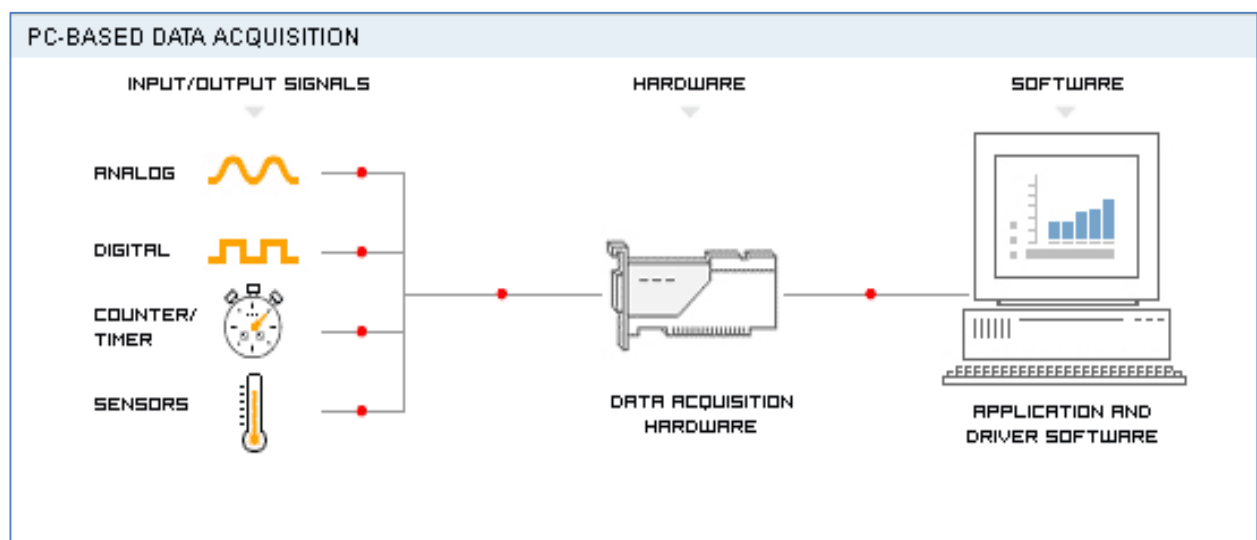


Data Acquisition in LabVIEW

Hans-Petter Halvorsen, 2016.10.28



Preface

This tutorial explains the basic concepts of a Data Acquisition in LabVIEW.

You find this document and additional resources online at:

<http://home.hit.no/~hansha/?tutorial=daq>

You should have some basic knowledge about LabVIEW, e.g., the “Introduction to LabVIEW” tutorial. This document is available for download at <http://home.hit.no/~hansha/>.

For more information about LabVIEW, visit: <http://home.hit.no/~hansha/?page=labview>.

In addition to LabVIEW, you will need to install the driver **NI-DAQmx**.

[Figure on title page: www.ni.com]

Table of Contents

Preface	2
Table of Contents	iii
1 Introduction to LabVIEW	1
1.1 Dataflow programming	1
1.2 Graphical programming.....	1
1.3 Benefits.....	2
2 Introduction to Data Acquisition.....	3
2.1 Introduction to DAQ - Data Acquisition.....	3
2.1.1 Physical input/output signals.....	4
2.1.2 DAQ device/hardware	4
2.1.3 Driver software	5
2.1.4 Your software application (Application software)	6
2.2 MAX – Measurement and Automation Explorer	6
2.3 NI-DAQmx.....	7
2.3.1 DAQ Assistant	8
2.4 NI USB-6008.....	8
3 Physical input/output signals	10
3.1 Transducers	10
3.2 Signals.....	11
3.2.1 Analog Signals	11
3.3 Digital Signals.....	13
4 MAX.....	15

5	NI-DAQmx	17
5.1.1	DAQ Assistant	18
5.2	Simulating a DAQ Device	18
6	DAQ Devices	20
6.1	Performing Analog-to-Digital Conversion.....	21
6.2	Performing Digital-to-Analog Conversion.....	21
6.3	Using Counters	21
6.4	Using Digital I/O.....	22
7	NI USB-6008	23
7.1	Connect NI USB-6008 to the PC.....	23
7.1.1	Testing the USB-6008 in MAX.....	24
7.2	Using NI USB-6008 in LabVIEW.....	30
7.3	DAQ Assistant	31
7.3.1	Analog Input	31
7.3.2	Analog Output.....	38
8	Logging to File	43
8.1	Writing to Measurement File	43
8.2	Read from Measurement File.....	46

1 Introduction to LabVIEW

LabVIEW (short for **L**aboratory **V**irtual **I**nstrumentation **E**ngineering **W**orkbench) is a platform and development environment for a visual programming language from National Instruments. The graphical language is named "G". Originally released for the Apple Macintosh in 1986, LabVIEW is commonly used for data acquisition, instrument control, and industrial automation on a variety of platforms including Microsoft Windows, various flavors of UNIX, Linux, and Mac OS X. Visit National Instruments at www.ni.com.

The code files have the extension ".vi", which is an abbreviation for "Virtual Instrument". LabVIEW offers lots of additional Add-Ons and Toolkits.

1.1 Dataflow programming

The programming language used in LabVIEW, also referred to as G, is a dataflow programming language. Execution is determined by the structure of a graphical block diagram (the LV-source code) on which the programmer connects different function-nodes by drawing wires. These wires propagate variables and any node can execute as soon as all its input data become available. Since this might be the case for multiple nodes simultaneously, G is inherently capable of parallel execution. Multi-processing and multi-threading hardware is automatically exploited by the built-in scheduler, which multiplexes multiple OS threads over the nodes ready for execution.

1.2 Graphical programming

LabVIEW ties the creation of user interfaces (called front panels) into the development cycle. LabVIEW programs/subroutines are called virtual instruments (VIs). Each VI has three components: a block diagram, a front panel, and a connector panel. The last is used to represent the VI in the block diagrams of other, calling VIs. Controls and indicators on the front panel allow an operator to input data into or extract data from a running virtual instrument. However, the front panel can also serve as a programmatic interface. Thus a virtual instrument can either be run as a program, with the front panel serving as a user interface, or, when dropped as a node onto the block diagram, the front panel defines the inputs and outputs for the given node through the connector pane. This implies each VI can be easily tested before being embedded as a subroutine into a larger program.

The graphical approach also allows non-programmers to build programs simply by dragging and dropping virtual representations of lab equipment with which they are already familiar. The LabVIEW programming environment, with the included examples and the documentation, makes it simple to create small applications. This is a benefit on one side, but there is also a certain danger of underestimating the expertise needed for good quality "G" programming. For complex algorithms or large-scale code, it is important that the programmer possess an extensive knowledge of the special LabVIEW syntax and the topology of its memory management. The most advanced LabVIEW development systems offer the possibility of building stand-alone applications. Furthermore, it is possible to create distributed applications, which communicate by a client/server scheme, and are therefore easier to implement due to the inherently parallel nature of G-code.

1.3 Benefits

One benefit of LabVIEW over other development environments is the extensive support for accessing instrumentation hardware. Drivers and abstraction layers for many different types of instruments and buses are included or are available for inclusion. These present themselves as graphical nodes. The abstraction layers offer standard software interfaces to communicate with hardware devices. The provided driver interfaces save program development time. The sales pitch of National Instruments is, therefore, that even people with limited coding experience can write programs and deploy test solutions in a reduced time frame when compared to more conventional or competing systems. A new hardware driver topology (DAQmxBase), which consists mainly of G-coded components with only a few register calls through NI Measurement Hardware DDK (Driver Development Kit) functions, provides platform independent hardware access to numerous data acquisition and instrumentation devices. The DAQmxBase driver is available for LabVIEW on Windows, Mac OS X and Linux platforms.

2 Introduction to Data Acquisition

This chapter explains the basic concepts of using DAQ in LabVIEW.

Topics:

- Introduction to DAQ - Data Acquisition
- MAX – Measurement and Automation Explorer
- NI-DAQmx

LabVIEW is very powerful when it comes to creating DAQ applications. LabVIEW includes a set of VIs that let you configure, acquire data from, and send data to DAQ devices. Often, one device can perform a variety of functions, such as analog-to-digital (A/D) conversion, digital-to-analog (D/A) conversion, digital I/O, and counter/timer operations. Each device supports different DAQ and signal generation speeds. Also, each DAQ device is designed for specific hardware, platforms and operating systems.

National Instruments, the inventor of LabVIEW, also make DAQ devices, so the integration with the DAQ devices from NI and the LabVIEW software is seamless and makes it easy to do I/O operations from the LabVIEW environment.

2.1 Introduction to DAQ - Data Acquisition

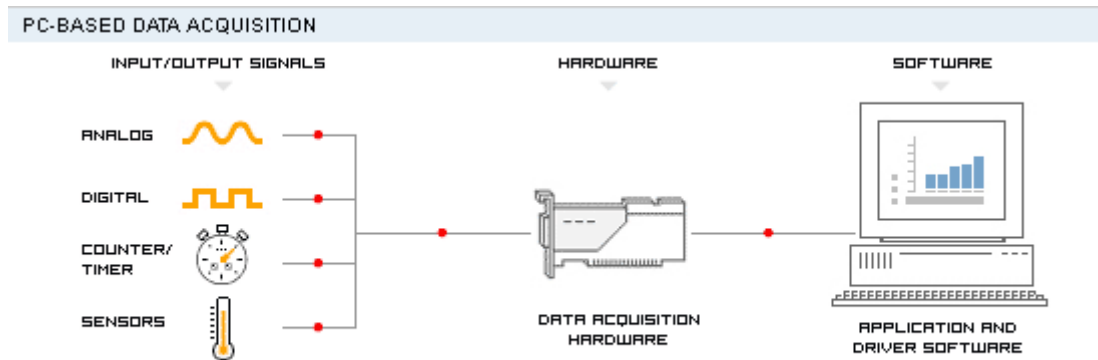
The purpose of data acquisition is to measure an electrical or physical phenomenon such as voltage, current, temperature, pressure, or sound. PC-based data acquisition uses a combination of modular hardware, application software, and a computer to take measurements. While each data acquisition system is defined by its application requirements, every system shares a common goal of acquiring, analyzing, and presenting information. Data acquisition systems incorporate signals, sensors, actuators, signal conditioning, data acquisition devices, and application software.

So summing up, Data Acquisition is the process of:

- Acquiring signals from real-world phenomena
- Digitizing the signals

- Analyzing, presenting and saving the data

The DAQ system has the following parts involved, see Figure:



The parts are:

- Physical input/output signals
- DAQ device/hardware
- Driver software
- Your software application (Application software)

For an Introduction to Data Acquisition, see this webcast:

<http://zone.ni.com/wv/app/doc/p/id/wv-169>

2.1.1 Physical input/output signals

A physical input/output signal is typically a voltage or current signal.

A voltage signal can typically be a 0-5V signal, while a current signal can typically be a 4-20mA signal.

2.1.2 DAQ device/hardware

DAQ hardware acts as the interface between the computer and the outside world. It primarily functions as a device that digitizes incoming analog signals so that the computer can interpret them.

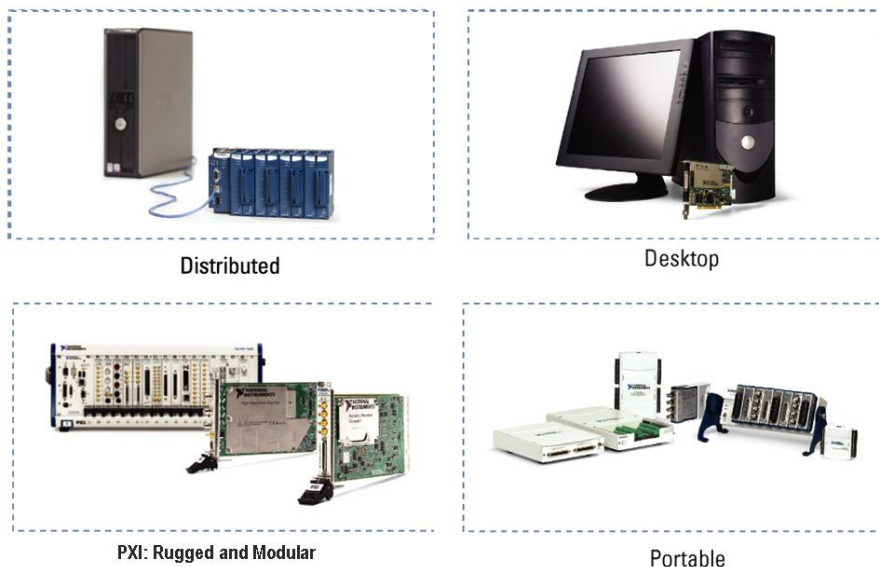
A DAQ device (Data Acquisition Hardware) usually has these functions:

- Analog input
- Analog output

- Digital I/O
- Counter/timers

We have different DAQ devices, such as:

- “**Desktop** DAQ devices” where you need to plug a PCI DAQ board into your computer. The software is running on a computer.
- “**Portable** DAQ devices” for connection to the USB port, Wi-Fi connections, etc. The software is running on a computer
- “**Distributed** DAQ devices” where the software is developed on your computer and then later downloaded to the distributed DAQ device.



[www.ni.com]

2.1.3 Driver software

Driver software is the layer of software for easily communicating with the hardware. It forms the middle layer between the application software and the hardware. Driver software also prevents a programmer from having to do register-level programming or complicated commands in order to access the hardware functions.

Driver software from National Instruments:

- NI-DAQmx

- NI-DAQmx Base

The **DAQ Assistant**, included with NI-DAQmx, is a graphical, interactive guide for configuring, testing, and acquiring measurement data. With a single click, you can even generate code based on your configuration, making it easier and faster to develop complex operations. Because DAQ Assistant is completely menu-driven, you will make fewer programming errors and drastically decrease the time from setting up your DAQ system to taking your first measurement.

NI-DAQmx Base offers a subset of NI-DAQmx functionality on Windows and Linux, Mac OS X, Windows Mobile and Windows CE.

2.1.4 Your software application (Application software)

Application software adds analysis and presentation capabilities to the driver software. Your software application normally does such tasks as:

- Real-time monitoring
- Data analysis
- Data logging
- Control algorithms
- Human machine interface (HMI)

In order to create your DAQ application you need a programming development tool, such as LabVIEW.

2.2 MAX – Measurement and Automation Explorer

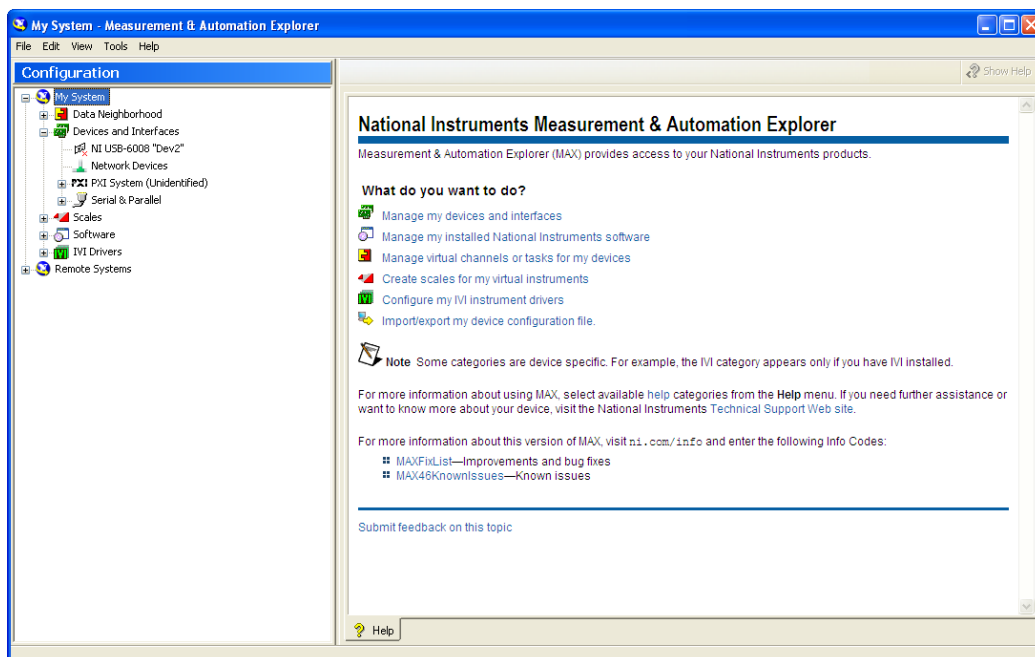
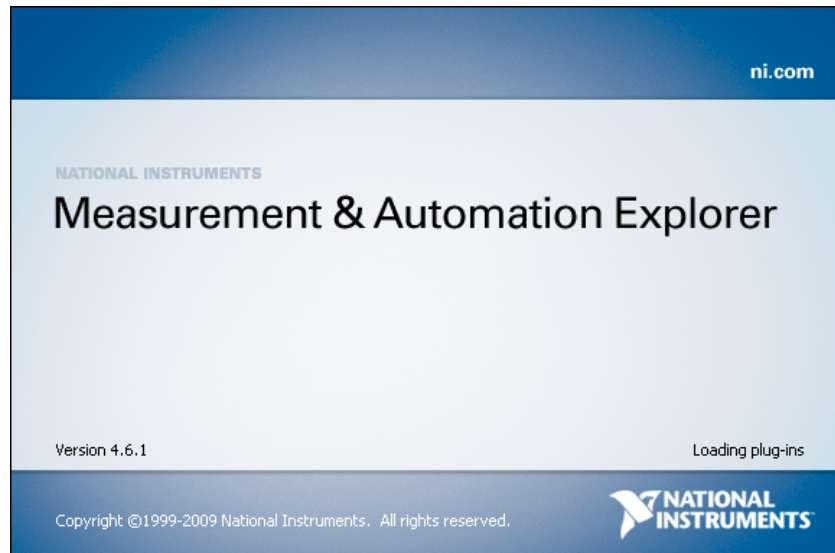
Measurement & Automation Explorer (MAX) provides access to your National Instruments devices and systems.

With MAX, you can:

- Configure your National Instruments hardware and software
- Create and edit channels, tasks, interfaces, scales, and virtual instruments
- Execute system diagnostics
- View devices and instruments connected to your system

- Update your National Instruments software

In addition to the standard tools, MAX can expose item-specific tools you can use to configure, diagnose, or test your system, depending on which NI products you install. As you navigate through MAX, the contents of the application menu and toolbar change to reflect these new tools.

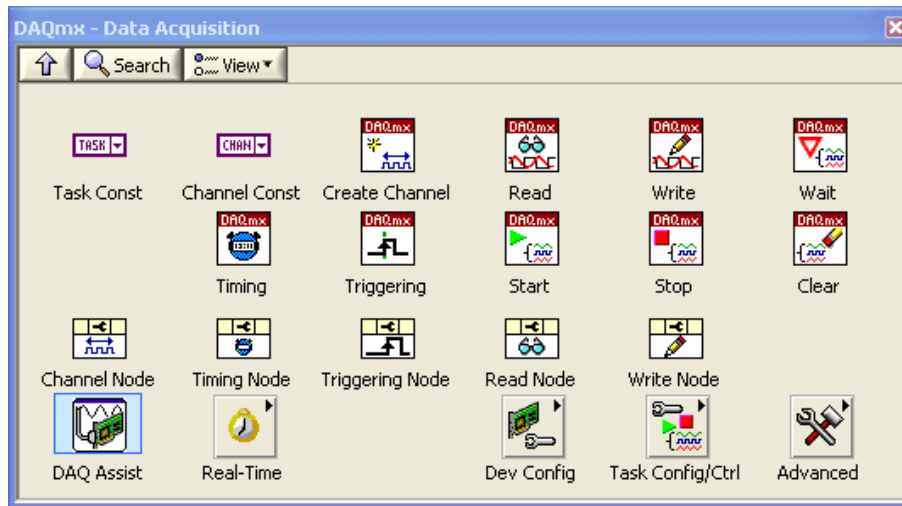


2.3 NI-DAQmx

The NI-DAQmx Driver software is the layer of software for easily communicating with the hardware. It forms the middle layer between the application software and the hardware.

Driver software also prevents a programmer from having to do register-level programming or complicated commands in order to access the hardware functions.

The DAQmx palette in LabVIEW:



The “DAQ Assistant” is an easy way to start using the DAQ features in LabVIEW. We will learn more about the “DAQ Assistant” in a later chapter.

2.3.1 DAQ Assistant

The **DAQ Assistant**, included with NI-DAQmx, is a graphical, interactive guide for configuring, testing, and acquiring measurement data. With a single click, you can even generate code based on your configuration, making it easier and faster to develop complex operations. Because DAQ Assistant is completely menu-driven, you will make fewer programming errors and drastically decrease the time from setting up your DAQ system to taking your first measurement.

2.4 NI USB-6008

NI USB-6008 is a simple and low-cost multifunction I/O device from National Instruments.



The device has the following specifications:

- 8 analog inputs (12-bit, 10 kS/s)
- 2 analog outputs (12-bit, 150 S/s)
- 12 digital I/O
- USB connection, No extra power-supply needed
- Compatible with LabVIEW, LabWindows/CVI, and Measurement Studio for Visual Studio .NET
- NI-DAQmx driver software

The NI USB-6008 is well suited for education purposes due to its small size and easy USB connection.

3 Physical input/output signals

Data acquisition involves gathering signals from measurement sources and digitizing the signal for storage, analysis, and presentation on a PC. Data acquisition (DAQ) systems come in many different PC technology forms for great flexibility when choosing your system. Scientists and engineers can choose from PCI, PXI, PCI Express, PXI Express, PCMCIA, USB, Wireless and Ethernet data acquisition for test, measurement, and automation applications. There are five components to be considered when building a basic DAQ system

- Transducers and sensors
- Signals
- Signal conditioning
- DAQ hardware
- Driver and application software

In this chapter we focus on Transducers, sensors and Signals.

3.1 Transducers

Data acquisition begins with the physical phenomenon to be measured. This physical phenomenon could be the temperature of a room, the intensity of a light source, the pressure inside a chamber, the force applied to an object, or many other things. An effective DAQ system can measure all of these different phenomena.

A transducer is a device that converts a physical phenomenon into a measurable electrical signal, such as voltage or current. The ability of a DAQ system to measure different phenomena depends on the transducers to convert the physical phenomena into signals measurable by the DAQ hardware. Transducers are synonymous with sensors in DAQ systems. There are specific transducers for many different applications, such as measuring temperature, pressure, or fluid flow. Below we see some common phenomena and the transducers used to measure them.

Phenomenon	Transducer
------------	------------

Temperature	Thermocouple, RTD, Thermistor
Light	Photo Sensor
Sound	Microphone
Force and Pressure	Strain Gage, Piezoelectric Transducer
Position and Displacement	Potentiometer, LVDT, Optical Encoder
Acceleration	Accelerometer
pH	pH Electrode

Different transducers have different requirements for converting phenomena into a measurable signal. Some transducers may require excitation in the form of voltage or current. Other transducers may require additional components and even resistive networks to produce a signal.

Refer to [ni.com/sensors](https://www.ni.com/sensors) for more information on transducers.

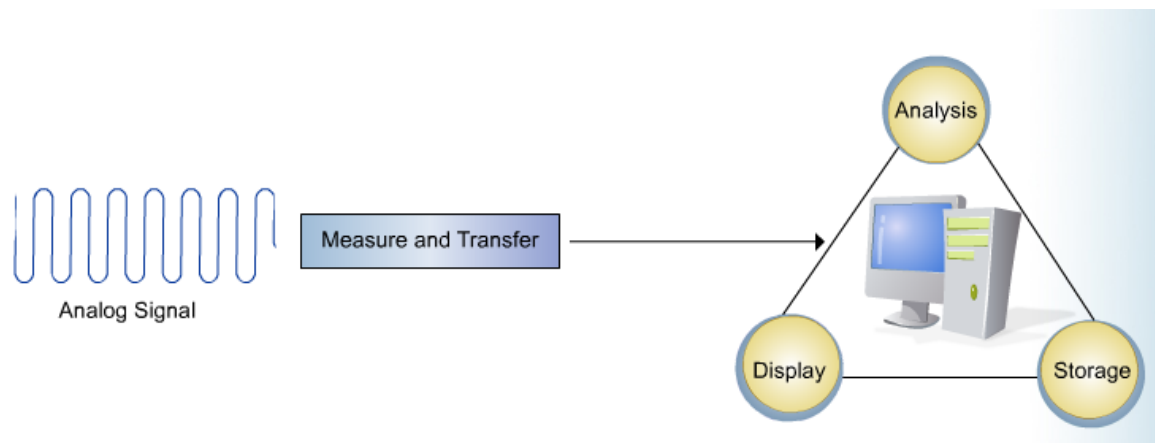
3.2 Signals

The appropriate transducers convert physical phenomena into measurable signals. However, different signals need to be measured in different ways. For this reason, it is important to understand the different types of signals and their corresponding attributes. Signals can be categorized into two groups:

- Analog
- Digital

3.2.1 Analog Signals

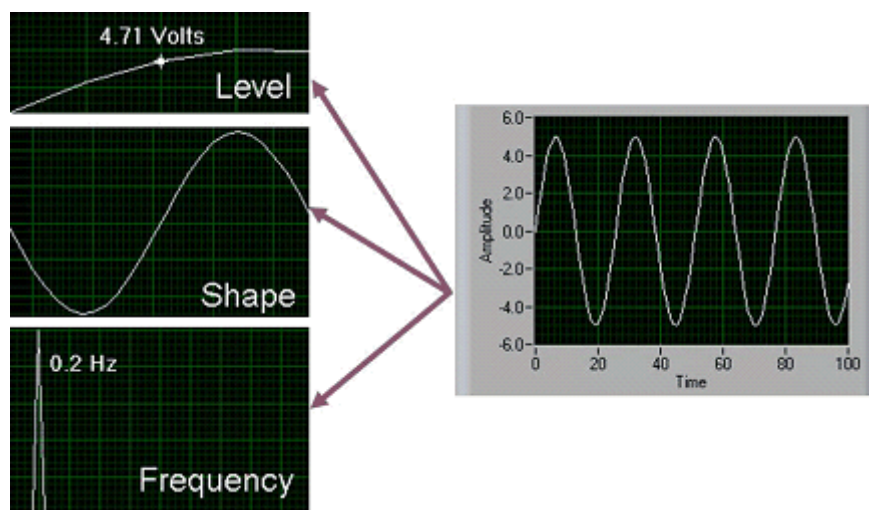
Analog input is the process of measuring an analog signal and transferring the measurement to a computer for analysis, display, or storage. An analog signal is a signal that varies continuously. Analog input is most commonly used to measure voltage or current. You can use many types of devices to perform analog input, such as multifunction DAQ (MIO) devices, high-speed digitizers, digital multimeters, and Dynamic Signal Acquisition (DSA) devices.



[www.ni.com]

An analog signal can be at any value with respect to time. A few examples of analog signals include voltage, temperature, pressure, sound, and load. The three primary characteristics of an analog signal is:

- Level
- Shape
- Frequency



Level

Because analog signals can take on any value, the level gives vital information about the measured analog signal. The intensity of a light source, the temperature in a room, and the pressure inside a chamber are all examples that demonstrate the importance of the level of a signal. When measuring the level of a signal, the signal generally does not change quickly

with respect to time. The accuracy of the measurement, however, is very important. A DAQ system that yields maximum accuracy should be chosen to aid in analog level measurements.

Shape

Some signals are named after their specific shape - sine, square, sawtooth, and triangle. The shape of an analog signal can be as important as the level, because by measuring the shape of an analog signal, you can further analyze the signal, including peak values, DC values, and slope. Signals where shape is of interest generally change rapidly with respect to time, but system accuracy is still important. The analysis of heartbeats, video signals, sounds, vibrations, and circuit responses are some applications involving shape measurements.

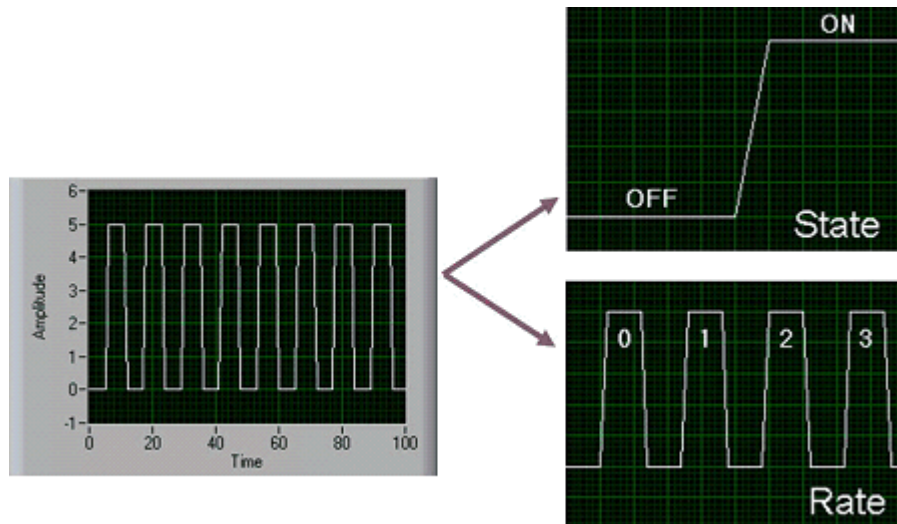
Frequency

All analog signals can be categorized by their frequency. Unlike the level or shape of the signal, frequency cannot be directly measured. The signal must be analyzed using software to determine the frequency information. This analysis is usually done using an algorithm known as the Fourier transform.

When frequency is the most important piece of information, it is important to consider including both accuracy and acquisition speed. Although the acquisition speed for acquiring the frequency of a signal is less than the speed required for obtaining the shape of a signal, the signal must still be acquired fast enough that the pertinent information is not lost while the analog signal is being acquired. The condition that stipulates this speed is known as the Nyquist Sampling Theorem. Speech analysis, telecommunication, and earthquake analysis are some examples of common applications where the frequency of the signal must be known.

3.3 Digital Signals

A digital signal cannot take on any value with respect to time. Instead, a digital signal has two possible levels: high and low. Digital signals generally conform to certain specifications that define characteristics of the signal. Digital signals are commonly referred to as transistor-to-transistor logic (TTL). TTL specifications indicate a digital signal to be low when the level falls within 0 to 0.8 V, and the signal is high between 2 to 5 V. The useful information that can be measured from a digital signal includes the state and the rate.



State

Digital signals cannot take on any value with respect to time. The state of a digital signal is essentially the level of the signal - on or off, high or low. Monitoring the state of a switch - open or closed - is a common application showing the importance of knowing the state of a digital signal.

Rate

The rate of a digital signal defines how the digital signal changes state with respect to time. An example of measuring the rate of a digital signal includes determining how fast a motor shaft spins. Unlike frequency, the rate of a digital signal measures how often a portion of a signal occurs. A software algorithm is not required to determine the rate of a signal

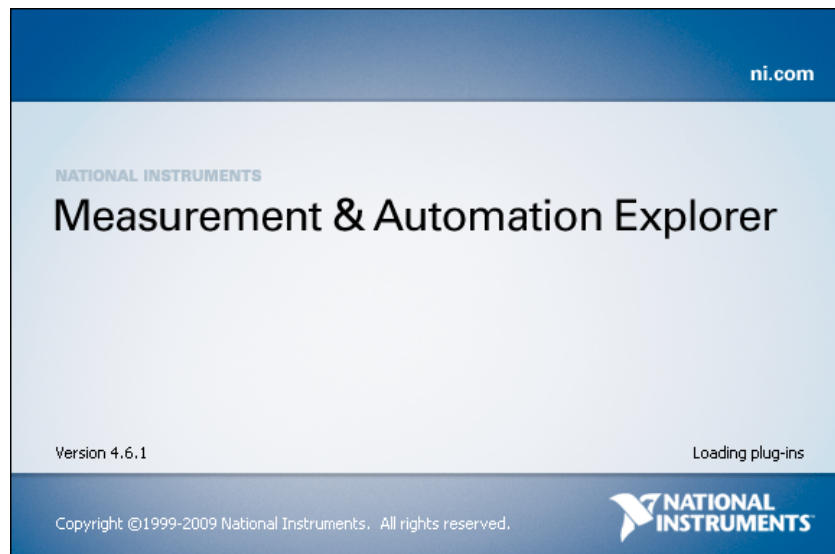
4 MAX

Measurement & Automation Explorer (MAX) provides access to your National Instruments devices and systems.

With MAX, you can:

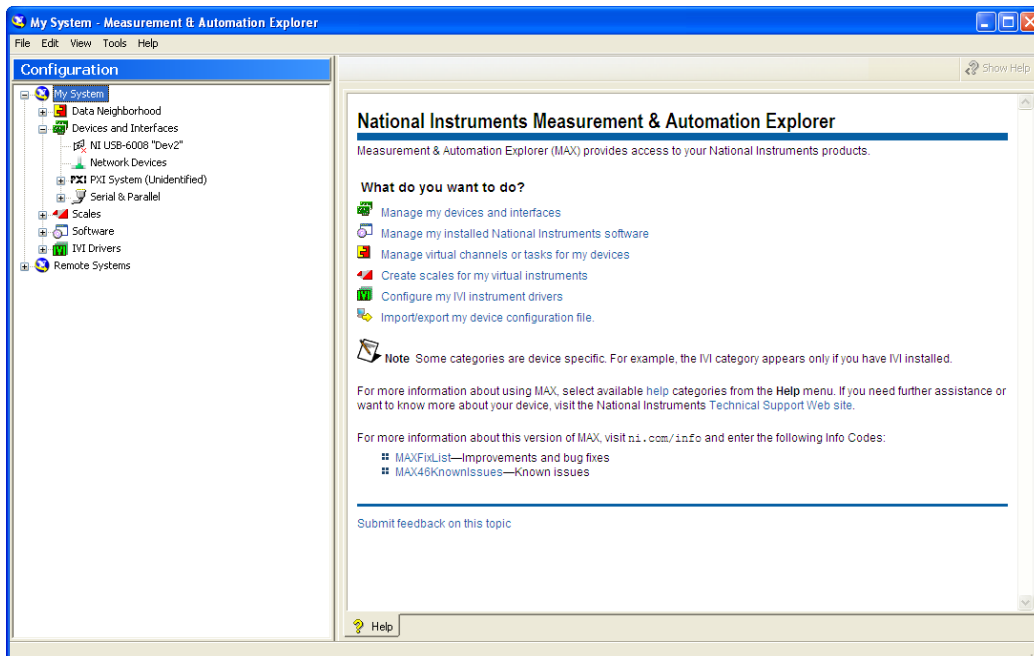
- Configure your National Instruments hardware and software
- Create and edit channels, tasks, interfaces, scales, and virtual instruments
- Execute system diagnostics
- View devices and instruments connected to your system
- Update your National Instruments software

In addition to the standard tools, MAX can expose item-specific tools you can use to configure, diagnose, or test your system, depending on which NI products you install. As you navigate through MAX, the contents of the application menu and toolbar change to reflect these new tools.



LabVIEW installs MAX to establish all devices and channel configuration parameters. MAX reads the information the Device Manager records in the Windows Registry and assigns a logical device number to each DAQ device.

You use the device number to refer to the device in LabVIEW. You can access MAX by selecting Tools»Measurement & Automation Explorer in LabVIEW. This displays the primary MAX window.



Before using a data acquisition board, you must confirm that the software can communicate with the board by configuring the devices. For Windows, the Windows Configuration Manager keeps track of all the hardware installed in the computer, including National Instruments DAQ devices. The Windows Configuration Manager automatically detects and configures Plug & Play (PnP) devices.

Windows Configuration Manager

If you have a PnP device, such as an E Series MIO device, the Windows Configuration Manager automatically detects and configures the device. If you have a non-PnP device, or legacy device, you must configure the device manually using the Add New Hardware option in the Control Panel. You can verify the Windows Configuration by accessing the Device Manager.

5 NI-DAQmx

Driver software is the layer of software for easily communicating with the hardware. It forms the middle layer between the application software and the hardware. Driver software also prevents a programmer from having to do register-level programming or complicated commands in order to access the hardware functions.

Driver software from National Instruments:

- NI-DAQmx
- NI-DAQmx Base

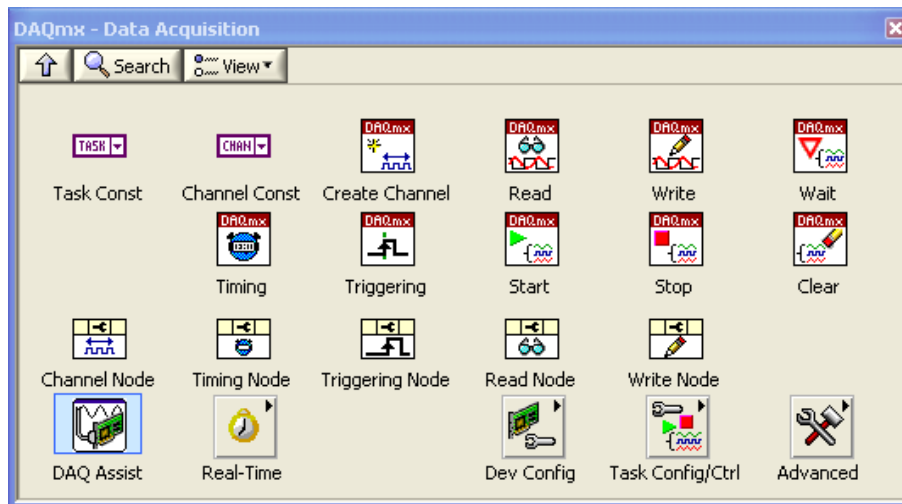
The **DAQ Assistant**, included with NI-DAQmx, is a graphical, interactive guide for configuring, testing, and acquiring measurement data. With a single click, you can even generate code based on your configuration, making it easier and faster to develop complex operations. Because DAQ Assistant is completely menu-driven, you will make fewer programming errors and drastically decrease the time from setting up your DAQ system to taking your first measurement.

NI-DAQmx Base offers a subset of NI-DAQmx functionality on Windows and Linux, Mac OS X, Windows Mobile and Windows CE.

National Instruments DAQ boards have a driver engine that communicates between the board and the application software. There are two driver engines, NI-DAQmx and Traditional NI-DAQ. You can also use the DAQ Assistant, an Express VI that communicates with NI-DAQmx, in LabVIEW to communicate with the DAQ board. In addition, National Instruments provides Measurement & Automation Explorer (MAX) for configuring DAQ boards.

The NI-DAQmx Driver software is the layer of software for easily communicating with the hardware. It forms the middle layer between the application software and the hardware. Driver software also prevents a programmer from having to do register-level programming or complicated commands in order to access the hardware functions.

The DAQmx palette in LabVIEW:



5.1.1 DAQ Assistant

The **DAQ Assistant**, included with NI-DAQmx, is a graphical, interactive guide for configuring, testing, and acquiring measurement data. With a single click, you can even generate code based on your configuration, making it easier and faster to develop complex operations. Because DAQ Assistant is completely menu-driven, you will make fewer programming errors and drastically decrease the time from setting up your DAQ system to taking your first measurement.

Scales

You can configure custom scales for your measurements using MAX. This is very useful when working with sensors. It allows you to bring a scaled value into your application without having to work directly with the raw values. For example, you can use a temperature sensor that represents temperature with a voltage. The conversion equation for the temperature is, $\text{Voltage} \times 100 = \text{Celsius}$. After a scale is set, you can use it in your application program, providing the temperature value, rather than the voltage.

When performing analog input, the task can be timed to:

- Acquire 1 Sample
- Acquire n Samples
- Acquire Continuously

5.2 Simulating a DAQ Device

You can create NI-DAQmx simulated devices in NI-DAQmx 7.4 or later. Using NI-DAQmx simulated devices:

You can try NI products in your application without the hardware.

Later, when you acquire the hardware, you can import the NI-DAQmx simulated device configuration to the physical device using the MAX Portable Configuration Wizard.

You can work on your applications on a portable system and upon returning to the original system, you can easily import your application work.

Creating NI-DAQmx Simulated Devices

To create an NI-DAQmx simulated device, right-click Devices and Interfaces and select Create New. The Create New dialog box prompts you to select a device to add. Select NI-DAQmx Simulated Device and click Finish. In the Choose Device dialog box, select the family of devices for the device you want to simulate. Select the device and click OK. If you select a PXI device, you are prompted to select a chassis number and PXI slot number. If you select an SCXI chassis, the SCXI configuration panels open.

6 DAQ Devices

DAQ hardware acts as the interface between the computer and the outside world. It primarily functions as a device that digitizes incoming analog signals so that the computer can interpret them

A DAQ device (Data Acquisition Hardware) usually has these functions:

- Analog input
- Analog output
- Digital I/O
- Counter/timers

We have different DAQ devices, such as:

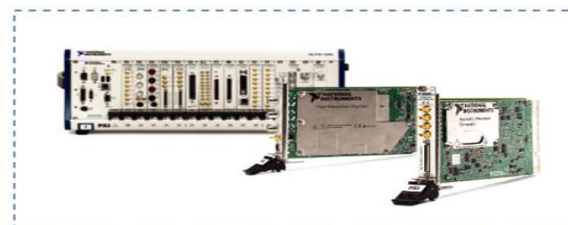
- “**Desktop** DAQ devices” where you need to plug a PCI DAQ board into your computer. The software is running on a computer.
- “**Portable** DAQ devices” for connection to the USB port, Wi-Fi connections, etc. The software is running on a computer
- “**Distributed** DAQ devices” where the software is developed on your computer and then later downloaded to the distributed DAQ device.



Distributed



Desktop



PXI: Rugged and Modular



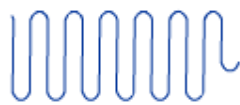
Portable

[www.ni.com]

Most DAQ devices have four standard elements: analog input, analog output, digital I/O, and counters. The DAQ device transfers the measured signals to a computer through different bus structures. For example, you can plug a DAQ device into the PCI bus or the USB port of a computer or the Personal Computer Memory Card International Association (PCMCIA) socket of a laptop. You also can use PXI/CompactPCI to create a portable, versatile, and rugged measurement system.

6.1 Performing Analog-to-Digital Conversion

Analog-to-digital conversion is a process of acquiring and translating signals into digital data so that a computer can process it. Analog-to-digital converters (ADCs) are circuit components that convert a voltage level into a series of ones and zeroes. ADCs sample the analog signal on each rising or falling edge of a sample clock. In each cycle, the ADC takes a snapshot of the analog signal, measures and converts it into a digital value. The ADC obtains and approximates the signal with fixed precision and converts it into a series of digital values.



Analog Signal



Analog-to-Digital Converters



Digital Signal

[www.ni.com]

6.2 Performing Digital-to-Analog Conversion

Digital-to-analog conversion is the opposite of analog-to-digital conversion. In digital-to-analog conversion, the computer generates the data.

6.3 Using Counters

A counter is a digital timing device. You typically use counters for event counting, frequency measurement, period measurement, position measurement, and pulse generation.

6.4 Using Digital I/O

Digital signals are electrical signals that transfer digital data over a wire. These signals typically have only two states: on and off, also known as high and low, or 1 and 0. When sending a digital signal across a wire, the sender applies a voltage to the wire and the receiver uses the voltage level to determine the value being sent. The voltage ranges for each digital value depend on the voltage level standard being used.

Digital signals have many users:

- Digital signals control or measure digital devices such as switches or LEDs

7 NI USB-6008

NI USB-6008 is a simple and low-cost multifunction I/O device from National Instruments.



The device has the following specifications:

- 8 analog inputs (12-bit, 10 kS/s)
- 2 analog outputs (12-bit, 150 S/s)
- 12 digital I/O
- USB connection, No extra power-supply needed
- Compatible with LabVIEW, LabWindows/CVI, and Measurement Studio for Visual Studio .NET
- NI-DAQmx driver software

The NI USB-6008 is well suited for education purposes due to its small size and easy USB connection.

7.1 Connect NI USB-6008 to the PC

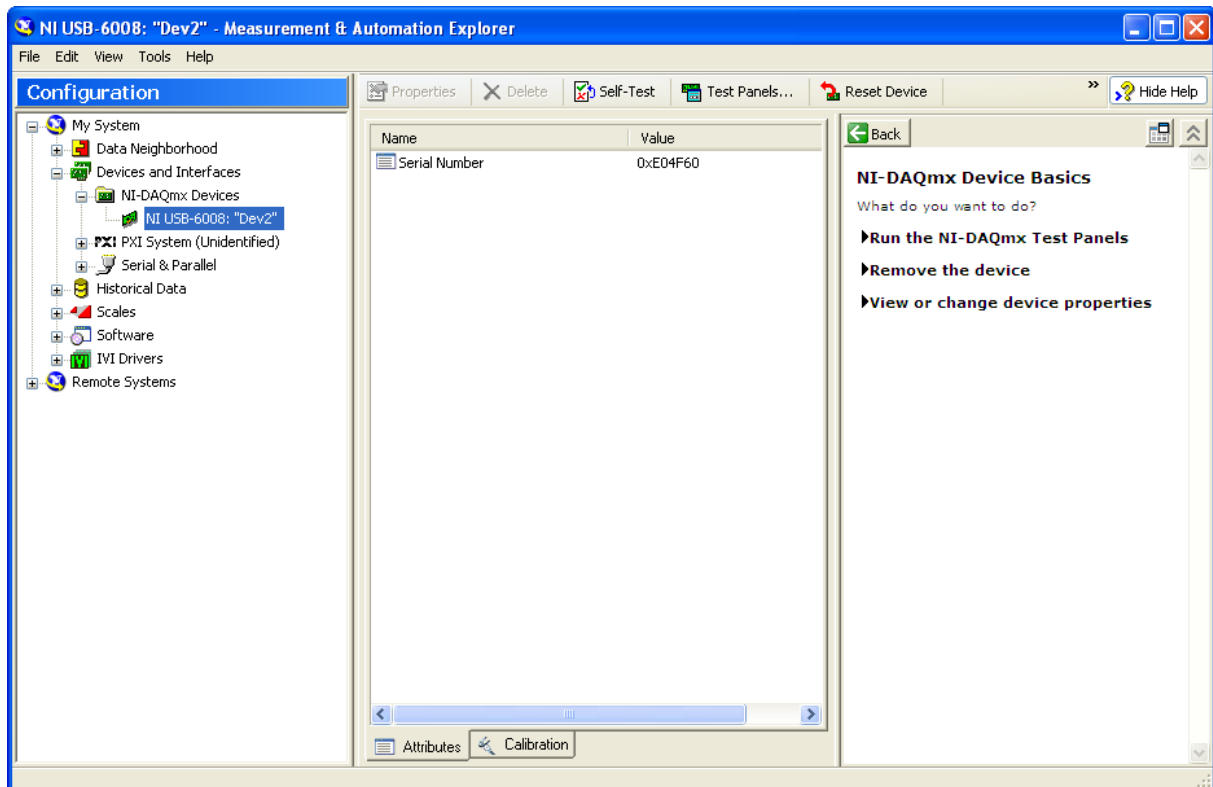
Configuring and testing: USB-6008 can be configured and tested using MAX (Measurement and Automation Explorer), which is installed with the NI-DAQmx Driver Software.

The first time you connect the USB-6008 to the PC, the Windows Hardware Installer Wizard will open.

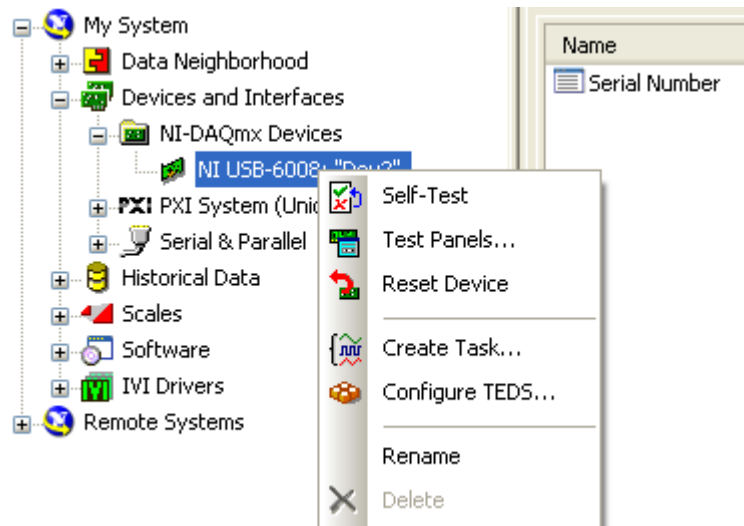
The wizard searches the PC for the necessary driver software for the USB-6008. This driver software was installed along with the installation of the NI-DAQ software. When the wizard has finished the installation of the driver software, the USB-6008 is ready for use.

7.1.1 Testing the USB-6008 in MAX

Before you start to use the USB-6008 in an application, you should test the device in the Measurement and Automation Explorer (**MAX**).



In the MAX window, expand the "Devices and Interfaces" node and then "NI DAQmx Devices". Right-click on the NI USB-6008 device and select "Self-Test".

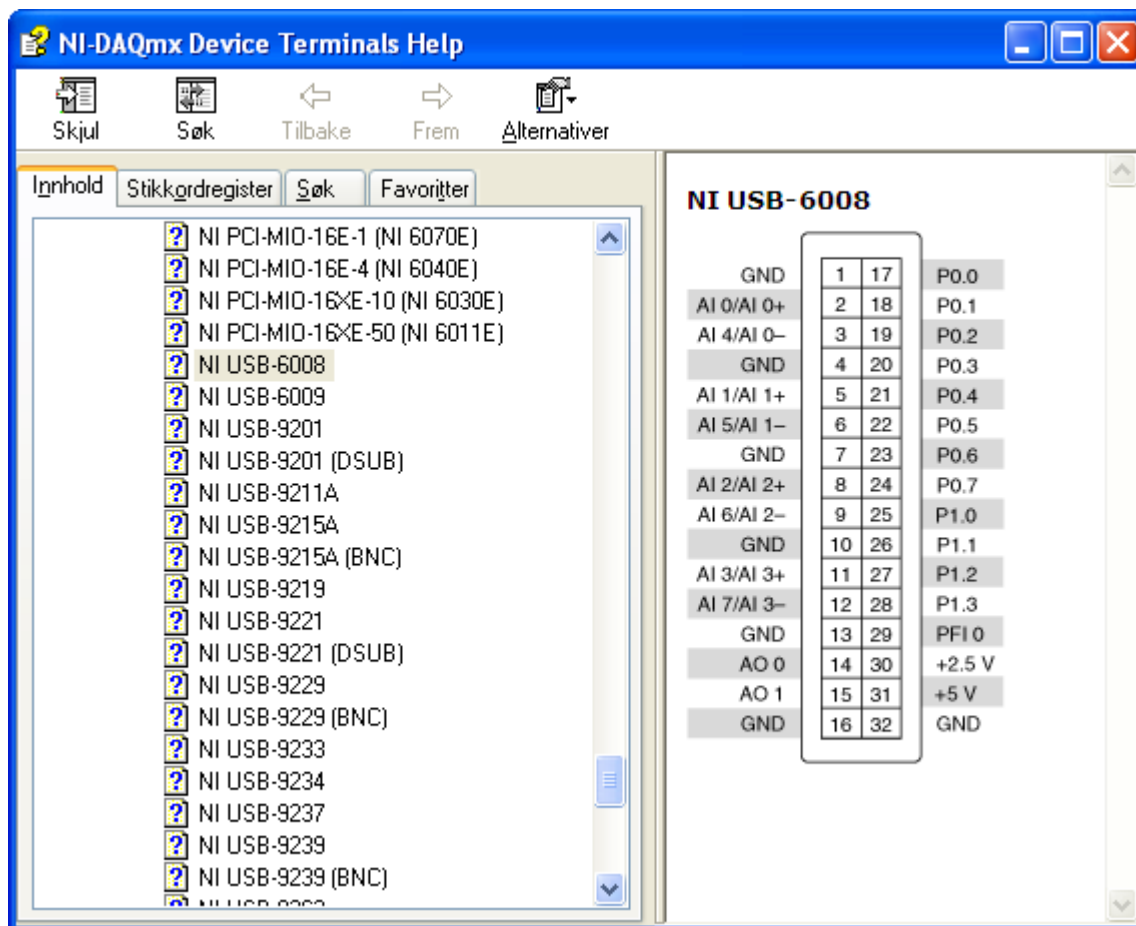


Hopefully the self-test passes without errors. Then, you should test the individual channels of the USB-6008 to check that the input signals are detected correctly by the USB-6008, and that the output signals generated by the USB-6009 have correct values. This I/O can be tested in several ways, depending on which channels you actually want to test.

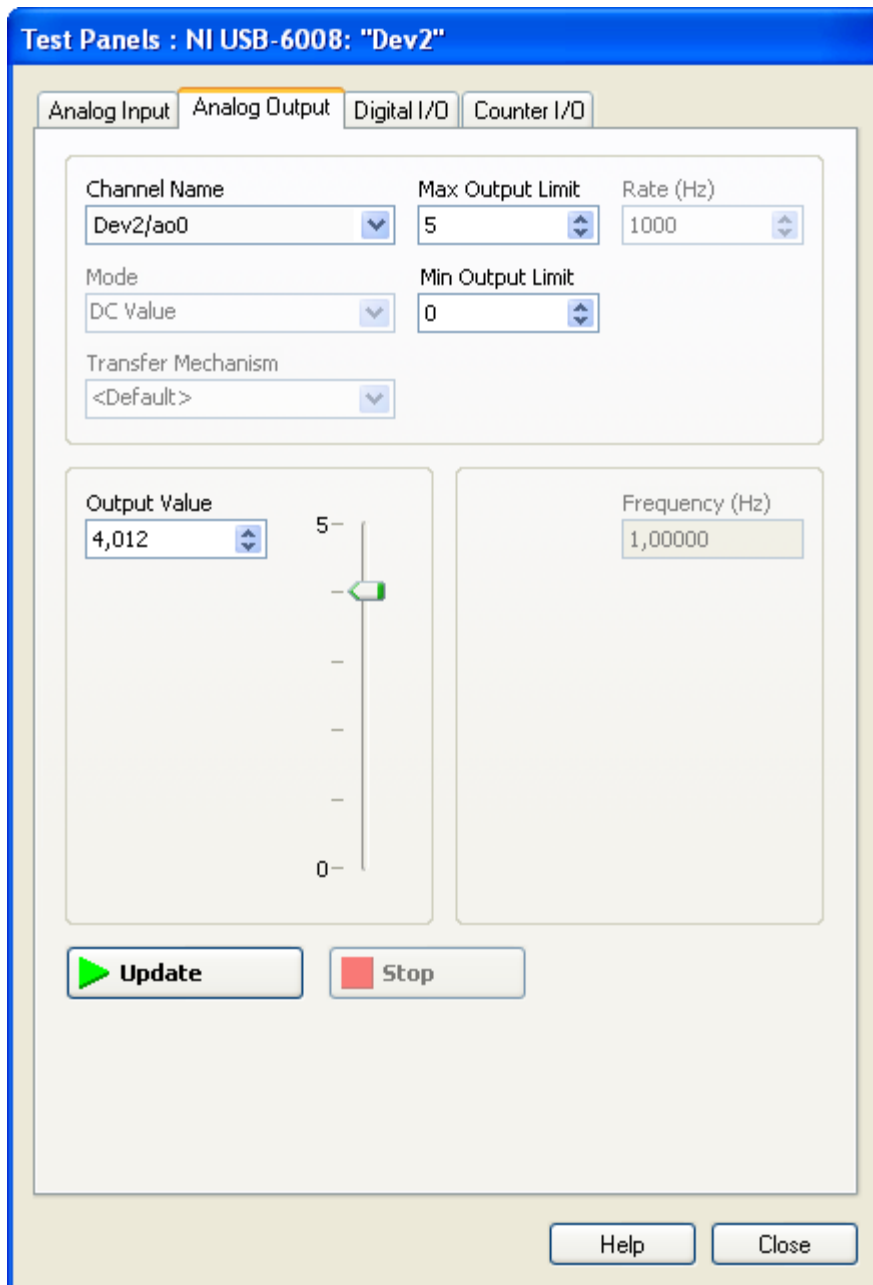
We will perform a simple loopback test:

Here, let us test analog output channel 0 (AO0) and the analog input channel 0 (AI0) to see if they work correctly. We will perform a very simple test, which is sufficient if we are to check that both AO0 and AI0 work correctly. The test procedure, which is denoted *loopback*, is to connect the AI0 channel to the AO0 channel. Then we generate some legal voltage at AO0. If AI0 detects the same voltage, we know that both AO0 and AI0 work. (We may then repeat this procedure for other channels.) If for some reason AI0 detects some other voltage than the value we set for AO0, then there is an error in either the AI0 channel or in the the AO0 channel, and further investigations are necessary.

To prepare for the loopback test, we wire together AI0 and AO0. To see the terminals of the USB-6000, select "Device Pinouts" from the right-click menu.

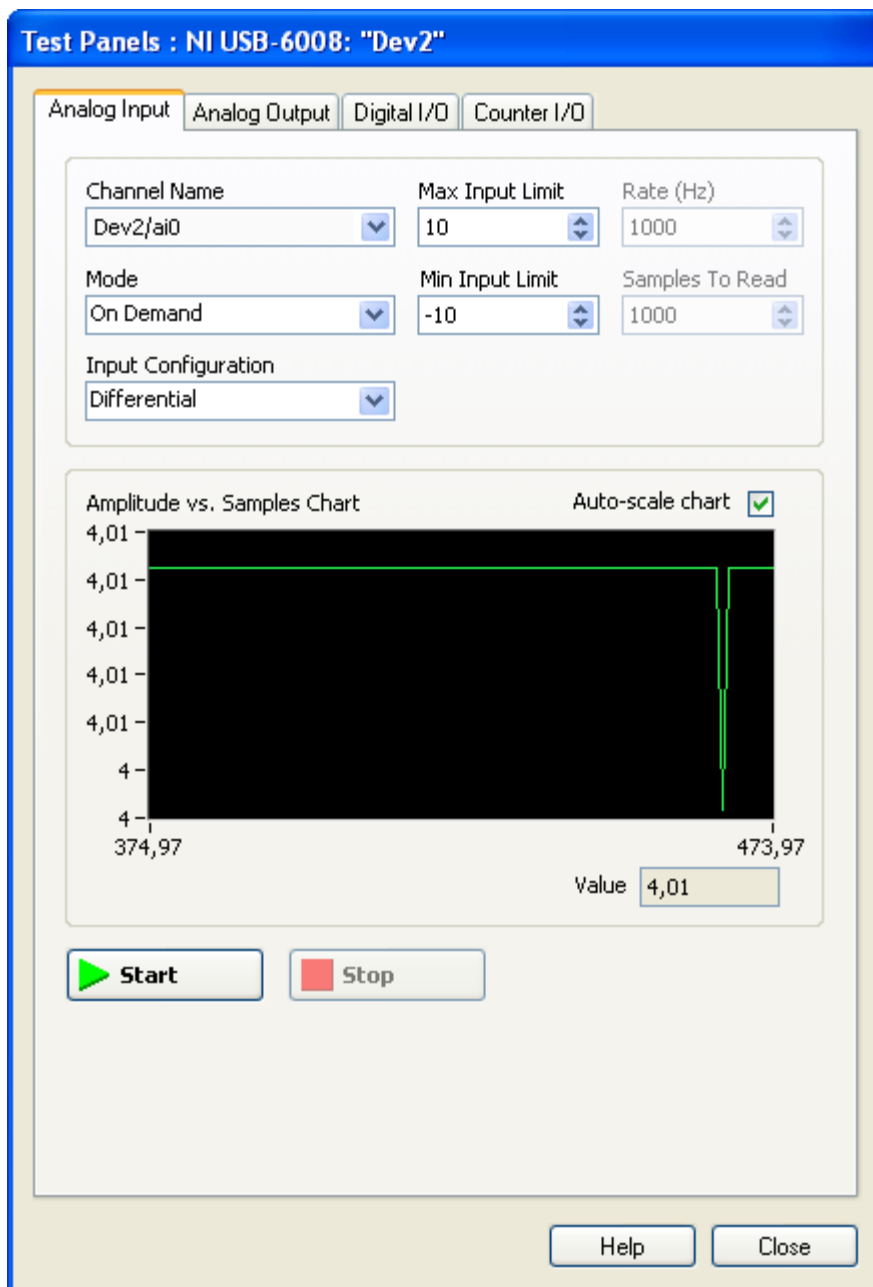


To actually perform the loopback test, right-click on the NI USB-6008 device in MAX, and then select "Test Panels.." in order to open the Test Panels. In the Test Panels window, select the Analog Output tab.



In the Analog Output tab, select any voltage between 0V and 5V.

Next, click the Analog Input tab in the Test Panels window.



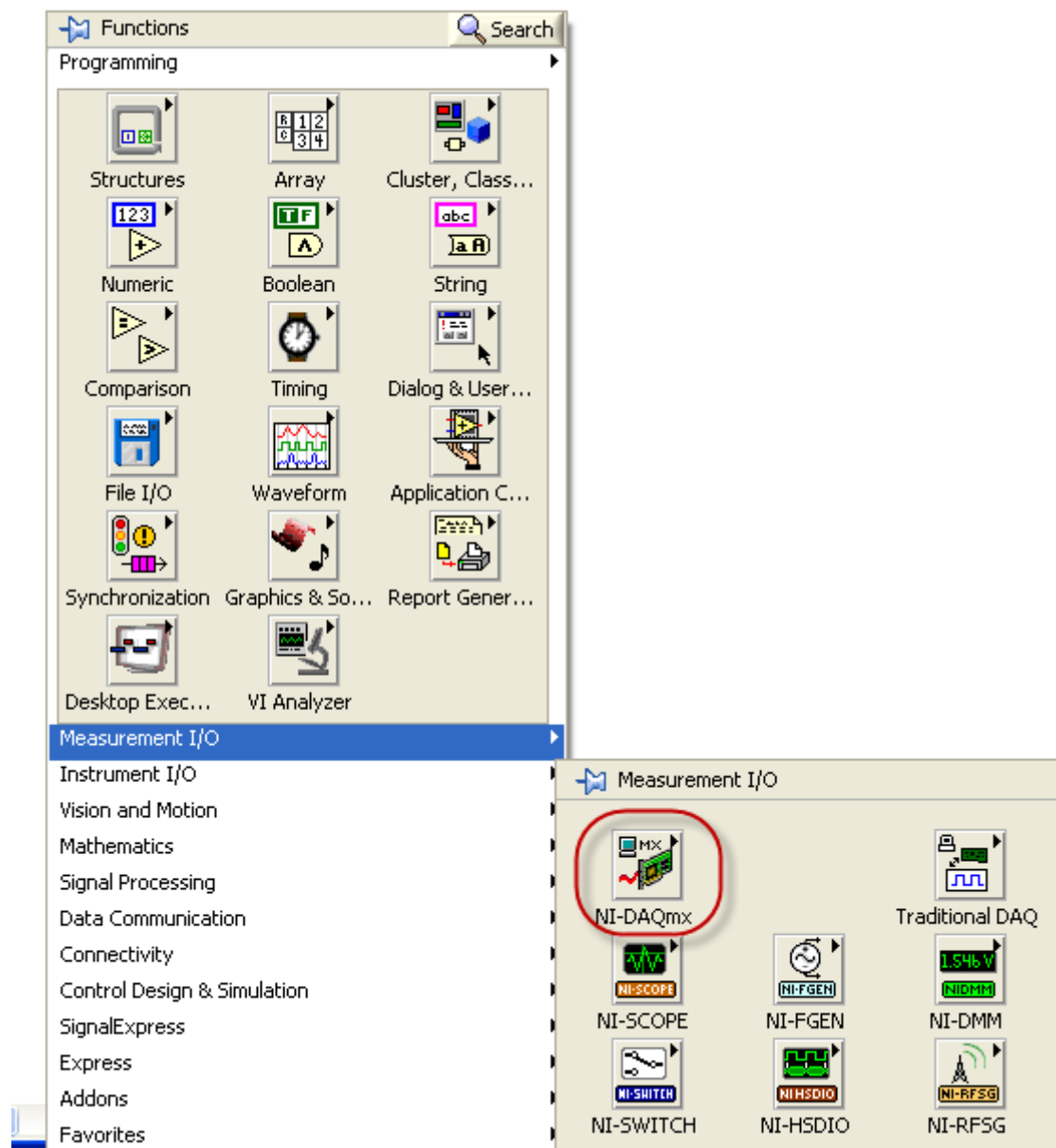
The Analog Input tab should indicate the same (or almost the same) voltage as is set out on A00. There may be a small difference between the values due to the limited resolution in the DA-converter (digital-to-analog) and in the AD-converter (analog-to-digital).



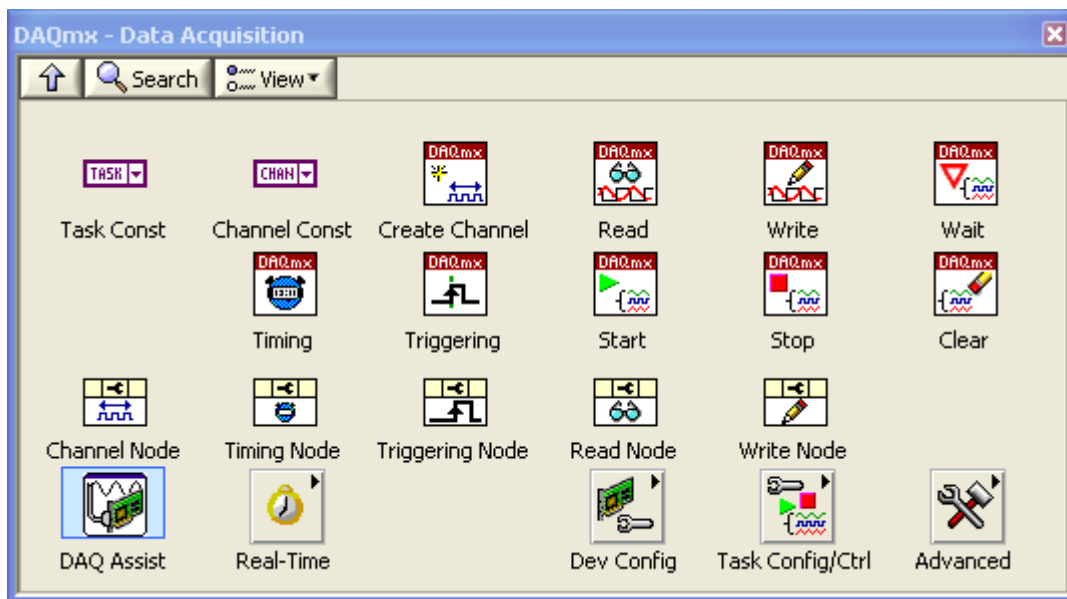
You should also always use a **multi-meter** to check if the voltage levels on the output and input channels are correct according to your settings.

7.2 Using NI USB-6008 in LabVIEW

In order to use the NI USB-6008 in LabVIEW you need to use the DAQmx functions, see Figure below.



DAQmx – Data Acquisition palette:

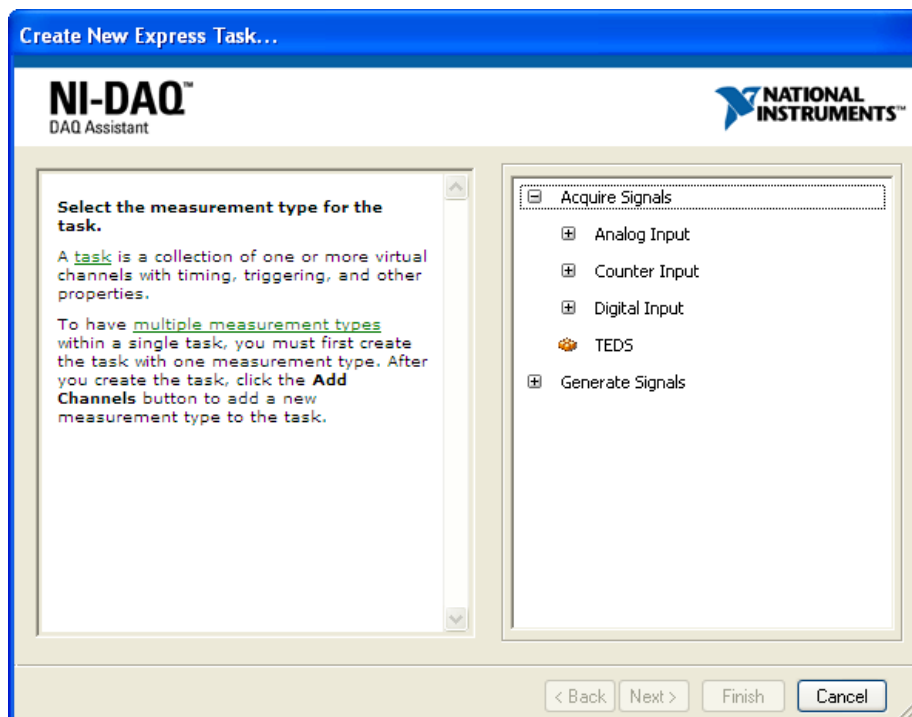


7.3 DAQ Assistant

The easiest way is to use the DAQ Assistant.

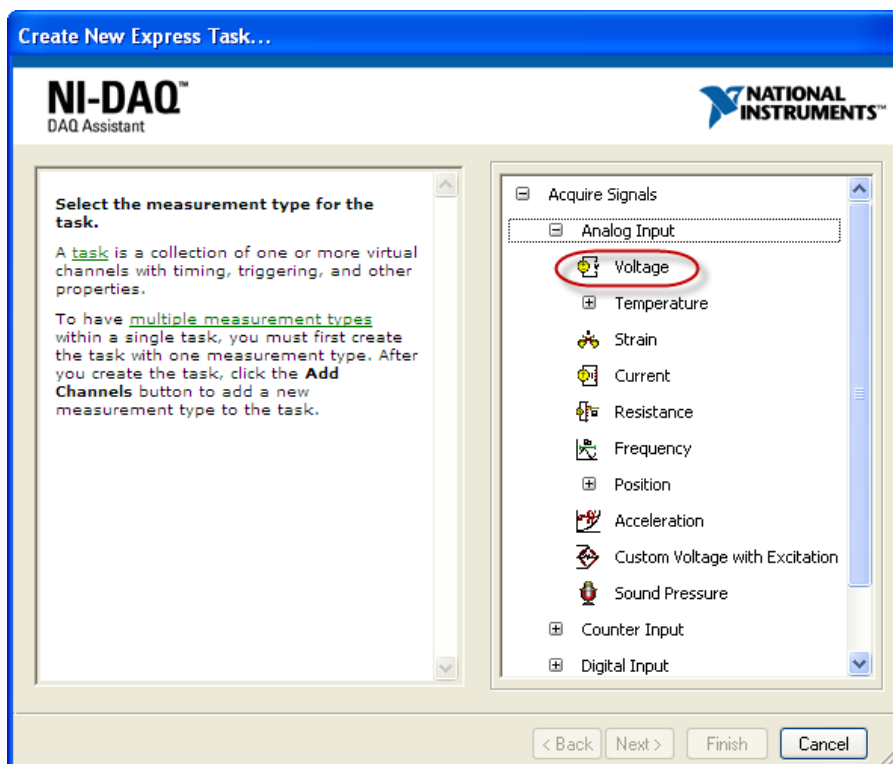
7.3.1 Analog Input

When you drag the DAQ Assistant icon on your Block Diagram, the following window appears:

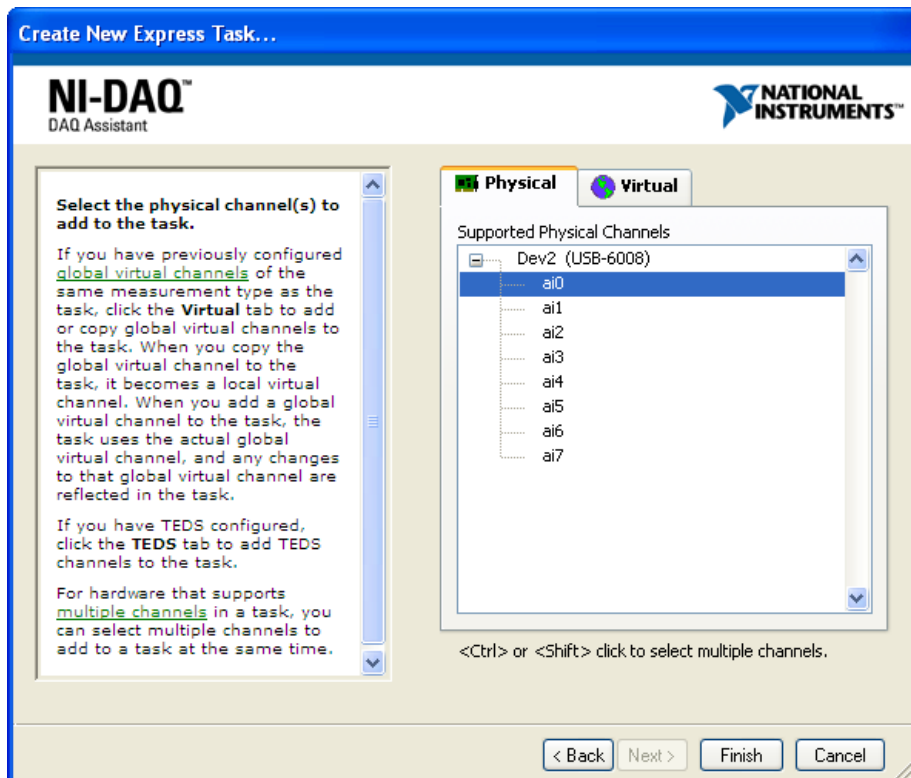


In this window you need to select either “Acquire Signals” (i.e., Input Signals) or “Generate Signals” (i.e., Output Signals).

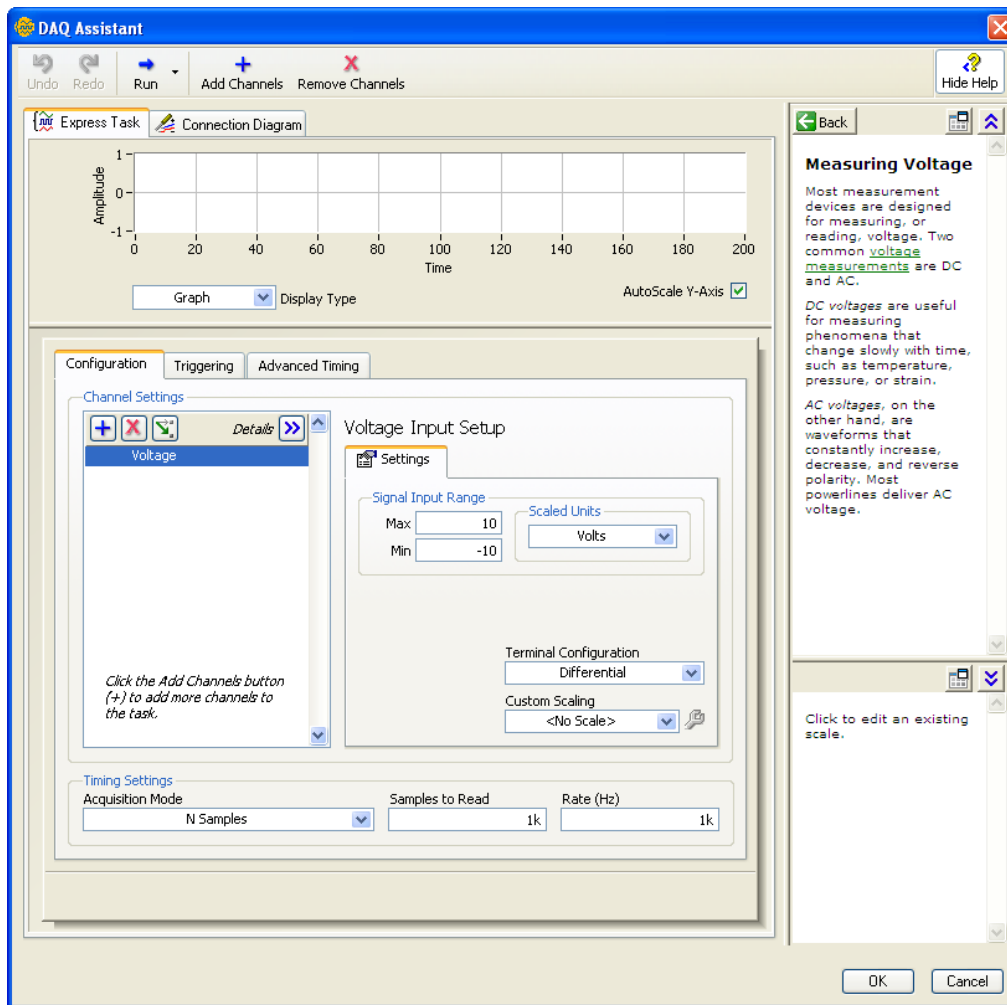
Select Acquire Signals → Analog Input → Voltage.



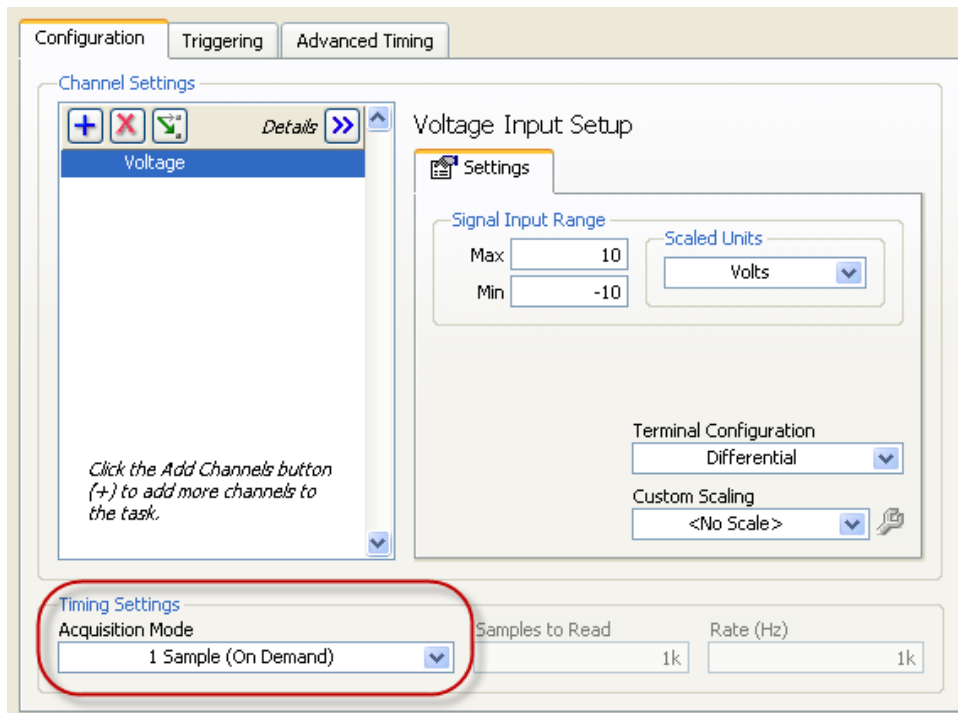
In the next window you select which Analog Input you want to use. Select ai0 (Analog Input channel 0) and click Finish.



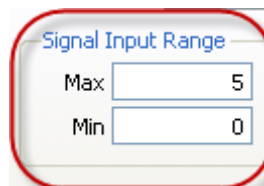
The following window appears:



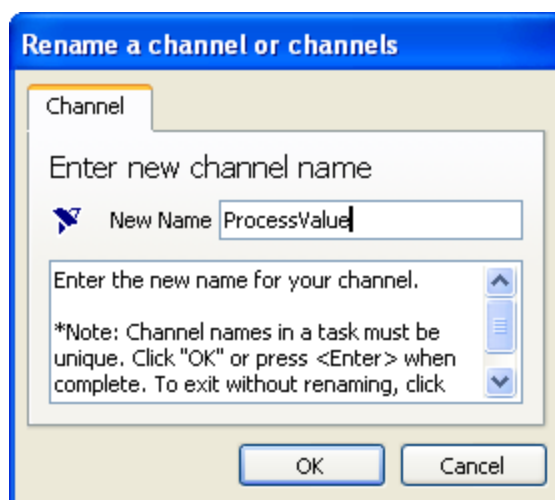
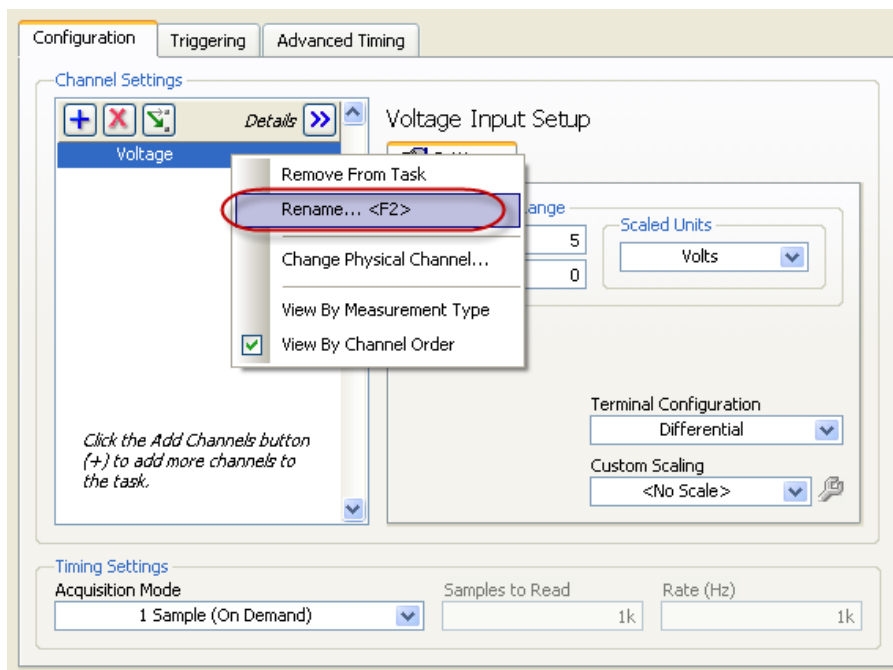
In the **Timing Settings** Select **"1 Sample (On Demand)"**.



The next step is to select the Signal Input Range. A common signal is 0-5V.



You may also rename the name of the channel (right-click on the name):

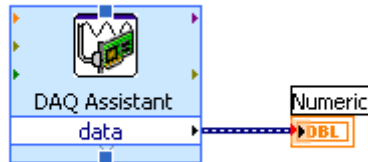


You are now finished with the configuration. Click OK in the DAQ Assistant window. The DAQ Assistant icon appears on the Block Diagram:



Example:

Wire the data output to a numeric indicator like this (and hit the Run button):



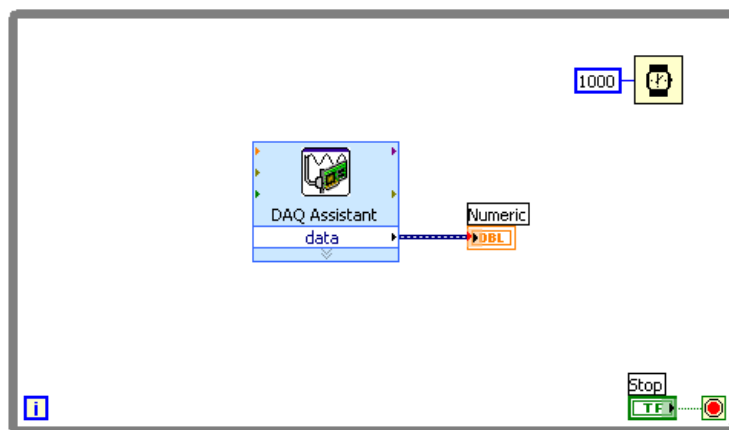
Then numeric indicator will show, e.g., the following value:



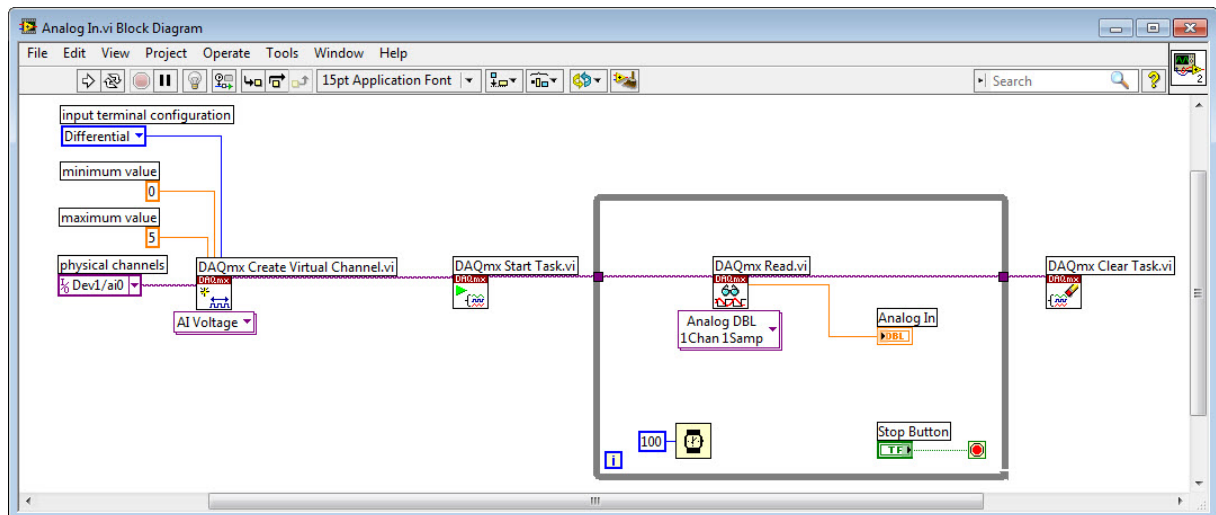
[End of Example]

Example:

If you want a “continuous” acquisition, put a While loop around the DAQ Assistant like this:



We can also communicate with the DAQ device without using the DAQ Assistant:

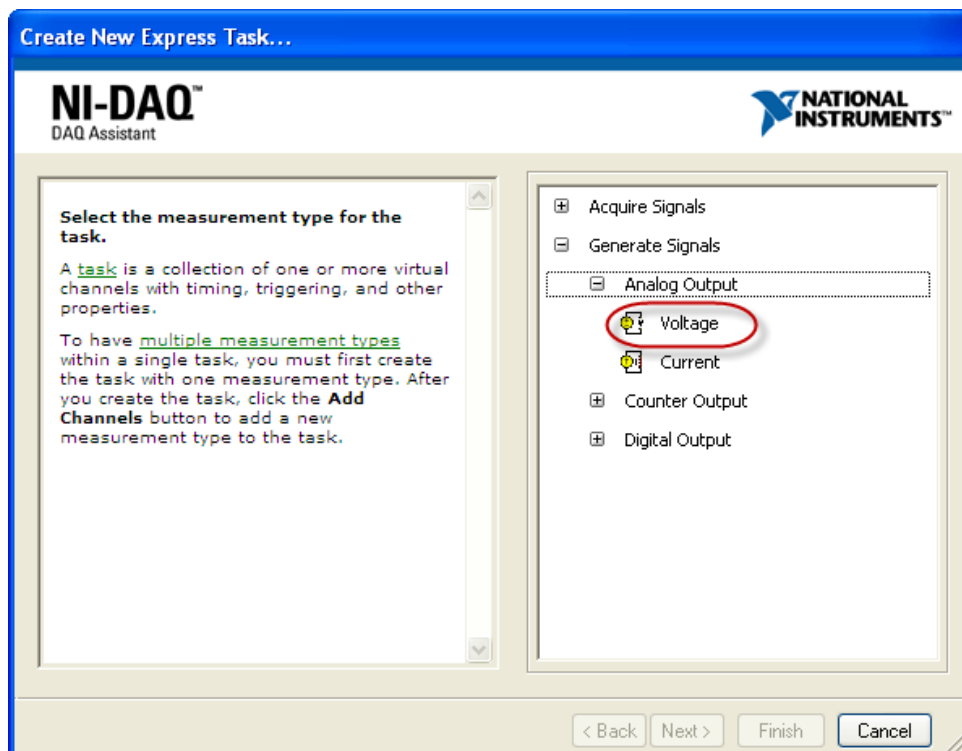


For more advanced applications this approach is recommended.

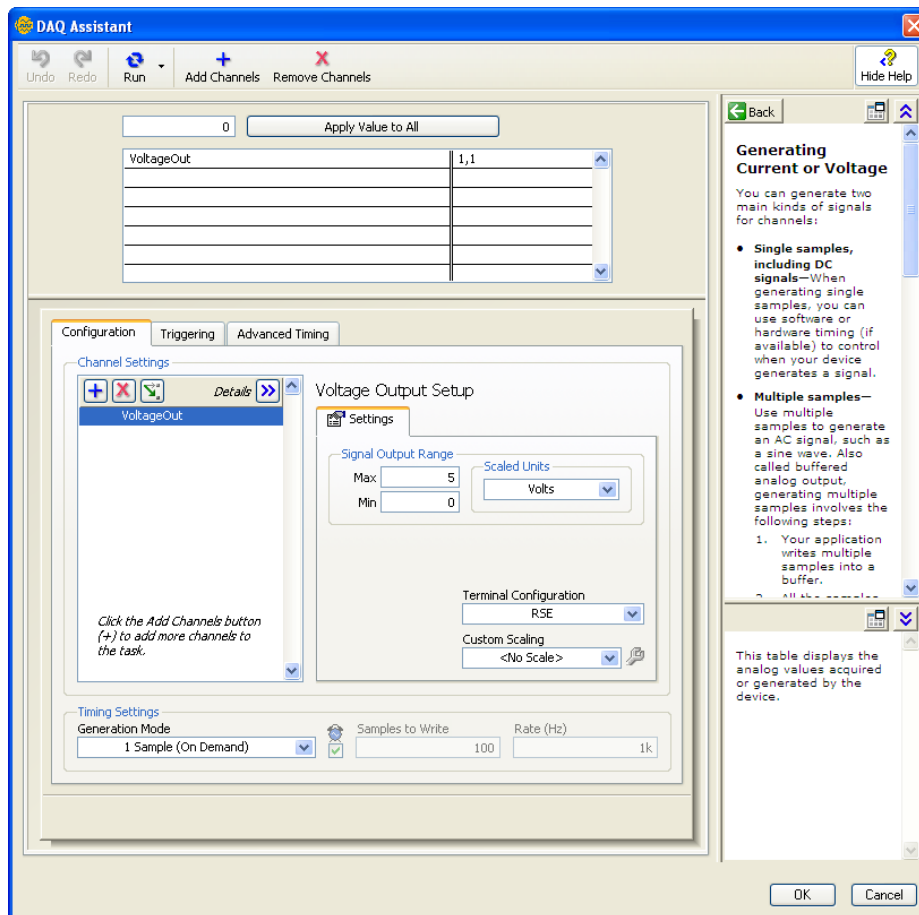
[End of Example]

7.3.2 Analog Output

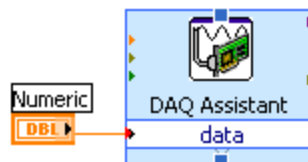
Analog Output is similar.



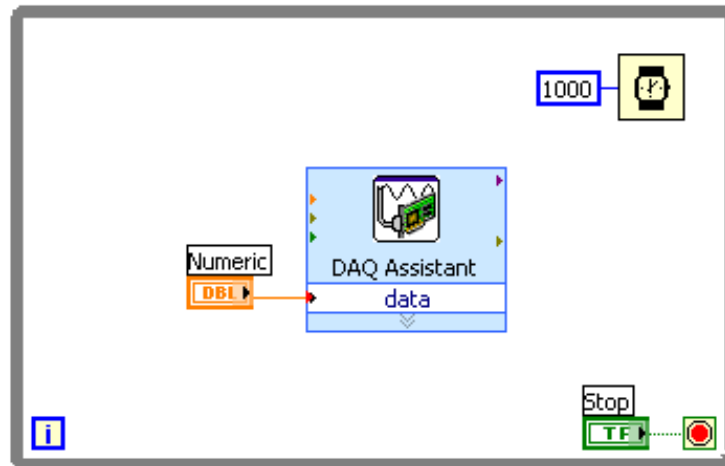




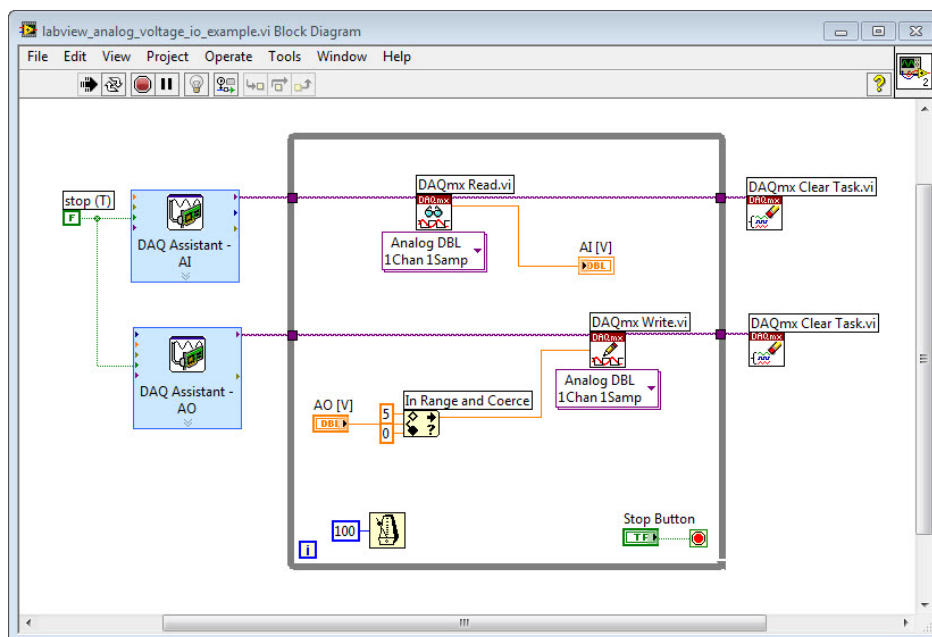
Example:



Or inside a loop for “continuous” writing to the DAQ device:

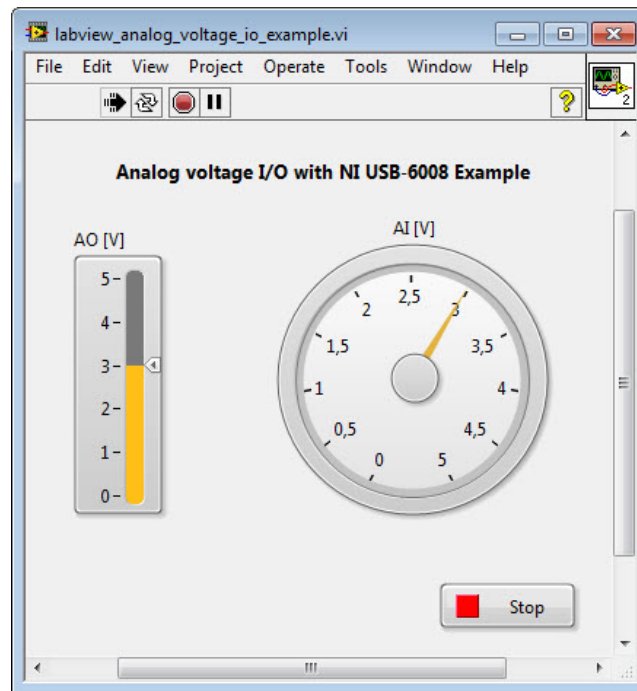


However you should not use the DAQ Assistant inside a loop because of the lack of performance. The following is therefore better:



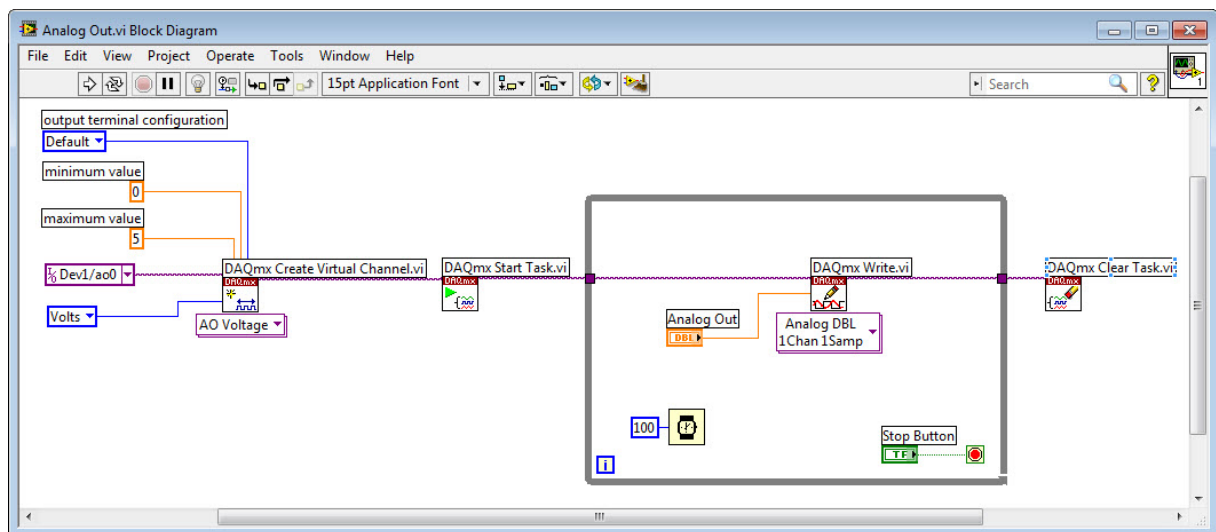
In this example we have used some of the other Vis in the DAQmx palette as well.

The Front Panel may look like this.



In this example we assume that we connect wires for “Analog Out” and “Analog In” together like this (a so-called loopback test).

We can also communicate with the DAQ device without using the DAQ Assistant:



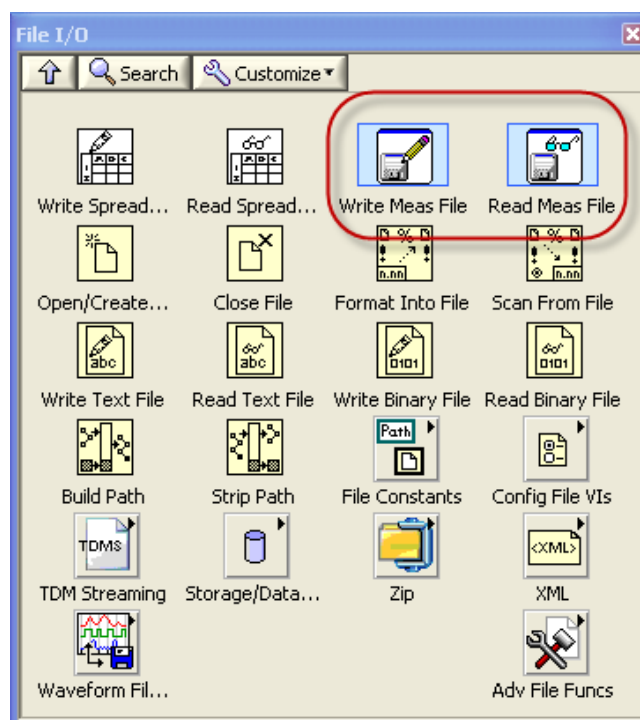
For more advanced applications this approach is recommended.

[End of Example]

8 Logging to File

In many cases you want to write your data you get from the DAQ device to a text file for later use. In this chapter we will learn how to write to a measurement file in LabVIEW. We will also learn how to read the same file.

You can use the “Write to Measurement File” function on the File I/O palette in LabVIEW for writing data to text files. You can save your data in a tab separated text (LVM data file format) or as a binary (TDMS file format) file.



If we use the LVM, it is easy to open and view the data in Notepad.

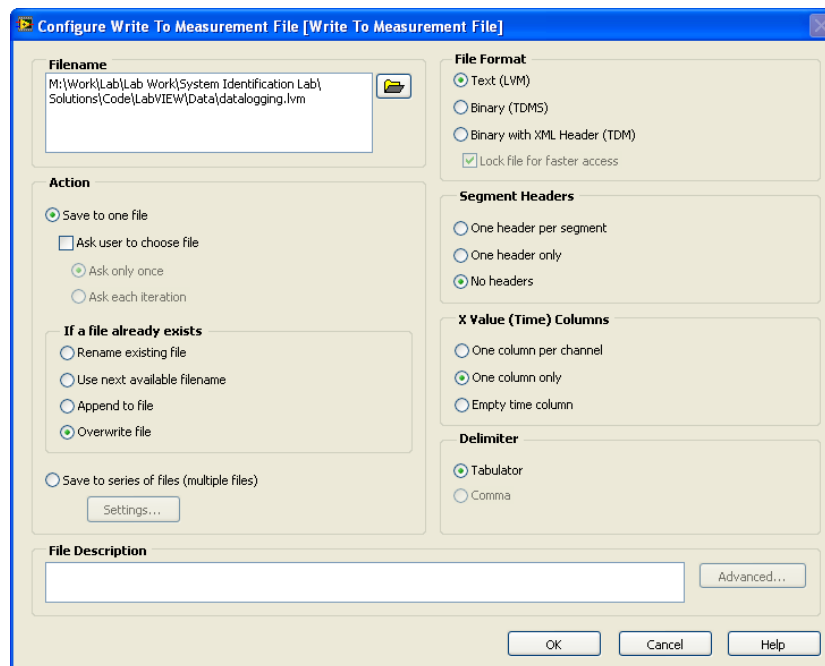
8.1 Writing to Measurement File

We will use the “Write to Measurement File” function in the File I/O palette in LabVIEW for writing data to text files. We will also focus on the LVM data file format, not the TDMS file format which give binary files.



When you drag in the “Write to Measurement File”, a configuration dialog window will automatically pop up.

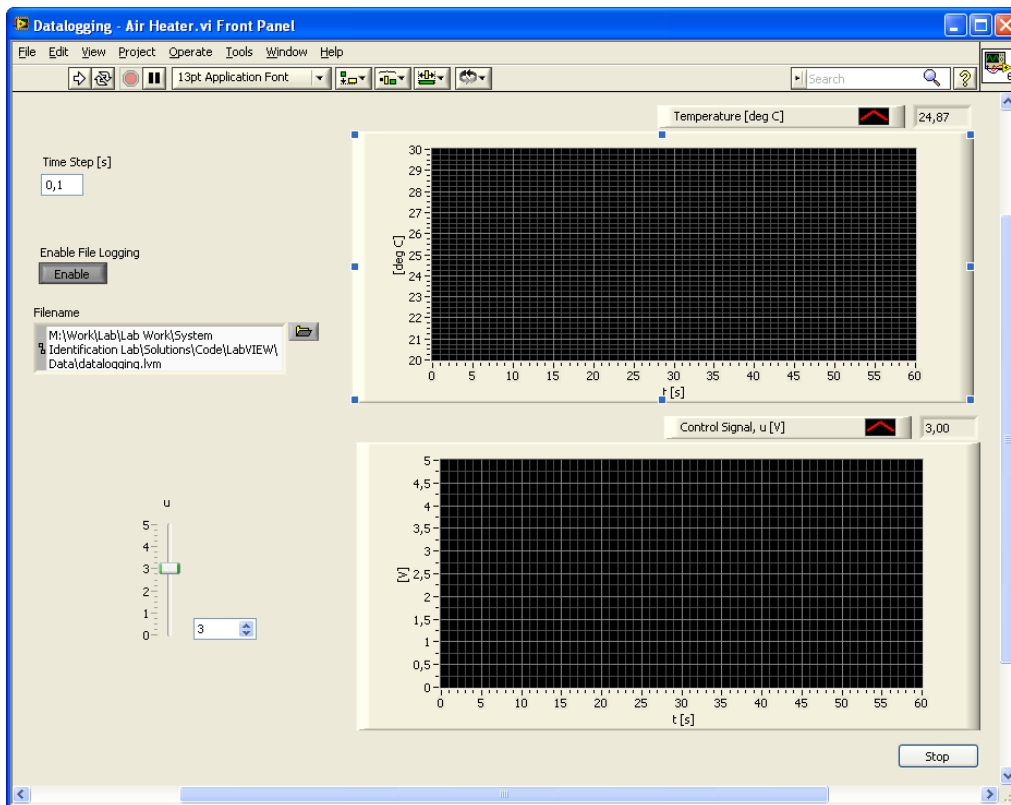
Recommended settings for the “Write To Measurement File” is as follows:



Example:

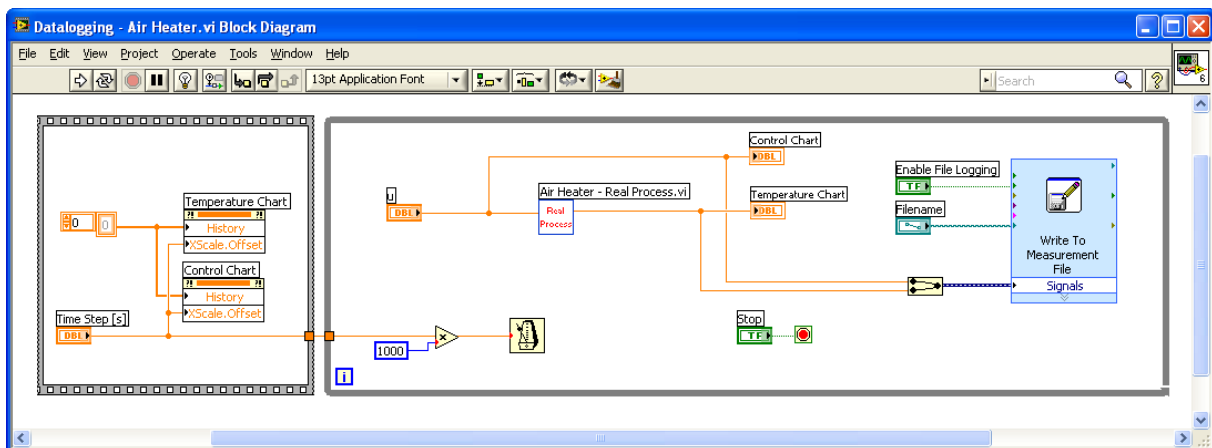
Example of LabVIEW Program that write data to a Measurement File:

Front Panel:

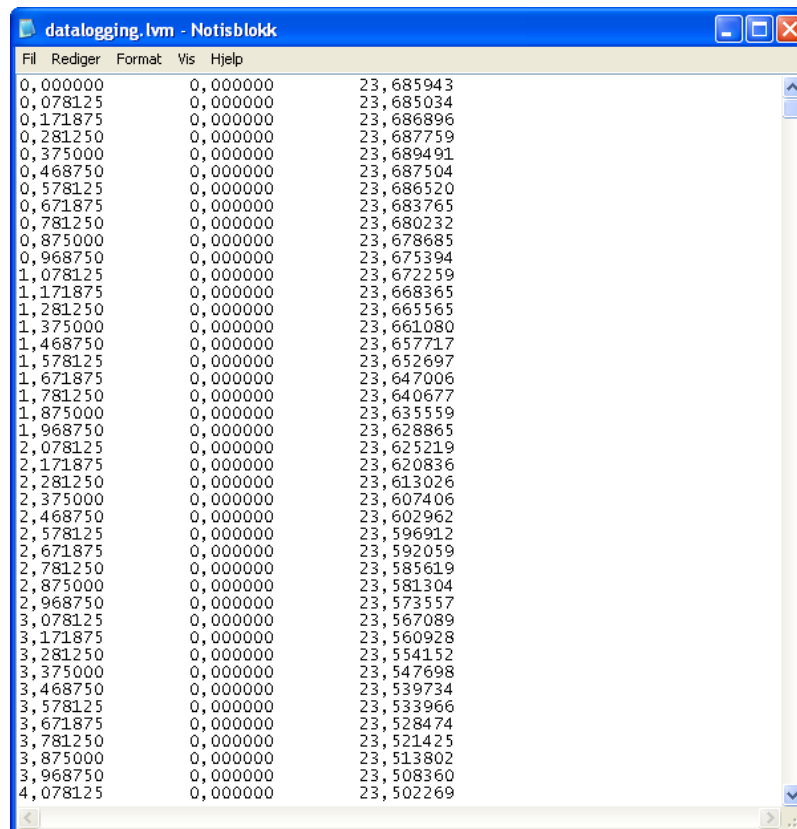


In this application we log data from a process based on a manual control signal. Both the input signal u and the output signal (temperature) is saved to a Measurement File.

Block Diagram:



The LVM file may look something like this:

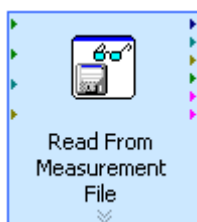


Time	Input Signal	Output Signal
0,000000	0,000000	23,685943
0,078125	0,000000	23,685034
0,171875	0,000000	23,686896
0,281250	0,000000	23,687759
0,375000	0,000000	23,689491
0,468750	0,000000	23,687504
0,578125	0,000000	23,686520
0,671875	0,000000	23,683765
0,781250	0,000000	23,680232
0,875000	0,000000	23,678685
0,968750	0,000000	23,675394
1,078125	0,000000	23,672259
1,171875	0,000000	23,668365
1,281250	0,000000	23,665565
1,375000	0,000000	23,661080
1,468750	0,000000	23,657717
1,578125	0,000000	23,652697
1,671875	0,000000	23,647006
1,781250	0,000000	23,640677
1,875000	0,000000	23,635559
1,968750	0,000000	23,628865
2,078125	0,000000	23,625219
2,171875	0,000000	23,620836
2,281250	0,000000	23,613026
2,375000	0,000000	23,607406
2,468750	0,000000	23,602962
2,578125	0,000000	23,596912
2,671875	0,000000	23,592059
2,781250	0,000000	23,585619
2,875000	0,000000	23,581304
2,968750	0,000000	23,573557
3,078125	0,000000	23,567089
3,171875	0,000000	23,560928
3,281250	0,000000	23,554152
3,375000	0,000000	23,547698
3,468750	0,000000	23,539734
3,578125	0,000000	23,533966
3,671875	0,000000	23,528474
3,781250	0,000000	23,521425
3,875000	0,000000	23,513802
3,968750	0,000000	23,508360
4,078125	0,000000	23,502269

The first column is the time, the second column is the input signal, and the third column is the output signal.

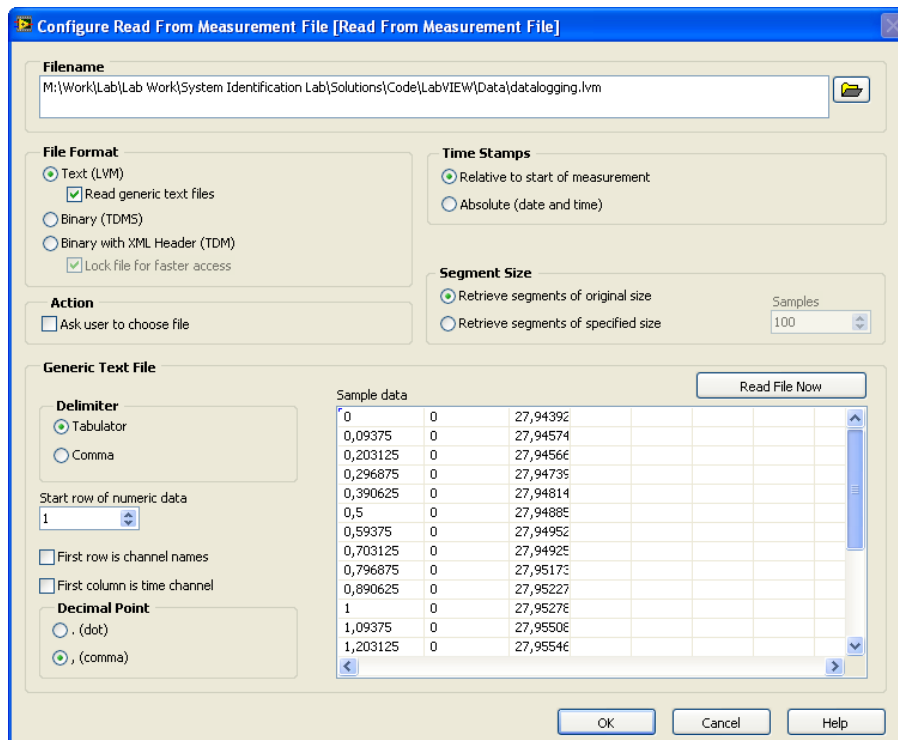
[End of Example]

8.2 Read from Measurement File



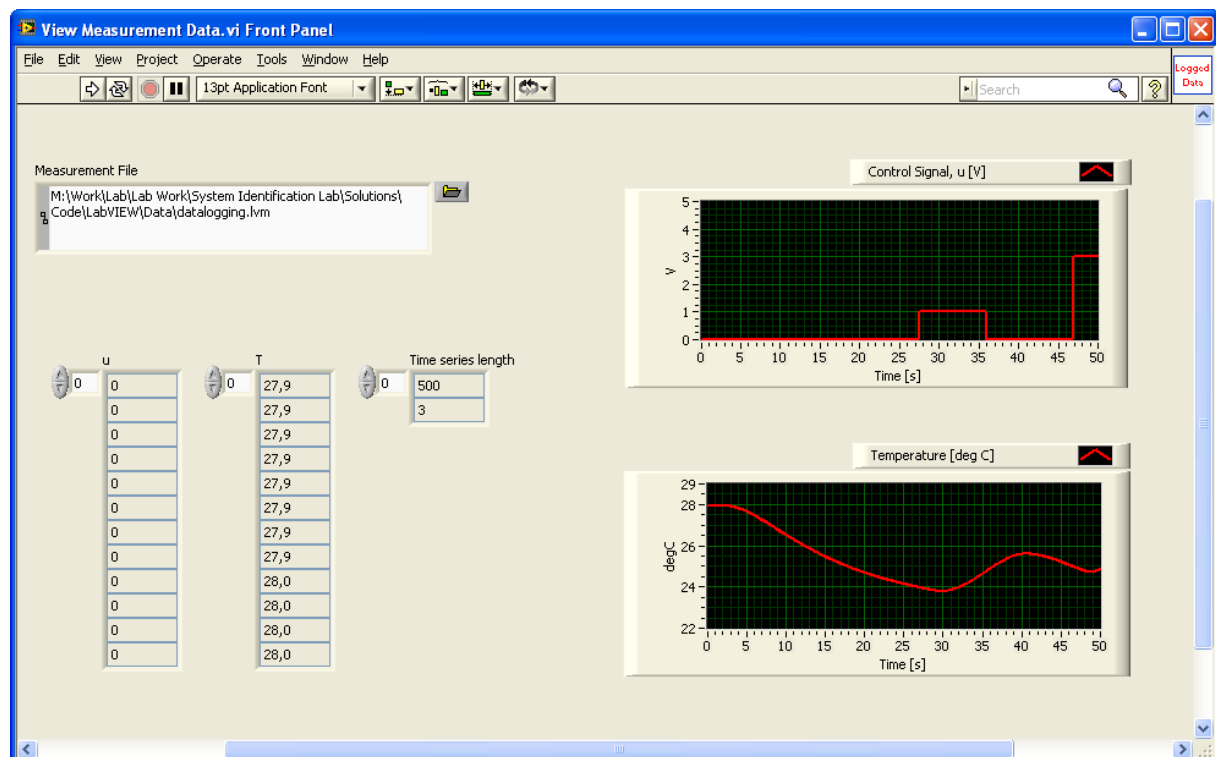
When you drag in the “Read from Measurement File”, a configuration dialog window will automatically pop up.

Recommended settings for the “**Read From Measurement File**”:



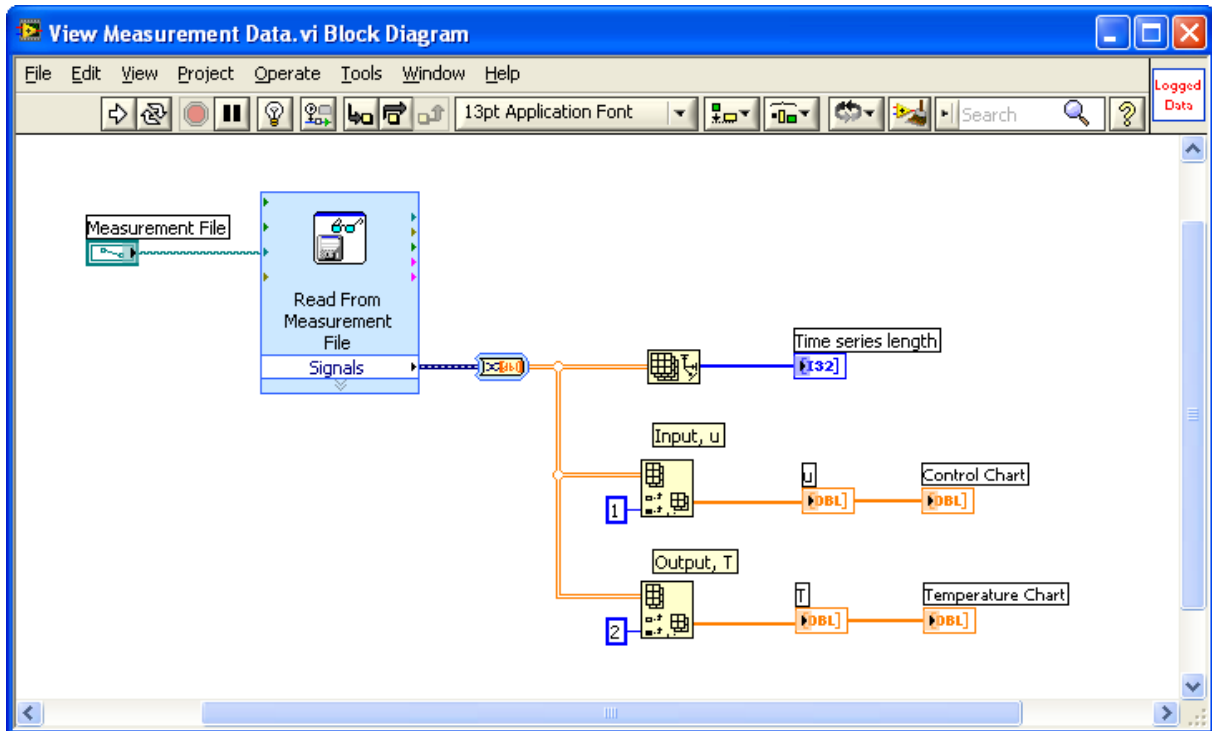
Example:

Example of LabVIEW Program that read data from a Measurement File:



This application reads the data and plots it in 2 different graphs.

Block Diagram:



[End of Example]



Hans-Petter Halvorsen, M.Sc.

E-mail: hans.p.halvorsen@hit.no

Blog: <http://home.hit.no/~hansha/>



University College of Southeast Norway

www.usn.no
