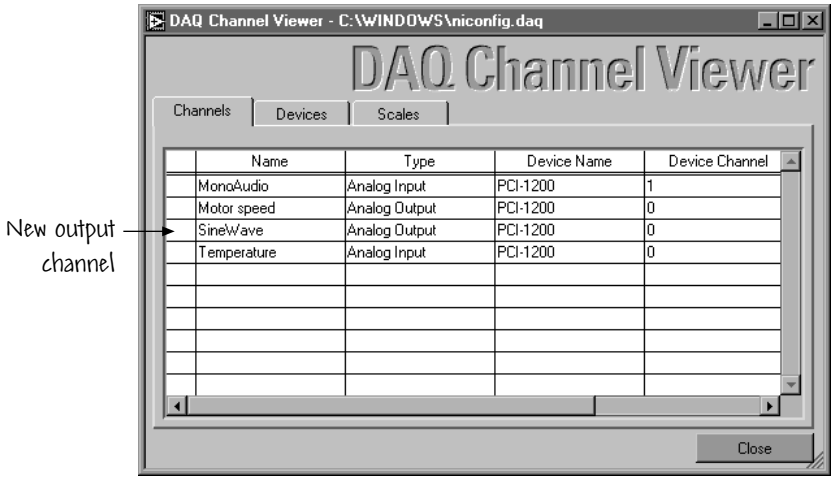


FIGURE 8.70
The **DAQ Channel Viewer** final configuration, including the **SineWave** output.

shown in Figure 8.72. In this situation, we use the signal-generation VI twice to generate multiple waveforms and the Build Array function to assemble the multi-dimensional array of input data. If you attempt to construct this VI or use the one provided in the Learning directory, you will need to configure the MAX for the

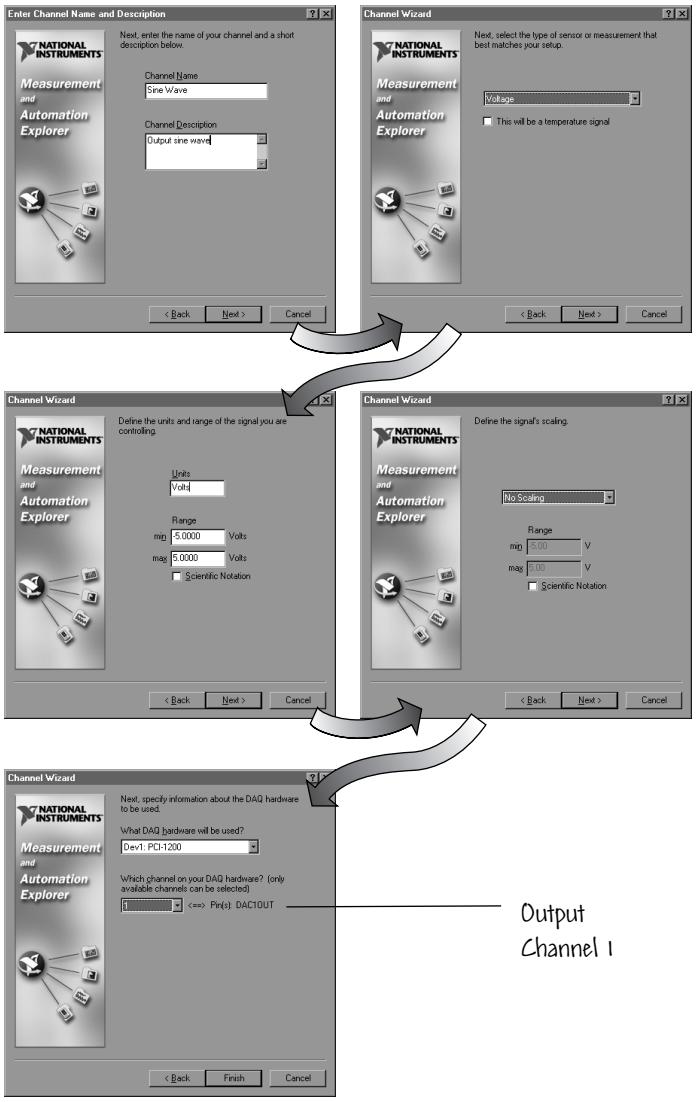


FIGURE 8.71
Configuring the analog output channel using the MAX.

additional output. A version of the VI shown in Figure 8.72 is called AO Waveforms.vi and is located in Chapter 8 of the Learning directory. ◆

8.10 DIGITAL I/O

Digital I/O components on DAQ devices consist of parts that generate or accept binary on/off signals that are often used to control processes, generate patterns for testing, and communicate with peripheral equipment. Digital lines are grouped

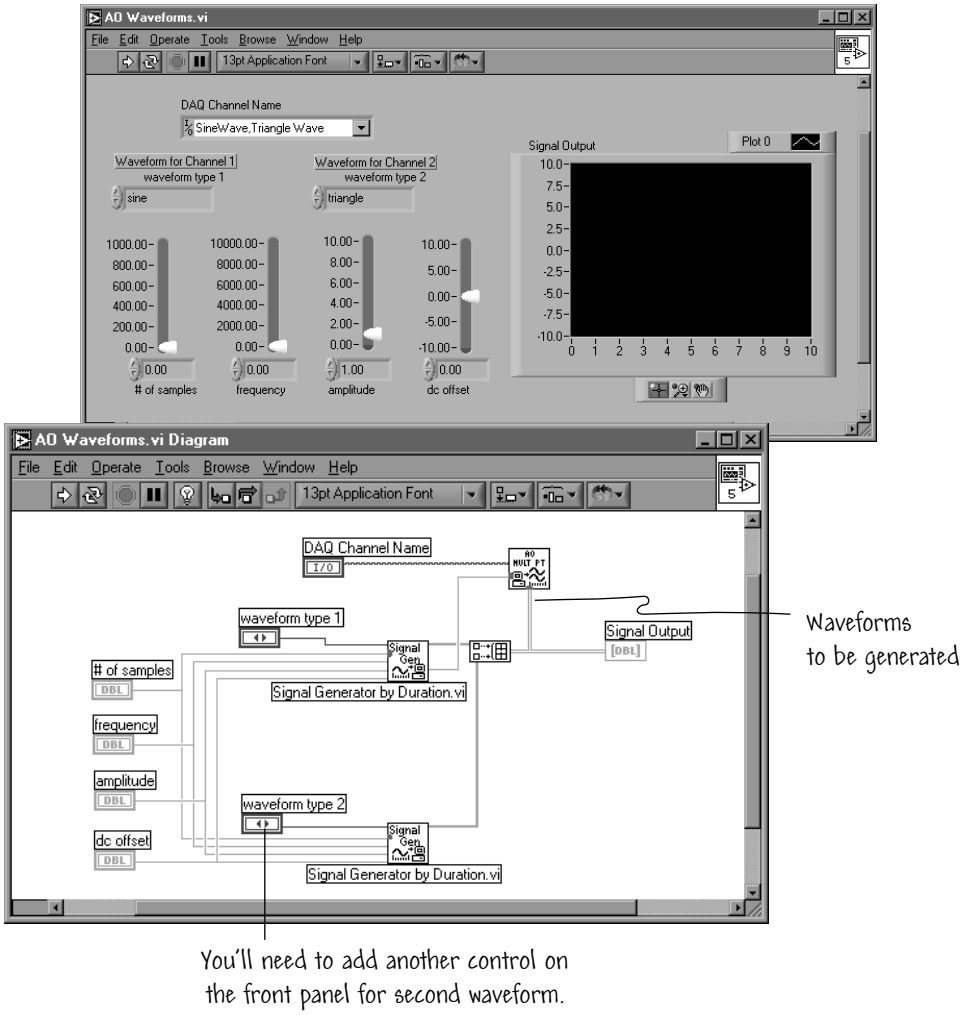


FIGURE 8.72
A VI designed to output multiple waveforms.

into ports generally consisting of four or eight lines per port, as depicted in Figure 8.73. Usually all lines within the same port must all be either input lines or output lines. Since a port contains multiple digital lines, by writing to or reading from a port you can set or retrieve simultaneously the states of multiple lines.

There are two types of digital DAQ acquisition: **immediate** (or nonlatched) and **handshaked** (or latched). With immediate digital I/O, the DAQ system updates the digital lines immediately. This is the only type of digital I/O considered in this chapter. Handshaked digital I/O occurs when a device accepts or transfers data after a digital pulse (that is, the handshake) has been received. There are two

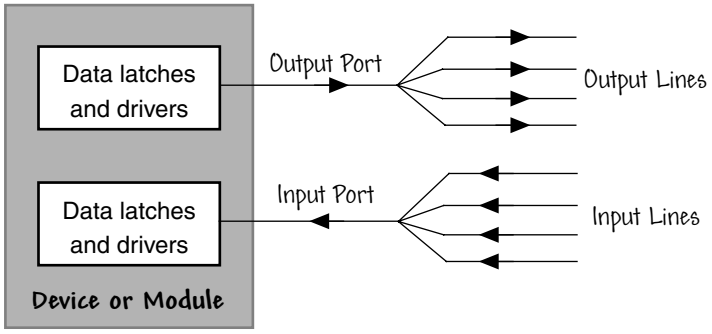


FIGURE 8.73
Digital I/O components on DAQ devices consist of hardware parts that generate or accept binary on/off signals.

types of handshaked digital I/O: **nonbuffered** and **buffered**. Not all devices and modules support latched (handshaked) digital I/O—refer to the documentation for your specific hardware for more information.

In this chapter we focus on the issues surrounding the transfer of digital data across a single port. The most common way to use digital lines is with immediate digital I/O, and all DAQ devices with digital components support this mode. When your VI calls a function in immediate digital I/O mode, the digital line or port states are immediately updated or the current digital value of an input line is returned. You can use the Easy I/O VIs for immediate digital I/O.

LabVIEW inputs or outputs only one value on each digital line in the immediate digital I/O mode. You can completely configure the port (and sometimes the line) direction in software, and you can switch directions repeatedly in a program. A typical example of when you might use immediate digital I/O is in controlling or monitoring relays. You can also use multiple ports or groups of ports to perform digital I/O functions; however, in order to group digital ports, you must use Intermediate or Advanced VIs.

Figure 8.74 shows the Easy I/O VIs and how their various inputs and outputs are wired. The four Easy I/O VIs can read data from or write data to a single digital line or to an entire port immediately.

*Unless you wire a positive value to the **iteration** input on the digital I/O VIs, all of these functions will automatically configure the DAQ board for the appropriate operation each time you call the VI. If you are using the Easy I/O VIs in a loop, consider wiring the **iteration** input to the loop iteratation terminal—this will avoid unneccesary port reconfiguration on every iteration.*

Practice with Digital I/O

The objective of this exercise is to practice with digital input and output using the four LabVIEW Easy I/O VIs. Before proceeding to constructing your VIs you

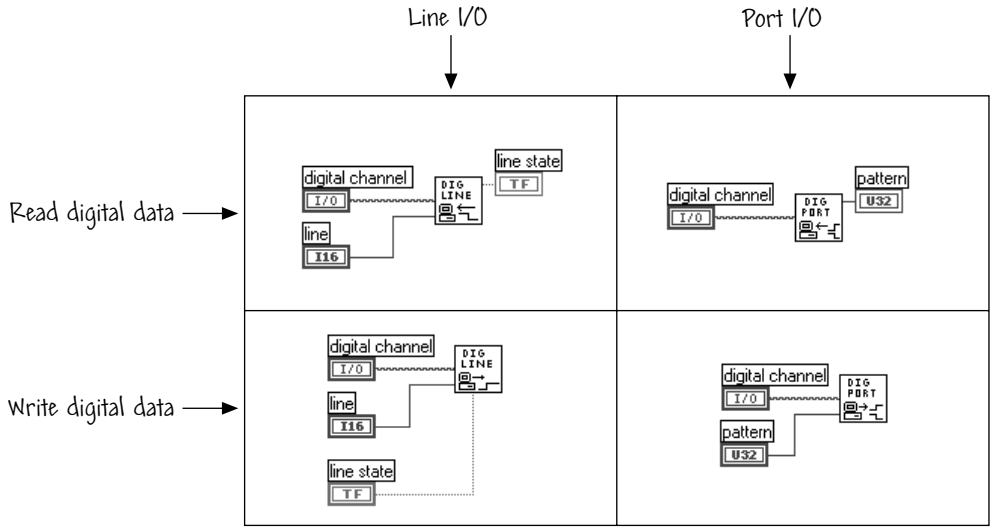


FIGURE 8.74
Wiring the digital Easy I/O VIs.

will need to use the MAX to configure the channels for digital I/O to both a line and to a port.

If you have configured channels using the MAX in the previous examples in this chapter, you are familiar with the process—the process for configuring channels for digital I/O is similar to the process for analog signals. Refer to the various sections in the earlier parts of this chapter on using the MAX if you need to refresh your memory. When you are ready to begin configuring the system, open the MAX, right-click on **Data Neighborhood** and select **Create New...**. As before, at the next screen select **Virtual Channel** and then press **Finish**. When the window **Create New Channel** appears, select **Digital I/O** from the ring. Click **Next** to continue the configuration process.

The digital channel name may refer to either a port or a line in a port. You do not need to specify device, line, or port width, since these inputs are not used by LabVIEW if a channel name is specified in the digital channel input.

Follow the steps shown in Figure 8.75 to configure the **Relay** channel as a **Write to Line** digital channel type. When that is complete, continue the configuration process by following the steps shown in Figure 8.76 to configure the **Relay Group** channel as **Write to Port** digital channel type.

In some applications, you may need to invert the digital values, that is, change a signal from on to off, and vice versa. This is a method of scaling the digital signals. With the MAX you can specify inversion on a per-line basis, so that the scaling of individual lines in a port can have different inversion.

When you are finished configuring the channels for digital output, exit the MAX and construct the two VIs:

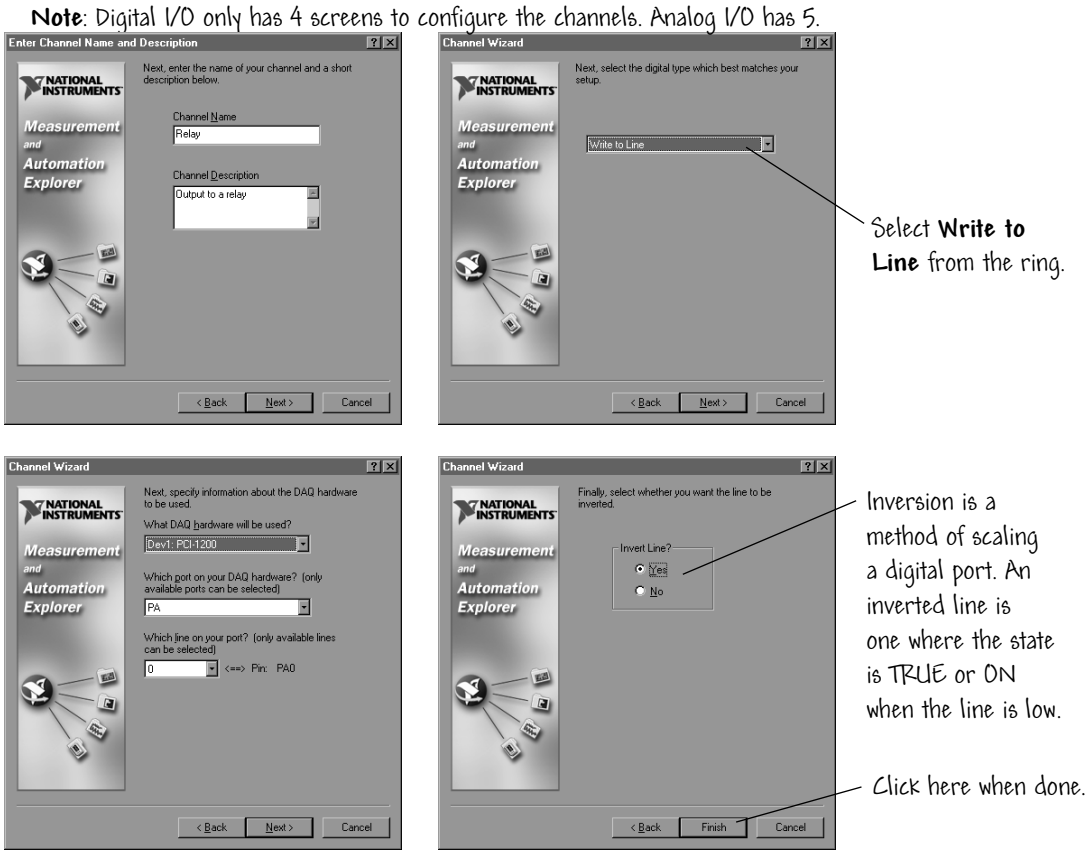


FIGURE 8.75
Configuring the DAQ system to write to a digital line.

- DIO Write 1 Line.vi: Sets a line on a port to a logical high or low state.
- DIO Write Port.vi: Outputs a digital pattern to the specified port.

One new object used in the VI is the Boolean Array to Number function shown on the block diagram in Figure 8.77. This function is located on the **Boolean** subpalette of the **Functions** palette. Use the online help to read about this function. These two VIs write to a line and port, respectively. When you are finished constructing the VIs, save them as DIO Write 1 Line.vi and DIO Write Port.vi in Users Stuff in the Learning directory.

If you are having problems with your VI, you can find working versions of the VIs shown in Figure 8.77 in Chapter 8 of the Learning directory. The VIs are called DIO Write 1 Line.vi and DIO Write Port.vi.

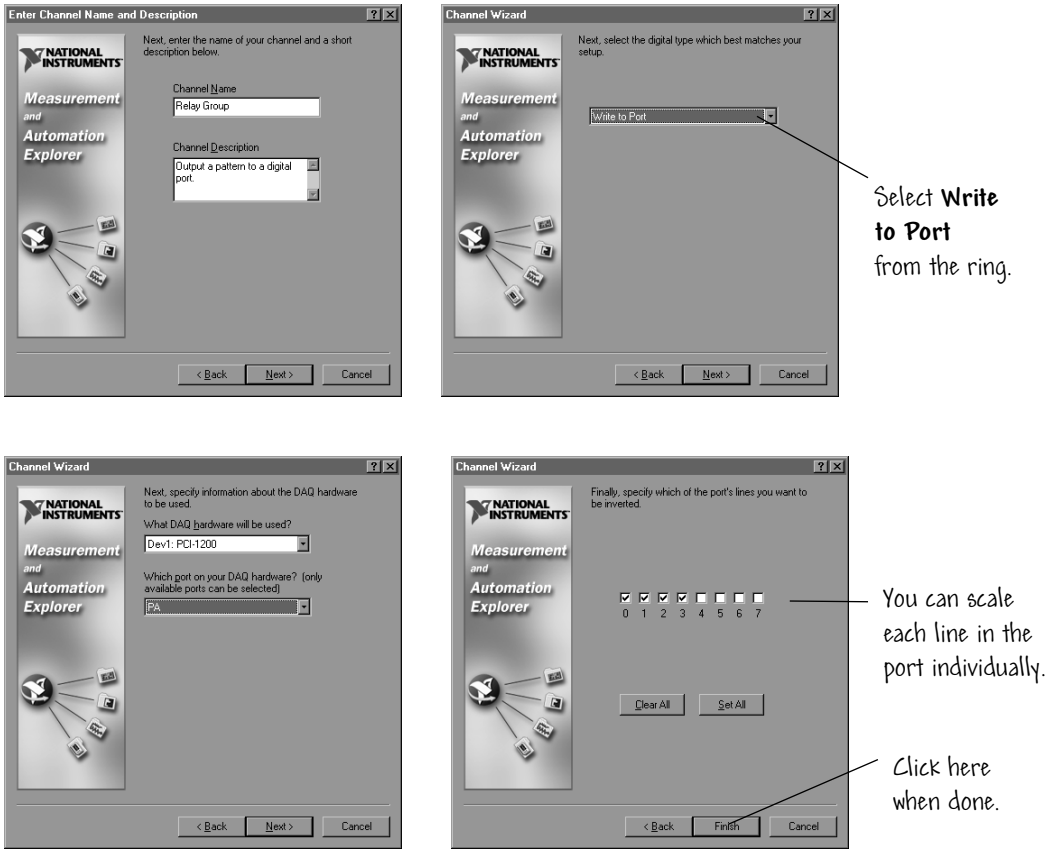


FIGURE 8.76
Configuring the DAQ system to write to a digital port.

A new object used in the VI is the Number to Boolean Array function shown in the block diagram in Figure 8.78. This function can be found on the **Boolean** subpalette of the **Functions** palette. Use the online help to read about this function.

If you are interested in practicing with digital read from a line or port, use the MAX to configure the system for digital read from a line and from a port. Then build the VIs shown in Figures 8.78.

- DIO Read 1 Line.vi: Read the state of a digital line.
- DIO Read Port.vi: Reads the state of all lines in a port.

If you are having problems with your VIs, you can find working versions of the VIs shown in Figure 8.78 in Chapter 8 *of the* Learning *directory. The VIs are called* DIO Read 1 Line.vi *and* DIO Read Port.vi. ◆

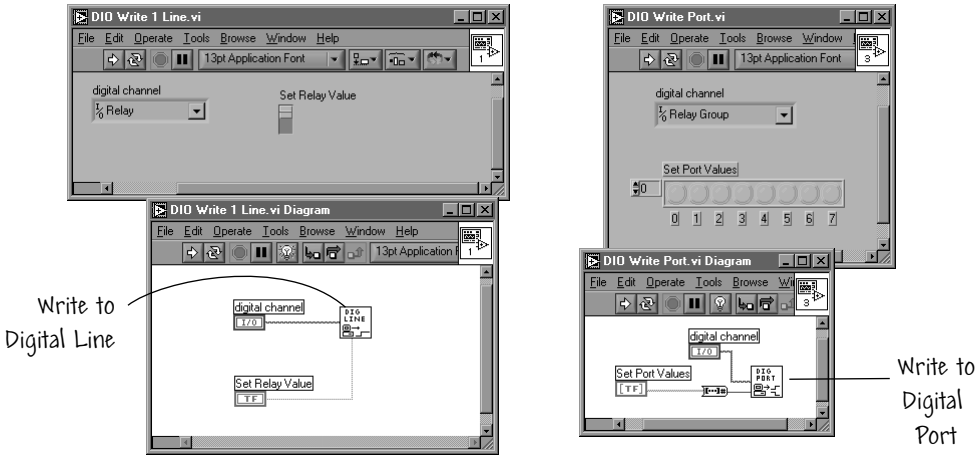


FIGURE 8.77
Two VIs to write to a line and port, respectively.

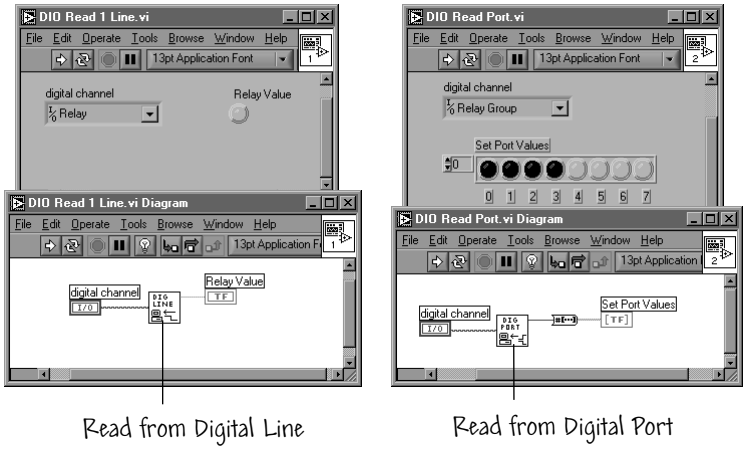
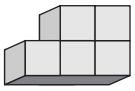


FIGURE 8.78
Two VIs to read from a line and port, respectively.

BUILDING BLOCK

8.11 BUILDING BLOCKS: DIGITAL ALARMS



We depart now from the building block exercise based on the volume-measuring system used in Chapters 3–7 and turn to an exercise using data acquisition. The goal is to develop a VI to read from an analog input channel and to write to a digital channel. You should use the MAX to configure the channels.

A real world example where you can use this VI is in measuring the temperature of a room. If the temperature rises above a certain value, you turn on the air conditioning.

If you have simulated VIs, you must wire a control or constant to the device input on the AI Sample Channels.vi *and* Write to Digital Port.vi.

You can use the front panel and block diagram of the VI illustrated in Figure 8.79 as a guide for your own work.

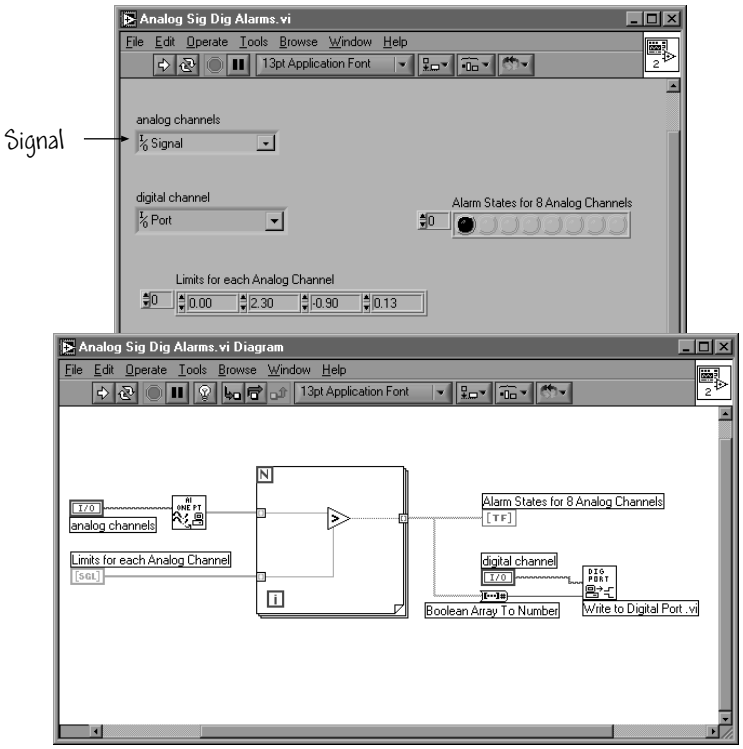


FIGURE 8.79
The front panel and block diagram for the Analog Sig Dig Alarms.vi.

The VI depicted in Figure 8.79 currently has only one analog channel for input and one digital channel for output. You can change that by listing the channel names separated by a comma in **analog channels**. You will have to configure the channels appropriately with the MAX. The analog input should be named **Signal** and the digital output named **Port**. The two inputs **analog channels** and **digital channel** are DAQ Channel name controls.

Before running the VI, you will need to set values for the limits for each analog channel. Run the VI to verify that you are acquiring data properly and writing out to the digital port. You might consider placing probes at strategic locations on the block diagram during the debugging phase. For example, placing a probe prior to the Boolean Array to Number function and one after the same function will help you to visualize the operation of the function. When you are finished with your VI, close it and save it as Analog Sig Dig Alarms.vi in the Users Stuff folder in the Learning directory.

8.12 RELAXING READING: USING DAQ IN STUDENT LABORATORIES

Electrical engineering students at the University of Pennsylvania use virtual instrumentation in a variety of laboratory classes. The Undergraduate Electrical Engineering Laboratory is the primary teaching laboratory and the place where most students are exposed to instrumentation for the first time. In a typical laboratory assignment the students design a virtual instrument that reads and writes analog data, plots the data on an XY graph and also writes the data to a spreadsheet file. An example of the VI used in the circuits and systems subject area is shown in Figure 8.80.

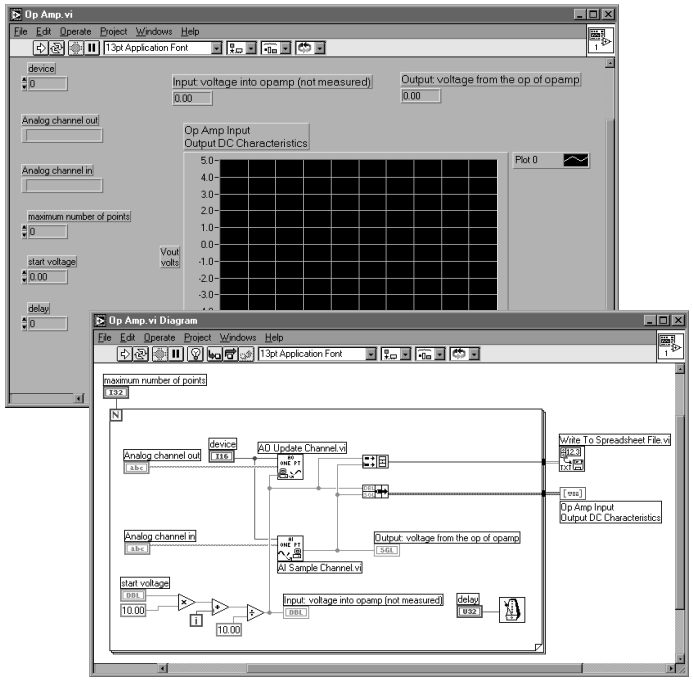


FIGURE 8.80
The front panel and block diagram for the Op Amp.vi

The VI shown in Figure 8.80 is located in the Chapter 8 *folder in the* Learning *directory. It is called* Op Amp.vi. *If you attempt to use the VI remember that you must configure the channels properly (using the Channel Wizard).*

The stated goals of incorporating LabVIEW in the laboratory at the University of Pennsylvania are to introduce students to

- Computer-based data processing concepts
- Laboratory automation concepts
- The idea that software is a powerful tool in instrumentation

The emphasis is on *building* VIs to satisfy the laboratory objectives, rather than using existing VIs.

Future plans include the introduction of the Internet so that students can explore remote data acquisition, and a possible next step would be to integrate *Hi Q* into the curriculum as a way to assist in report writing (more on *Hi Q* in Chapter 12). Some excellent resources for learning more about how virtual instrumentation can be used successfully in the classroom are the following websites:

http://www.ee.upenn.edu/rcal
and
http://www.ee.upenn.edu/rca/software/labview.html

The upenn websites have links to other interesting LabVIEW-oriented websites, and you might consider surfing around to see how other colleges and universities are utilizing virtual instruments.

For more information, contact:

Professor Siddharth Deliwala
Dept. of Electrical Engineering
Room 329, Moore School of Electrical Engineering
University of Pennsylvania
200 S 33rd Street
Philadelphia, PA 19104-6390
e-mail: deliwala@ee.upenn.edu

8.13 SUMMARY

DAQ systems consist of the following elements:

- Signals
- Transducers
- Signal-conditioning hardware

- DAQ board or module
- Application software

There are two types of signals:

- Analog—provides level, shape, or frequency content information
- Digital—provides state or rate information

With many transducers it is necessary to provide for signal conditioning. Some types of signal conditioning are

- Amplification
- Transducer excitation
- Linearization
- Isolation
- Filtering

There are two types of signal sources:

- Grounded sources—devices that plug into the building ground
- Floating sources—isolated from the building ground system

There are three types of measurement systems:

- Differential—use this type whenever possible!
- Referenced single-ended—use single-ended types if you require more channels
- Nonreferenced single-ended

Multifunction DAQ boards typically include:

- Analog-to-digital convertors (ADCs)
- Digital-to-analog convertors (DACs)
- Digital I/O ports
- Counter/timer circuits

When configuring the DAQ board, you should consider how the following parameters will affect the quality of the digitized signal:

- Resolution: Increasing resolution increases the precision of the ADC.
- Range: Decreasing range increases precision.
- Limit settings: Changing limit settings to reflect the signal range increases precision.

The Measurement & Automation Explorer is a utility that helps configure the channels on the DAQ board according to the sensors to which they are connected.

The LabVIEW DAQ VIs are organized into palettes corresponding to the type of operation involved—analog input, analog output, counter operations, or

digital I/O. Under this palette, the DAQ VIs are organized into six palettes (we covered the first three topics in this chapter):

- **Analog Input**
- **Analog Output**
- **Digital I/O**
- **Counter**
- **Calibration and Configuration**
- **Signal Conditioning**

Each palette contains VIs or palettes of VIs organized as Easy I/O, Intermediate, Utility, and Advanced VIs. The Easy I/O VIs consist of high-level VIs that perform basic analog input, analog output, and digital I/O. We discussed the Easy I/O VIs only—they are ideal for simple analog and digital I/O or for getting started with DAQ in LabVIEW.

KEY TERMS

Analog-to-digital converter (ADC): An electronic device (often an integrated circuit) that converts an analog voltage to a digital number.

ADC resolution: The resolution of the ADC measured in bits. An ADC with 16 bits has a higher resolution (and thus a higher degree of accuracy) than a 12-bit ADC.

Bipolar: A signal range that includes both positive and negative values (e.g., −5 V to 5 V).

Channel: Pin or wire lead where analog or digital signals enter or leave a data acquisition device.

Channel name: A unique name given to a channel configuration in the Measurement & Automation Explorer.

Code width: The smallest detectable change in an input voltage of a DAQ device.

Digital-to-analog convertor (DAC): An electronic device (often an integrated circuit) that converts a digital number to an analog voltage or current.

DAQ Channel Wizard: Utility that guides you through naming and configuring your DAQ analog and digital channels.

DAQ Solution Wizard: Utility that generates solutions for DAQ applications.

Data acquisition (DAQ): Process of acquiring data from plug-in devices.

Differential measurement system: A method of configuring your device to read signals in which you do not connect inputs to a fixed reference (such as the earth or a building ground).

Floating signal sources: Signal sources with voltage signals that are not connected to an absolute reference or system ground. *Also* called nonreferenced signal sources.

Grounded signal sources: Signal sources with voltage signals that are referenced to system ground, such as the earth or building ground. *Also* called referenced signal sources.

Handshaked digital I/O: A type of digital I/O where a device accepts or transfers data after a signal pulse has been received. *Also* called latched digital I/O.

Immediate digital I/O: A type of digital I/O where the digital line or port is updated immediately or returns the digital value of an input line. *Also* called nonlatched digital I/O.

Input/output (I/O): The transfer of data to or from a computer system involving communication channels and DAQ interfaces.

Limit settings: The settings that you specify as the maximum and minimum voltages on analog input signals.

Linearization: A type of signal conditioning in which the voltage levels from transducers are linearized, so that the voltages can be scaled to measure physical phenomena.

LSB: Least significant bit.

Measurement & Automation Explorer: Provides access to all National Instruments DAQ, GPIB, IMAQ, IVI, Motion, VISA, and VXI devices.

Non-referenced single-ended (NRSE) measurement system: All measurements are made with respect to a common reference, but the voltage at this reference may vary with respect to the measurement system ground.

Range: The minimum and maximum analog signal levels that the analog-to-digital convertor can digitize.

Referenced single-ended (RSE) measurement system: All measurements are made with respect to a common reference or ground. *Also* called a grounded measurement system.

Sampling rate: The rate at which the DAQ board samples an incoming signal.

SCXI: Signal Conditioning eXtensions for Instrumentation. The National Instruments product line for conditional low-level signals within an external chassis near the sensors so that only high-level signals in a noisy environment are sent to the DAQ board.

Signal conditioning: The manipulation of signals to prepare them for digitizing.

Unipolar: A signal range that is always positive (e.g., 0 V to 10 V).

Update rate: The rate at which voltage values are generated per second.

EXERCISES

E8.1 Assume you are sampling a transducer that varies between 80 mV and 120 mV. Your board has a voltage range of 0 to +10 V, ±10 V, and ±5 V. What limit setting would you select for maximum precision if you use a DAQ board with 12-bit resolution? Use the limit setting values in Table 8.3 for your calculations.

E8.2 From the LabVIEW startup screen, click on the **Search Examples** button. Click on the following links: I/O Interfaces≫Data Acquisition (DAQ)≫Software-

Created Instruments≫One Channel Simple Oscilloscope. This VI is a good example of how virtual instruments emulate actual instruments. Run this VI.

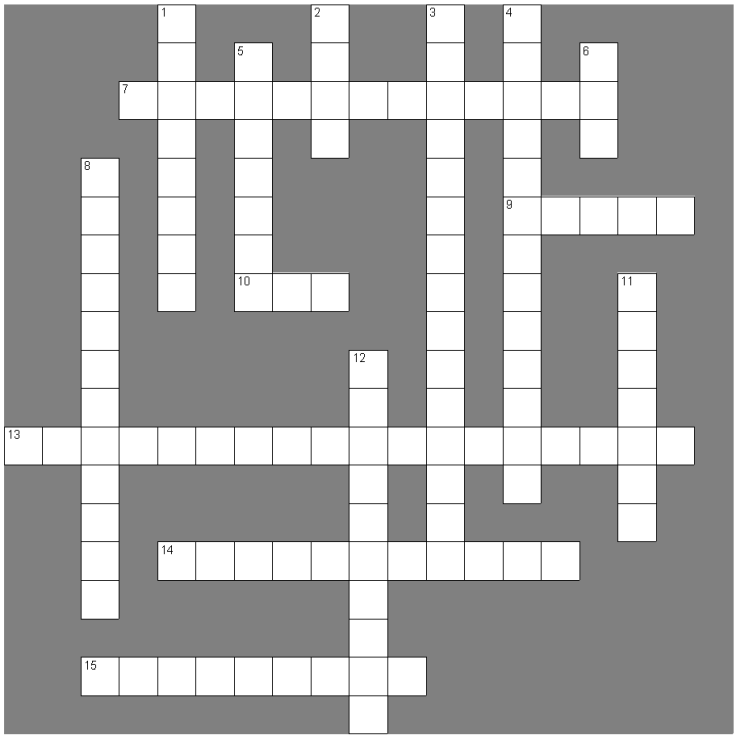
E8.3 If you have **Search Examples** open to Software-Created Instruments, click on the Benchtop Function Generator link. To open the **Search Examples**, refer to the previous exercise. This VI is another good example of how virtual instruments emulate actual instruments. Run this VI.

PROBLEMS

P8.1 You are a production engineer at a plant and you want to continuously measure the flow of fluid through pipes. The sensor that measures flow produces an analog signal that you can acquire. Create a VI that continuously acquires analog data until you press a stop button on the front panel. As you acquire data, display it on a chart. Hint: You will need to use the intermediate analog input VIs.

P8.2 You are acquiring analog data that measures the volume of a tank. You only want to take continual measurements when the machinery is on. Turning on the machinery can act as a trigger, specifically a digital trigger, to start measuring data. Write a VI that acquires analog data only after a digital trigger has occurred.

P8.3 Complete the crossword puzzle.



Across

7. The settings specifed as max and min voltages on analog input signals.
9. The min and max analog signal levels that the a-to-d convertor can digitize.
10. All measurements are made with respect to a common reference or ground.
13. The manipulation of signals to prepare them for digitizing.
14. A unique name given to a channel configuration in the MAX.
15. The smallest detectable change in an input voltage of a DAQ device.

Down

1. A signal range that is always positive.
2. All measurements are made with respect to a common reference, but the voltage may vary with respect to the ground.
3. Process of acquiring data from plug-in devices.
4. A type of signal conditioning in which the voltage levels from transducers are linearized.
5. A signal range that includes both positive and negative values.
6. Least significant bit.
8. The rate at which the DAQ board samples an incoming signal.
11. Pin or wire lead where signals enter or leave a DAQ device.
12. The rate at which voltage are generated.