Introduction to LabVIEW

GRAPHICAL PROGRAMMING
FOR ENGINEERS AND SCIENTISTS

LabVIEW® 8

3-Hour Hands-On
Course Goals

• Become comfortable with the LabVIEW environment and data flow execution
• Ability to use LabVIEW to solve problems
• LabVIEW Concepts
  – Acquiring, saving and loading data
  – Find and use math and complex analysis functions
  – Work with data types, such as arrays and clusters
  – Displaying and printing results

This is a list of the objectives of the course.
This course prepares you to do the following:
• Use LabVIEW to create applications.
• Understand front panels, block diagrams, and icons and connector panes.
• Use built-in LabVIEW functions.
• Create and save programs in LabVIEW so you can use them as subroutines.
• Create applications that use plug-in DAQ devices.
This course does not describe any of the following:
• Programming theory
• Every built-in LabVIEW function or object
• Analog-to-digital (A/D) theory
NI does provide free reference materials on the above topics on ni.com.
The LabVIEW Help is also very helpful:
LabVIEW»Help»Search the LabVIEW Help…
Virtual Instrumentation

For more than 25 years, National Instruments has revolutionized the way engineers and scientists in industry, government, and academia approach measurement and automation. Leveraging PCs and commercial technologies, virtual instrumentation increases productivity and lowers costs for test, control, and design applications through easy-to-integrate software, such as NI LabVIEW, and modular measurement and control hardware for PXI, PCI, USB, and Ethernet.

With virtual instrumentation, engineers use graphical programming software to create user-defined solutions that meet their specific needs, which is a great alternative to proprietary, fixed functionality traditional instruments. Additionally, virtual instrumentation capitalizes on the ever-increasing performance of personal computers. For example, in test, measurement, and control, engineers have used virtual instrumentation to downsize automated test equipment (ATE) while experiencing up to a 10 times increase in productivity gains at a fraction of the cost of traditional instrument solutions. Last year 25,000 companies in 90 countries invested in more than 6 million virtual instrumentation channels from National Instruments.
National Instruments LabVIEW is an industry-leading software tool for designing test, measurement, and control systems. Since its introduction in 1986, engineers and scientists worldwide who have relied on NI LabVIEW graphical development for projects throughout the product design cycle have gained improved quality, shorter time to market, and greater engineering and manufacturing efficiency. By using the integrated LabVIEW environment to interface with real-world signals, analyze data for meaningful information, and share results, you can boost productivity throughout your organization. Because LabVIEW has the flexibility of a programming language combined with built-in tools designed specifically for test, measurement, and control, you can create applications that range from simple temperature monitoring to sophisticated simulation and control systems. No matter what your project is, LabVIEW has the tools necessary to make you successful quickly.
Virtual Instrumentation Applications

Virtual instrumentation is applicable in many different types of applications, starting from design to prototyping and deployment. The LabVIEW platform provides specific tools and models to solve specific applications ranging from designing signal processing algorithms to making voltage measurements and can target any number of platforms from the desktop to embedded devices – with an intuitive, powerful graphical paradigm.

With version 8, LabVIEW scales from design and development on PCs to several embedded targets from ruggedized toaster size prototypes to embedded systems on chips. **LabVIEW streamlines system design with a single graphical development platform.** In doing so, LabVIEW encompasses better management of distributed, networked systems because as the targets for LabVIEW grow varied and embedded, you will need to be able to more easily distribute and communicate between various LabVIEW code pieces in your system.
Integrated Hardware Platforms

A virtual instrument consists of an industry-standard computer or workstation equipped with powerful application software, cost-effective hardware such as plug-in boards, and driver software, which together perform the functions of traditional instruments.

Virtual instruments represent a fundamental shift from traditional hardware-centered instrumentation systems to software-centered systems that exploit the computing power, productivity, display, and connectivity capabilities of popular desktop computers and workstations.

Although the PC and integrated circuit technology have experienced significant advances in the last two decades, software truly offers the flexibility to build on this powerful hardware foundation to create virtual instruments, providing better ways to innovate and significantly reduce cost. With virtual instruments, engineers and scientists build measurement and automation systems that suit their needs exactly (user-defined) instead of being limited by traditional fixed-function instruments (vendor-defined).
Section I – LabVIEW Environment

A. Getting Data into your Computer
   • Data Acquisition Devices
     – NI-DAQ
     – Simulated Data Acquisition
     – Sound Card

B. LabVIEW Environment
   • Front Panel / Block Diagram
   • Toolbar / Tools Palette

C. Components of a LabVIEW Application
   • Creating a VI
   • Data Flow Execution

D. Additional Help
   • Finding Functions
   • Tips for Working in LabVIEW

© National Instruments Corporation

Introduction to LabVIEW Hands-On
This LabVIEW course is designed for audiences with or without access to National Instruments hardware.

Each exercise is divided into three tracks, A, B, and C:

**Track A** is designed to be used with hardware supported by the National Instruments DAQmx driver. This includes most USB, PCI, and PXI data acquisition devices with analog input. Some signal conditioning and excitation is required to use a microphone with a DAQ device. (Some sensors, like microphones, require external power to work (excitation)).

**Track B** is designed to be used with no hardware. Hardware can be simulated with the NI-DAQmx Driver Version 7.5 and newer. A simulated NI-DAQmx device is a replica of a device created using the NI-DAQmx Simulated Device option in the Create New menu of MAX for the purpose of operating a function or program without hardware. An NI-DAQmx simulated device behaves similar to a real device. Its driver is loaded, and programs using it are fully verified.

**Track C** is designed to be used with a standard sound card and microphone. LabVIEW includes simple VIs for analog input and analog output using the soundcard built into many PCs. (This is very convenient for laptops because the soundcard and microphone are usually already built-in.)
Setting-Up Your Hardware for Your Selected Track

Track A – NI Data Acquisition with Microphone: USB-6009 with Microphone & LED
Suggested Hardware:

<table>
<thead>
<tr>
<th>Qty</th>
<th>Part Number</th>
<th>Description</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>779321-22</td>
<td>Low-cost USB DAQ</td>
<td>National Instruments</td>
</tr>
<tr>
<td>1</td>
<td>270-092</td>
<td>Electret Microphone</td>
<td>RadioShack</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>100 Ohm Resistor</td>
<td>RadioShack</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>220 Ohm Resistor</td>
<td>RadioShack</td>
</tr>
<tr>
<td>1</td>
<td>276-307</td>
<td>Light Emitting Diode (LED)</td>
<td>RadioShack</td>
</tr>
</tbody>
</table>

The following schematic was drawn with Multisim, a widely used SPICE schematic capture and simulation tool. Visit http://www.electronicsworkbench.com for more info.

Track B – Simulated NI Data Acquisition: NI-DAQ Software version 8.0 or newer

Track C – Third-party Soundcard: Soundcard and Microphone
Suggested Hardware:

<table>
<thead>
<tr>
<th>Qty</th>
<th>Part Number</th>
<th>Description</th>
<th>Supplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Standard Plug-in PC Microphone*</td>
<td>RadioShack</td>
</tr>
</tbody>
</table>

* Laptops often have a built-in microphone (no plug-in microphone is required)
What type of device should I use?

There are many types of data acquisition and control devices on the market. A few have been highlighted above. The trade-off usually falls between sampling rate (samples/second), resolution (bits), number of channels, and data transfer rate (usually limited by “bus” type: USB, PCI, PXI, etc.). Multifunction DAQ (data acquisition) devices are ideal because they can be used in a wide range of applications.

USB-6008 & USB-6009 Low-Cost USB DAQ

The National Instruments USB-6009 provides basic data acquisition functionality for applications such as simple data logging, portable measurements, and academic lab experiments. The NI USB-6008 and NI USB-6009 are ideal for students. Create your own measurement application by programming the NI USB-6009 using LabVIEW and NI-DAQmx driver software for Windows. For Mac OS X and Linux users, download and use the NI-DAQmx Base driver.

NI USB-6009 Specifications:
- Eight 14-bit analog inputs
- 12 digital I/O lines
  - 2 analog outputs
  - 1 counter

http://www.ni.com/daq/
What is MAX?

• MAX stands for Measurement & Automation Explorer.
• MAX configures and organizes all your National Instruments DAQ, PCI/PXI instruments, GPIB, IMAQ, IVI, Motion, VISA, and VXI devices.
• Used for configuring and testing devices.

The next level of software we are concerned with is called Measurement & Automation Explorer (MAX). MAX is a software interface that gives you access to all of your National Instruments DAQ, GPIB, IMAQ, IVI, Motion, VISA, and VXI devices. The shortcut to MAX will be placed on your desktop after installation. A picture of the icon is shown above. MAX is mainly used to configure and test your National Instruments hardware, but it does offer other functionality such as checking to see if you have the latest version of NI-DAQ installed. When you run an application using NI-DAQmx, the software reads the MAX configuration to determine the devices you have configured. Therefore, you must configure DAQ devices first with MAX.

The functionality of MAX is broken into seven categories:

• Data Neighborhood
• Devices and Interfaces
• IVI Instruments
• Scales
• Historical Data
• Software
• VI Logger Tasks

For this course, we will focus on Data Neighborhood, Devices and Interfaces, Scales, and Software. We will now step through each one of these categories and learn about the functionality each one offers.
Exercise 1 – Testing Your Device (Track A)

In this exercise you will use Measurement and Automation Explorer (MAX) to test your NI USB-6009 DAQ device.

1. Launch MAX by double-clicking the icon on the desktop or by selecting Start»Programs»National Instruments»Measurement & Automation.

2. Expand the Devices and Interfaces section to view the installed National Instruments devices. MAX displays the National Instruments hardware and software in the computer.

3. Expand the NI-DAQmx Devices section to view the installed hardware that is compatible with NI-DAQmx. The device number appears in quotes following the device name. The data acquisition VIs use this device number to determine which device performs DAQ operations. You will see your hardware listed as NI USB-6009: “Dev1”.

4. Perform a self-test on the device by right-clicking it in the configuration tree and choosing Self-Test or clicking “Self-Test” along the top of the window. This tests the system resources assigned to the device. The device should pass the test because it is already configured.

5. Check the pinout for your device. Right-click the device in the configuration tree and select Device Pinouts or click “Device Pinouts” along the top of the center window.

6. Open the test panels. Right-click the device in the configuration tree and select Test Panels… or click “Test Panels…” along the top of the center window. The test panels allow you to test the available functionality of your device, analog input/output, digital input/output, and counter input/output without doing any programming.

7. On the Analog Input tab of the test panels, change Mode to “Continuous” and Rate to 10,000 Hz. Click “Start” and hum or whistle into your microphone to observe the signal that is plotted. Click “Finish” when you are done.

8. On the Digital I/O tab notice that initially the port is configured to be all input. Observe under Select State the LEDs that represent the state of the input lines. Click the “All Output” button under Select Direction. Notice you now have switches under Select State to specify the output state of the different lines. Toggle line 0 and watch the LED light up. Click “Close” to close the test panels.

9. Close MAX.
(End of Exercise)
In this exercise you will use Measurement and Automation Explorer (MAX) to configure a simulated DAQ device.

1. Launch MAX by double-clicking the icon on the desktop or by selecting `Start»Programs»National Instruments»Measurement & Automation`.

2. Expand the **Devices and Interfaces** section to view the installed National Instruments devices. MAX displays the National Instruments hardware and software in the computer. The device number appears in quotes following the device name. The data acquisition VIs use this device number to determine which device performs DAQ operations.

3. Create a simulated DAQ device for use later in this course. Simulated devices are a powerful tool for development without having hardware physically installed in your computer. Right-click **Devices and Interfaces** and select **Create New…»NI-DAQmx Simulated Device**. Click “Finish”.

4. Expand the M Series DAQ section. Select **PCI-6220** or any other PCI device of your choice. Click “OK”.

5. The NI-DAQmx Devices folder will expand and you will see a new entry for PCI-6220: “Dev1”. You have now created a simulated device!

6. Perform a self-test on the device by right-clicking it in the configuration tree and choosing **Self-Test** or clicking “Self-Test” along the top of the window. This tests the system resources assigned to the device. The device should pass the test because it is already configured.

7. Check the pinout for your device. Right-click the device in the configuration tree and select **Device Pinouts** or click “Device Pinouts” along the top of the center window.

8. Open the test panels. Right-click the device in the configuration tree and select **Test Panels…** or click “Test Panels…” along the top of the center window. The test panels allow you to test the available functionality of your device, analog input/output, digital input/output, and counter input/output without doing any programming.

9. On the **Analog Input** tab of the test panels, change **Mode** to “Continuous”. Click “Start” and observe the signal that is plotted. Click “Stop” when you are done.
10. On the **Digital I/O** tab notice that initially the port is configured to be all input. Observe under **Select State** the LEDs that represent the state of the input lines. Click the “All Output” button under **Select Direction**. Notice you now have switches under **Select State** to specify the output state of the different lines. Click “Close” to close the test panels.

11. Close MAX.

(End of Exercise)
Exercise 1 – Setting Up Your Device (Track C)

In this exercise, you will use Windows utilities to verify your sound card and prepare it for use with a microphone.

1. Prepare your microphone for use. Double-click the volume control icon on the task bar to open up the configuration window. The sound configuration window can also be found from the Windows Control Panel: Start Menu»Control Panel»Sounds and Audio Devices»Advanced.

2. If you do not see a microphone section, go to Options»Properties»Recording and place a checkmark in the box next to Microphone. This will display the Microphone volume control. Click “OK”.

3. Uncheck the Mute box if it is not already unchecked. Make sure that the volume is turned up.

4. Close the volume control configuration window.

5. Open the Sound Recorder by selecting Start»Programs»Accessories»Entertainment»Sound Recorder.

6. Click the record button and speak into your microphone. Notice how the sound signal is displayed in the Sound Recorder.

7. Click stop and close the Sound Recorder without saving changes when you are finished.

(End of Exercise)
LabVIEW

LabVIEW is a graphical programming language that uses icons instead of lines of text to create applications. In contrast to text-based programming languages, where instructions determine program execution, LabVIEW uses dataflow programming, where the flow of data determines execution order.

You can purchase several add-on software toolkits for developing specialized applications. All the toolkits integrate seamlessly in LabVIEW. Refer to the National Instruments Web site for more information about these toolkits.

LabVIEW also includes several wizards to help you quickly configure your DAQ devices and computer-based instruments and build applications.

LabVIEW Example Finder

LabVIEW includes hundreds of example VIs you can use and incorporate into VIs that you create. In addition to the example VIs that ship with LabVIEW, you also can access hundreds of example VIs on the NI Developer Zone (zone.ni.com). You can modify an example VI to fit an application, or you can copy and paste from one or more examples into a VI that you create.
LabVIEW Programs Are Called Virtual Instruments (VIs)

Each VI has 2 Windows

Front Panel
• User Interface (UI)
  – Controls = Inputs
  – Indicators = Outputs

Block Diagram
• Graphical Code
  – Data travels on wires from
    controls through functions to
    indicators
  – Blocks execute by Dataflow

LabVIEW programs are called virtual instruments (VIs).

Controls are inputs and indicators are outputs.

Each VI contains three main parts:
• Front Panel – How the user interacts with the VI.
• Block Diagram – The code that controls the program.
• Icon/Connector – Means of connecting a VI to other VIs.

In LabVIEW, you build a user interface by using a set of tools and objects. The user interface is known as the front panel. You then add code using graphical representations of functions to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart.

Users interact with the Front Panel when the program is running. Users can control the program, change inputs, and see data updated in real time. Controls are used for inputs such as, adjusting a slide control to set an alarm value, turning a switch on or off, or to stop a program. Indicators are used as outputs. Thermometers, lights, and other indicators display output values from the program. These may include data, program states, and other information.

Every front panel control or indicator has a corresponding terminal on the block diagram. When a VI is run, values from controls flow through the block diagram, where they are used in the functions on the diagram, and the results are passed into other functions or indicators through wires.
Use the Controls palette to place controls and indicators on the front panel. The Controls palette is available only on the front panel. To view the palette, select Window » Show Controls Palette. You also can display the Controls palette by right-clicking an open area on the front panel. Tack down the Controls palette by clicking the pushpin on the top left corner of the palette.
Use the **Functions** palette to build the block diagram. The **Functions** palette is available only on the block diagram. To view the palette, select **Window»Show Functions Palette**. You also can display the **Functions** palette by right-clicking an open area on the block diagram. Tack down the **Functions** palette by clicking the pushpin on the top left corner of the palette.
If automatic tool selection is enabled and you move the cursor over objects on the front panel or block diagram, LabVIEW automatically selects the corresponding tool from the Tools palette. Toggle automatic tool selection by clicking the Automatic Tool Selection button in the Tools palette.

Use the Operating tool to change the values of a control or select the text within a control.

Use the Positioning tool to select, move, or resize objects. The Positioning tool changes shape when it moves over a corner of a resizable object.

Use the Labeling tool to edit text and create free labels. The Labeling tool changes to a cursor when you create free labels.

Use the Wiring tool to wire objects together on the block diagram.

Other important tools:
• Click the **Run** button to run the VI. While the VI runs, the **Run** button appears with a black arrow if the VI is a top-level VI, meaning it has no callers and therefore is not a subVI.

• Click the **Continuous Run** button to run the VI until you abort or pause it. You also can click the button again to disable continuous running.

• While the VI runs, the **Abort Execution** button appears. Click this button to stop the VI immediately.

*Note:* Avoid using the **Abort Execution** button to stop a VI. Either let the VI complete its data flow or design a method to stop the VI programmatically. By doing so, the VI is at a known state. For example, place a button on the front panel that stops the VI when you click it.

• Click the **Pause** button to pause a running VI. When you click the **Pause** button, LabVIEW highlights on the block diagram the location where you paused execution. Click the **Pause** button again to continue running the VI.

• Select the **Text Settings** pull-down menu to change the font settings for the VI, including size, style, and color.

• Select the **Align Objects** pull-down menu to align objects along axes, including vertical, top edge, left, and so on.

• Select the **Distribute Objects** pull-down menu to space objects evenly, including gaps, compression, and so on.

• Select the **Resize Objects** pull-down menu to change the width and height of front panel objects.
• Select the **Reorder** pull-down menu when you have objects that overlap each other and you want to define which one is in front or back of another. Select one of the objects with the Positioning tool and then select from **Move Forward**, **Move Backward**, **Move To Front**, and **Move To Back**.

**Note:** The following items only appear on the block diagram toolbar.

• Click the **Highlight Execution** button to see the flow of data through the block diagram. Click the button again to disable execution highlighting.

• Click **Retain Wire Values** button to save the wire values at each point in the flow of execution so that when you place a probe on a wire, you can immediately obtain the most recent value of the data that passed through the wire.

• Click the **Step Into** button to single-step into a loop, subVI, and so on. Single-stepping through a VI steps through the VI node to node. Each node blinks to denote when it is ready to execute. By stepping into the node, you are ready to single-step inside the node.

• Click the **Step Over** button to step over a loop, subVI, and so on. By stepping over the node, you execute the node without single-stepping through the node.

• Click the **Step Out** button to step out of a loop, subVI, and so on. By stepping out of a node, you complete single-stepping through the node and go to the next node.

**Additional Tools:**

![Additional Buttons on the Diagram Toolbar](image)

- **Run Button**
- **Continuous Run Button**
- **Abort Execution**
- **Pause/Continue Button**
- **Text Settings**
- **Align Objects**
- **Distribute Objects**
- **Reorder**
- **Resize front panel objects**

- **Execution Highlighting Button**
  - **Step Into Button**
  - **Step Over Button**
  - **Step Out Button**
  - **Retain Wire Values**
When you create an object on the Front Panel, a terminal will be created on the Block Diagram. These terminals give you access to the Front Panel objects from the Block Diagram code.

Each terminal contains useful information about the Front Panel object it corresponds to. For example, the color and symbols provide information about the data type. For example: The dynamic data type is a polymorphic data type represented by dark blue terminals. Boolean terminals are green with TF lettering.

In general, blue terminals should wire to blue terminals, green to green, and so on. This is not a hard-and-fast rule; LabVIEW will allow a user to connect a blue terminal (dynamic data) to an orange terminal (fractional value), for example. But in most cases, look for a match in colors.

Controls have an arrow on the right side and have a thick border. Indicators have an arrow on the left and a thin border. Logic rules apply to wiring in LabVIEW: Each wire must have one (but only one) source (or control), and each wire may have multiple destinations (or indicators).
LabVIEW follows a dataflow model for running VIs. A block diagram node executes when all its inputs are available. When a node completes execution, it supplies data to its output terminals and passes the output data to the next node in the dataflow path. Visual Basic, C++, JAVA, and most other text-based programming languages follow a control flow model of program execution. In control flow, the sequential order of program elements determines the execution order of a program.

Consider the block diagram above. It adds two numbers and then multiplies by 2 from the result of the addition. In this case, the block diagram executes from left to right, not because the objects are placed in that order, but because one of the inputs of the Multiply function is not valid until the Add function has finished executing and passed the data to the Multiply function. Remember that a node executes only when data are available at all of its input terminals, and it supplies data to its output terminals only when it finishes execution. In the second piece of code, the Simulate Signal Express VI receives input from the controls and passes its result to the Graph.

You may consider the add-multiply and the simulate signal code to co-exist on the same block diagram in parallel. This means that they will both begin executing at the same time and run independent of one another. If the computer running this code had multiple processors, these two pieces of code could run independent of one another (each on its own processor) without any additional coding.
Debugging Techniques

• Finding Errors
  
  Click on broken Run button. Window showing error appears.

• Execution Highlighting
  
  Click on Execution Highlighting button; data flow is animated using bubbles. Values are displayed on wires.

• Probes
  
  Right-click on wire to display probe and it shows data as it flows through wire segment.
  
  You can also select Probe tool from Tools palette and click on wire.

When your VI is not executable, a broken arrow is displayed in the Run button in the palette.

• Finding Errors: To list errors, click on the broken arrow. To locate the bad object, click on the error message.

• Execution Highlighting: Animates the diagram and traces the flow of the data, allowing you to view intermediate values. Click on the light bulb on the toolbar.

• Probe: Used to view values in arrays and clusters. Click on wires with the Probe tool or right-click on the wire to set probes.

• Retain Wire Values: Used in conjunction with probes to view the values from the last iteration of the program.

• Breakpoint: Set pauses at different locations on the diagram. Click on wires or objects with the Breakpoint tool to set breakpoints.
Exercise 2 – Acquiring a Signal with DAQ (Track A)

Note: Before beginning this exercise, copy the Exercises and Solutions Folders to the desktop of your computer.

Complete the following steps to create a VI that acquires data continuously from your DAQ device.

1. Launch LabVIEW.

2. In the Getting Started window, click the New or VI from Template link to display the New dialog box.

3. Open a data acquisition template. From the Create New list, select VI»From Template»DAQ»Data Acquisition with NI-DAQmx.vi and click “OK”.

4. Display the block diagram by clicking it or by selecting Window»Show Block Diagram. Read the instructions written there about how to complete the program.

5. Double-click the DAQ Assistant to launch the configuration wizard.

6. Configure an analog input operation.
   a. Choose Analog Input»Voltage.
   b. Choose Dev1 (USB-6009)»ai0 to acquire data on analog input channel 0 and click “Finish.”
   c. In the next window you define parameters of your analog input operation.
      To choose an input range that works well with your microphone, on the settings tab enter 2 Volts for the maximum and 2 Volts for the minimum. On the task timing tab, choose “Continuous” for the acquisition mode and enter 10000 for the rate. Leave all other choices set to their default values. Click “OK” to exit the wizard.

7. Place the Filter Express VI to the right of the DAQ Assistant on the block diagram. From the functions palette, select Express»Signal Analysis»Filter and place it on the block diagram inside the while loop. When you bring up the functions palette, press the small push pin in the upper left hand corner of the palette. This will tack down the palette so that it doesn’t disappear. This step will be omitted in the following exercises, but should be repeated. In the configuration window under Filtering Type, choose “Highpass.” Under Cutoff Frequency, use a value of 300 Hz. Click “OK.”
8. Make the following connections on the block diagram by hovering your mouse over the terminal so that it becomes the wiring tool and clicking once on each of the terminals you wish to connect:
   a. Connect the “Data” output terminal of the DAQ Assistant VI to the “Signal” input of the Filter VI.
   b. Create a graph indicator for the filtered signal by right-clicking on the “Filtered Signal” output terminal and choose Create»Graph Indicator.
9. Return to the front panel by selecting Window»Show Front Panel or by pressing <Ctrl+E>.
10. Run your program by clicking the run button. Hum or whistle into the microphone to observe the changing voltage on the graph.
11. Click stop once you are finished.
12. Save the VI as “Exercise 2 – Acquire.vi” in your Exercises folder and close it.

Note: The solution to this exercise is printed in the back of this manual.

Tip: You can place the DAQ Assistant on your block diagram from the Functions Palette. Right-click the block diagram to open the Functions Palette and go to Express»Input to find it.

(End of Exercise)
Exercise 2 – Acquiring a Signal with DAQ (Track B)

Note: Before beginning this exercise, copy the Exercises and Solutions Folders to the desktop of your computer.

Complete the following steps to create a VI that acquires data continuously from your simulated DAQ device.

1. Launch LabVIEW.
2. In the Getting Started window, click the New or VI from Template link to display the New dialog box.
3. Open a data acquisition template. From the Create New list, select VI › From Template › DAQ › Data Acquisition with NI-DAQmx.vi and click “OK”.
4. Display the block diagram by clicking it or by selecting Window › Show Block Diagram. Read the instructions written there about how to complete the program.
5. Double-click the DAQ Assistant to launch the configuration wizard.
6. Configure an analog input operation.
   a. Choose Analog Input › Voltage.
   b. Choose Dev1 (PCI-6220) › ai0 to acquire data on analog input channel 0 and click “Finish.”
   c. In the next window you define parameters of your analog input operation.
   On the task timing tab, choose “Continuous” for the acquisition mode, enter 1000 for samples to read, and 10000 for the rate. Leave all other choices set to their default values. Click “OK” to exit the wizard.
7. On the block diagram, right-click the black arrow to the right of where it says “data.” Choose Create › Graph Indicator from the right-click menu.
8. Return to the front panel by selecting Window › Show Front Panel or by pressing <Ctrl+E>.
9. Run your program by clicking the run button. Observe the simulated sine wave on the graph.
10. Click stop once you are finished.
11. Save the VI as “Exercise 2 – Acquire.vi” in the Exercises folder. Close the VI.

Notes:
- The solution to this exercise is printed in the back of this manual.
- You can place the DAQ Assistant on your block diagram from the Functions Palette. Right-click the block diagram to open the Functions Palette and go to Express › Input to find it. When you bring up the functions palette, press the small push pin in the upper left hand corner of the palette. This will tack down the palette so that it doesn’t disappear. This step will be omitted in the following exercises, but should be repeated.
Click the Add Channels button to add more channels to the task.
Exercise 2 – Acquiring a Signal with the Sound Card (Track C)

**Note:** Before beginning this exercise, copy the Exercises and Solutions Folders to the desktop of your computer.

Complete the following steps to create a VI that acquires data from your sound card.

1. Launch LabVIEW.
2. In the *Getting Started* window, click the *Blank VI* link.
3. Display the block diagram by pressing <Ctrl+E> or selecting *Window*»*Show Block Diagram*.
4. Place the Acquire Sound Express VI on the block diagram. Right-click to open the functions palette and select *Express*»*Input*»*Acquire Sound*. Place the Express VI on the block diagram.
5. In the configuration window under *#Channels*, select 1 from the drop-down list and click “OK”.
6. Place the Filter Express VI to the right of the Acquire Signal VI on the block diagram. From the functions palette, select *Express*»*Signal Analysis*»*Filter* and place it on the block diagram. In the configuration window under *Filtering Type*, choose “Highpass.” Under Cutoff Frequency, use a value of 300 Hz. Click “OK.”
7. Make the following connections on the block diagram by hovering your mouse over the terminal so that it becomes the wiring tool and clicking once on each of the terminals you wish to connect:
   a. Connect the “Data” output terminal of the Acquire Signal VI to the “Signal” input of the Filter VI.
   b. Create a graph indicator for the filtered signal by right-clicking on the “Filtered Signal” output terminal and choose *Create*»*Graph Indicator*.
8. Return to the front panel by pressing <Ctrl+E> or *Window*»*Show Front Panel*.
9. Run your program by clicking the run button. Hum or whistle into your microphone and observe the data you acquire from your sound card.
10. Save the VI as “Exercise 2 – Acquire.vi” in the Exercises folder.
11. Close the VI.

**Note:** The solution to this exercise is printed in the back of this manual.

(End of Exercise)
The **Context Help window** displays basic information about LabVIEW objects when you move the cursor over each object. Objects with context help information include VIs, functions, constants, structures, palettes, properties, methods, events, and dialog box components.

To display the Context Help window, select Help»Show Context Help, press the <Ctrl+H> keys, or press the Show Context Help Window button in the toolbar.

Connections displayed in Context Help:

- **Required** – bold
- **Recommended** – normal
- **Optional** – dimmed

**Additional Help**

- VI, Function, & How-To Help is also available.
  - Help» VI, Function, & How-To Help
  - Right-click the VI icon and choose Help, or
  - Choose “**Detailed Help.**” on the context help window.

- LabVIEW Help – reference style help
  - Help»Search the LabVIEW Help…
Tips for Working in LabVIEW

• Keystroke Shortcuts
  – <Ctrl+H> – Activate/Deactivate Context Help Window
  – <Ctrl+B> – Remove Broken Wires From Block Diagram
  – <Ctrl+E> – Toggle Between Front Panel and Block Diagram
  – <Ctrl+Z> – Undo (Also in Edit Menu)

• Tools»Options… – Set Preferences in LabVIEW

• VI Properties – Configure VI Appearance, Documentation, etc.

LabVIEW has many keystroke shortcuts that make working easier. The most common shortcuts are listed above.

While the Automatic Selection Tool is great for choosing the tool you would like to use in LabVIEW, there are sometimes cases when you want manual control. Once the Automatic Selection Tool is turned off, use the Tab key to toggle between the four most common tools (Operate Value, Position/Size/Select, Edit Text, Set Color on Front Panel and Operate Value, Position/Size/Select, Edit Text, Connect Wire on Block Diagram). Once you are finished with the tool you choose, you can press <Shift+Tab> to turn the Automatic Selection Tool back on.

In the Tools»Options… dialog, there are many configurable options for customizing your Front Panel, Block Diagram, Colors, Printing, and much more.

Similar to the LabVIEW Options, you can configure VI specific properties by going to File»VI Properties… There you can document the VI, change the appearance of the window, and customize it in several other ways.
Section II – Elements of Typical Programs

A. Loops
   • While Loop
   • For Loop

B. Functions and SubVIs
   • Types of Functions
   • Creating Custom Functions (SubVI)
   • Functions Palette & Searching

C. Decision Making and File IO
   • Case Structure
   • Select (simple If statement)
   • File I/O
**Loops**

- **While Loops**
  - Terminal counts iteration
  - Always runs at least once
  - Runs until stop condition is met

- **For Loops**
  - Terminal counts iterations
  - Run according to input \( N \) of count terminal

Both the While and For Loops are located on the Functions >> Structures palette. The For Loop differs from the While Loop in that the For Loop executes a set number of times. A While Loop stops executing the subdiagram only if the value at the conditional terminal exists.

**While Loops**

Similar to a Do Loop or a Repeat-Until Loop in text-based programming languages, a While Loop, shown at the top right, executes a subdiagram until a condition is met. The While Loop executes the subdiagram until the conditional terminal, an input terminal, receives a specific Boolean value. The default behavior and appearance of the conditional terminal is **Stop If True**. When a conditional terminal is **Stop If True**, the While Loop executes its subdiagram until the conditional terminal receives a TRUE value. The iteration terminal (an output terminal), shown at left, contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns 0.

**For Loops**

A For Loop, shown above, executes a subdiagram a set number of times. The value in the count terminal (an input terminal) represented by the \( N \), indicates how many times to repeat the subdiagram. The iteration terminal (an output terminal), shown at left, contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns 0.
Place loops in your diagram by selecting them from the Structures palette of the Functions palette:

- **When selected, the mouse cursor becomes a special pointer that you use to enclose the section of code you want to repeat.**
- **Click the mouse button to define the top-left corner, click the mouse button again at the bottom-right corner, and the While Loop boundary is created around the selected code.**
- **Drag or drop additional nodes in the While Loop if needed.**
3 Types of Functions (from the Functions Palette)

Express VIs: interactive VIs with configurable dialog page (blue border)

Standard VIs: modularized VIs customized by wiring (customizable)

Functions: fundamental operating elements of LabVIEW; no front panel or block diagram (yellow)

LabVIEW 7.0 introduced a new type of subVI called Express VIs. These are interactive VIs that have a configuration dialog box that allows the user to customize the functionality of the Express VI. LabVIEW then generates a subVI based on these settings.

SubVIs are VIs (consisting of a front panel and a block diagram) that are used within another VI.

Functions are the building blocks of all VIs. Functions do not have a front panel or a block diagram.
What Types of Functions are Available?

• **Input and Output**
  – Signal and Data Simulation
  – Acquire and Generate Real Signals with DAQ
  – Instrument I/O Assistant (Serial & GPIB)
  – ActiveX for communication with other programs

• **Analysis**
  – Signal Processing
  – Statistics
  – Advanced Math and Formulas
  – Continuous Time Solver

• **Storage**
  – File I/O

LabVIEW includes several hundreds of pre-built functions that help you to acquire, analyze, and present data. You would generally use these functions as outlined on the slide above.

**LabVIEW Toolkits**

Additional toolkits are available for adding domain specific functionality to LabVIEW. These toolkits include:

<table>
<thead>
<tr>
<th>Application Deployment and Targeting Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>* LabVIEW PDA Module</td>
</tr>
<tr>
<td>* LabVIEW Real-Time Module</td>
</tr>
<tr>
<td>* LabVIEW FPGA Module</td>
</tr>
<tr>
<td>* LabVIEW Vision Development Module</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Embedded System Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>* DSP Test Integration Toolkit</td>
</tr>
<tr>
<td>* Embedded Test Integration Toolkit</td>
</tr>
<tr>
<td>* Digital Filter Design Toolkit</td>
</tr>
<tr>
<td>* LabVIEW FPGA Module</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal Processing and Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Sound and Vibration Toolkit</td>
</tr>
<tr>
<td>* Advanced Signal Processing Toolkit</td>
</tr>
<tr>
<td>* Modulation Toolkit</td>
</tr>
<tr>
<td>* Spectral Measurements Toolkit</td>
</tr>
<tr>
<td>* Order Analysis Toolkit</td>
</tr>
<tr>
<td>* Digital Filter Design Toolkit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control Design and Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Control Design and Simulation Bundle</td>
</tr>
<tr>
<td>* LabVIEW Real-Time Module</td>
</tr>
<tr>
<td>* System Identification Toolkit</td>
</tr>
<tr>
<td>* Control Design Toolkit</td>
</tr>
<tr>
<td>* LabVIEW Simulation Module</td>
</tr>
<tr>
<td>* State Diagram Toolkit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software Engineering and Optimization Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Execution Trace Toolkit for LabVIEW Real-Time</td>
</tr>
<tr>
<td>* Express VI Development Toolkit</td>
</tr>
<tr>
<td>* State Diagram Toolkit</td>
</tr>
<tr>
<td>* VI Analyzer Toolkit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image Processing and Acquisition</th>
</tr>
</thead>
<tbody>
<tr>
<td>* LabVIEW Vision Development Module</td>
</tr>
<tr>
<td>* NI Vision Builder for Automated Inspection</td>
</tr>
<tr>
<td>* NI-IMAQ for IEEE 1394</td>
</tr>
</tbody>
</table>

http://www.ni.com/toolkits/
Searching for Controls, VIs, and Functions

• Palettes are filled with hundreds of VIs
• Press the search button to index the all VIs for text searching
• Click and drag an item from the search window to the block diagram
• Double-click an item to open the owning palette

Use the buttons on top of the palette windows to navigate, search, and edit the palettes.
You can search for controls, VIs, and functions that either contain certain words or start with certain words. Double clicking a search result opens the palette that contains the search result. You also can click and drag the name of the control, VI, or function directly to the front panel or block diagram.
Creating SubVIs

After you build a VI, you can use it in another VI. A VI called from the block diagram of another VI is called a subVI. You can reuse a subVI in other VIs. To create a subVI, you need to build a connector pane and create an icon.

A subVI node corresponds to a subroutine call in text-based programming languages. A block diagram that contains several identical subVI nodes calls the same subVI several times.

The subVI controls and indicators receive data from and return data to the block diagram of the calling VI. Click the Select a VI icon or text on the Functions palette, navigate to and double-click a VI, and place the VI on a block diagram to create a subVI call to that VI.

A subVI input and output terminals and the icon can be easily customized. Follow the instructions below to quickly create a subVI.

Creating SubVIs from Sections of a VI

Convert a section of a VI into a subVI by using the Positioning tool to select the section of the block diagram you want to reuse and selecting Edit»Create SubVI. An icon for the new subVI replaces the selected section of the block diagram. LabVIEW creates controls and indicators for the new subVI, automatically configures the connector pane based on the number of control and indicator terminals you selected, and wires the subVI to the existing wires.

See Help»Search the LabVIEW Help…»SubVIs for more information.
A subVI node corresponds to a subroutine call in text-based programming languages. The node is not the subVI itself, just as a subroutine call statement in a program is not the subroutine itself. A block diagram that contains several identical subVI nodes calls the same subVI several times.

The modular approach makes applications easier to debug and maintain.

The functionality of the subVI does not matter for this example. The important point is the passing of two numeric inputs and one numeric output.
Exercise 3.1 – Analysis (Track A, B, & C)

Create a VI that produces a sine wave with a specified frequency and displays the data on a Waveform Chart until stopped by the user.

1. Open a blank VI from the Getting Started screen.
2. Place a chart on the front panel. Right-click to open the controls palette and select Controls » Modern » Graph » Waveform Chart.
3. Place a dial control on the front panel. From the controls palette, select Controls » Modern » Numeric » Dial. Notice that when you first place the control on the front panel, the label text is highlighted. While this text is highlighted, type “Frequency In” to give a name to this control.
4. Go to the block diagram (<Ctrl+E>) and place a while loop down. Right-click to open the functions palette and select Express » Execution Control » While Loop. Click and drag on the block diagram to make the while loop the correct size. Select the waveform chart and dial and drag them inside the while loop if they are not already. Notice that a stop button is already connected to the conditional terminal of the while loop.
5. Place the Simulate Signal Express VI on the block diagram. From the functions palette, select Express » Signal Analysis » Simulate Signal and place it on the block diagram inside the while loop. In the configuration window under Timing, choose “Simulate acquisition timing.” Click “OK.”
6. Place a Tone Measurements Express VI on the block diagram (Express » Signal Analysis » Tone Measurements). In the configuration window, choose Amplitude and Frequency measurements in the Single Tone Measurements section. Click “OK.”
7. Make the following connections on the block diagram by hovering your mouse over the terminal so that it becomes the wiring tool and clicking once on each of the terminals you wish to connect:

a. Connect the “Sine” output terminal of the Simulate Signal VI to the “Signals” input of the Tone Measurements VI.

b. Connect the “Sine” output to the Waveform Chart.

c. Create indicators for the amplitude and frequency measurements by right-clicking on each of the terminals of the Tone Measurements Express VI and selecting Create Numeric Indicator.

d. Connect the “Frequency In” control to the “Frequency” terminal of the Simulate Signal VI.

8. Return to the front panel and run the VI. Move the “Frequency In” dial and observe the frequency of the signal. Click the stop button once you are finished.

9. Save the VI as “Exercise 3.1 – Simulated.vi”.

10. Close the VI.

Notes

- When you bring up the functions palette, press the small push pin in the upper left hand corner of the palette. This will tack down the palette so that it doesn’t disappear. This step will be omitted in the following exercises, but should be repeated.

- The solution to this exercise is printed in the back of this manual.

(End of Exercise)
Exercise 3.2 – Analysis (Track A & B)

Create a VI that measures the frequency and amplitude of the signal from your (simulated) DAQ device and displays the acquired signal on a waveform chart. The instructions are the same as in Exercise 3.1, but a DAQ Assistant is used in place of the Simulate Signal VI. Try to do this without following the instructions!

1. Open a blank VI.
2. Place a chart on the front panel. Right click to open the controls palette and select Controls»Modern»Graph»Waveform Chart.
3. Go to the block diagram and place a while loop down (Express»Execution Control»While Loop).
4. Place a DAQ Assistant on the block diagram (Express»Input»DAQ Assistant). Choose analog input on channel ai0 of your (simulated) device and click “Finish.” On the task timing tab, choose “continuous” for the acquisition mode. If you are using the USB-6009, change the Input Range to -2 to 2 and the number of Samples to Read to 100.
5. Place the Filter Express VI to the right of the DAQ Assistant on the block diagram. From the functions palette, select Express»Signal Analysis»Filter and place it on the block diagram inside the while loop. In the configuration window under Filtering Type, choose “Highpass.” Under Cutoff Frequency, use a value of 300 Hz. Click “OK.”
6. Connect the “Data” output terminal of the DAQ Assistant VI to the “Signal” input of the Filter VI.
7. Connect the “Filtered Signal” terminal on the Filter VI to the Waveform Chart.
8. Place a Tone Measurements Express VI on the block diagram (Express»Signal Analysis»Tone). In the configuration window, choose Amplitude and Frequency measurements in the Single Tone Measurements section.
9. Create indicators for the amplitude and frequency measurements by right clicking on each of the terminals of the Tone Measurements Express VI and selecting Create»Numeric Indicator.
10. Connect the output of the Filter VI to the “Signals” input of the Tone Measurements Express VI.
11. Return to the front panel and run the VI. Observe your acquired signal and its frequency and amplitude. Hum or whistle into the microphone if you have a USB-6009 and observe the amplitude and frequency that you are producing.
12. Save the VI as “Exercise 3.2 - Data.vi”.
13. Close the VI.

Note: The solution to this exercise is printed in the back of this manual.
Exercise 3.2 – Analysis (Track C)

Create a VI that measures the frequency and amplitude of the signal from your sound card and displays the acquired signal on a waveform chart. The instructions are the same as in Exercise 3.1, but the Sound Signal VI is used in place of the Simulate Signal VI. Try to do this without following the instructions!

1. Open a blank VI.
2. Go to the block diagram and place a While Loop down (Express»Execution Control»While Loop).
3. Place the Acquire Sound Express VI on the block diagram (Express»Input»Acquire Sound).
4. Place a Filter Express VI on the block diagram. In the configuration window choose a highpass filter and a cutoff frequency of 300 Hz.
5. Place a Tone Measurements Express VI on the block diagram (Express»Signal Analysis»Tone). In the configuration window, choose Amplitude and Frequency measurements in the Single Tone Measurements section.
6. Create indicators for the amplitude and frequency measurements by right-clicking on each of the terminals of the Tone Measurements Express VI and selecting Create»Numeric Indicator.
7. Connect the “Data” terminal of the Acquire Sound Express VI to the “Signal” input of the Filter VI.
8. Connect the “Filtered Signal” terminal of the Filter VI to the “Signals” input of the Tone Measurements VI.
9. Create a graph indicator for the Filtered Signal by right-clicking on the “Filtered Signal” terminal and selecting Create»Graph Indicator.
10. Return to the front panel and run the VI. Observe the signal from your sound card and its amplitude and frequency. Hum or whistle into the microphone and observe the amplitude and frequency you are producing.
11. Save the VI as “Exercise 3.2-Data.vi”. Close the VI.

Note: The solution to this exercise is printed in the back of this manual.

(End of Exercise)