Getting Started with Instrument Control Using LabVIEW

Introduction

As a user of test and measurement instrumentation, you rely on your hardware to achieve the best possible measurement quality. After unpacking and installing your equipment, you walk through the steps provided in a quick start guide to help provide no less than the cursory training which provides you the hands-on knowledge needed to establish your test scenario setup and execution. At some point you will find that your basic setup evolves into an established system which performs a variety of routine tasks. Instead of relying on rote memory of procedures, you will likely look towards options to automate the instrument control, test execution and evaluation, and on to data archival and analysis – all of this to be handled by software.

This document will offer a cursory introduction to getting started with using LabVIEW graphical software on a Windows PC to perform test automation with practically any piece of instrumentation in your desired system or lab inventory. For simplicity, we will use a Bird 4421A Power Meter Display. We will first offer some insight into the benefits of LabVIEW and why you might prefer this software option over others. Then we will provide general details on acquiring and license-based usage of the software. We will then cover installation of the recommended software options. Finally, we share some of the basic building blocks for connecting to and controlling your instruments using an examples-based approach. Finally, we share some notes on instrument drivers.

Reasons for Using LabVIEW

LabVIEWi is a popular and powerful software platform developed by National Instruments for test, measurement, and automation applications. There are several reasons why LabVIEW is often chosen for test automation, but first and foremost is that its designers took great care in ensuring users would have an easy way to create graphical user interfaces (GUIs) that could be tailored to each specific application's needs. The graphical nature of LabVIEW allows users to visually design the flow of their programs. LabVIEW provides a range of data visualization tools, including charts, graphs, and customizable user interfaces, making it easier to interpret and display test results. This is especially important when presenting data to non-technical stakeholders.

LabVIEW (LV) promotes a modular approach to programming. Engineers can create reusable components called Virtual Instruments (VIs) that can be easily integrated into larger systems. This modularity simplifies test system maintenance and enhances code reusability. Furthermore, LabVIEW offers a vast library of pre-built functions and toolkits for tasks such as signal processing, control, and analysis. These libraries can save time and effort by providing ready-made solutions to common testing and measurement challenges.

LV can interface with other programming languages and software tools, allowing for integration with enterprise systems, databases, and communication with other applications. This is crucial for connecting test automation systems with wider business processes. Moreover, its cross-platform compatibility ensures it is available for Windows and macOS, and it supports cross-platform development, enabling engineers to create applications that run on different operating systems.

Looking to become a LabVIEW expert? National Instruments (NI) offers certification programs and extensive training options to help engineers and technicians become proficient in LabVIEW, ensuring that they can effectively implement test automation solutions.

Preparing Your System for LabVIEW Usage

For those who are just getting started and not looking to use the software as a full-time commitment, consider using the LabVIEW Community Edition. This free-to-use version provides you with all the capabilities found in the LabVIEW Pro editions, including, the LINX toolkit for use with Raspberry Pi, BeagleBoard, and Arduino along with access to the web development software (named G) for creating web-based applications when you need them. Bear in mind that all of this is offered to you in the LV Community Edition so long as you adhere to at least the following:

Only for personal, non-commercial use. You can build your own knowledge base, but it is expected that you are not developing code *for profit*.

You cannot use the software "for teaching or research at a degree-granting educational institution". NI may have discount offers specifically for colleges and universities, so it's best to speak with a sales specialist about this.

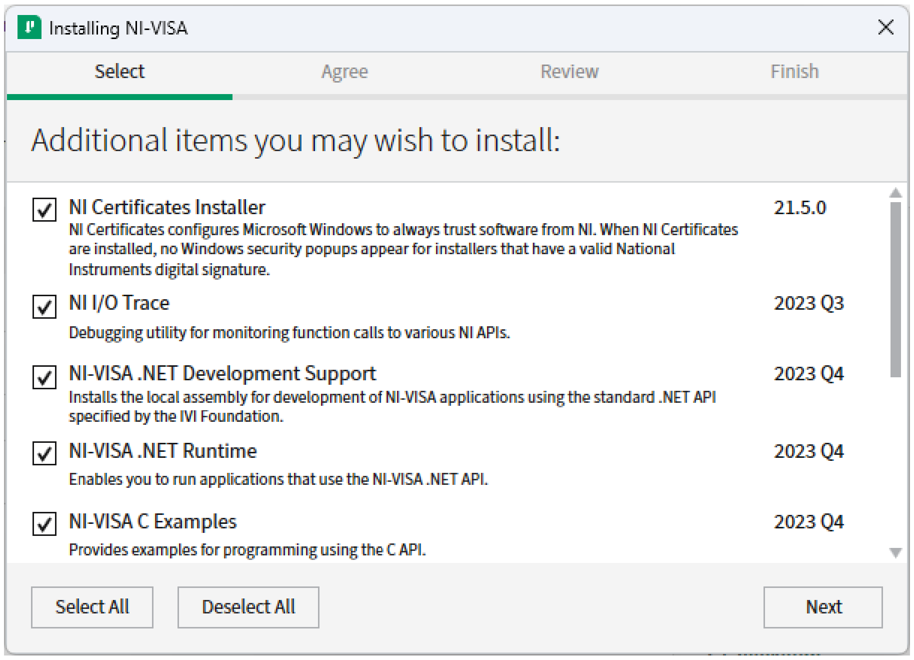
For more detail, see the NI Community Edition Software Usage Details.

First Install NI-VISA

Before you download and install LabVIEW, it is highly recommended that you first download and install NI's version of the **Virtual Instrument Software Architecture**, or **NI-VISA**. This will provide you with a great foundation for extensive instrument communication as well as some debugging tools. You can download NI-VISA from this landing page.

Note that if you do not already have a registered user account then you will need to create one.

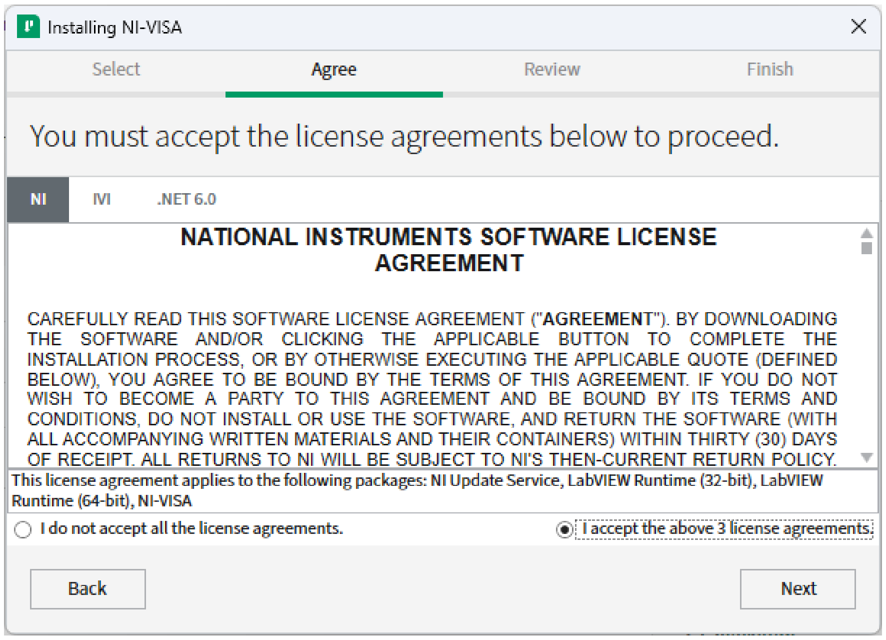
When the installer launches, it will offer you a few additional items. Go ahead and accept these by clicking **Next**.



Additional items you may wish to install:

✓	NI Certificates Installer NI Certificates configures Microsoft Windows to always trust software from NI. When NI Certificates are installed, no Windows security popups appear for installers that have a valid National Instruments digital signature.	21.5.0
✓	NI I/O Trace Debugging utility for monitoring function calls to various NI APIs.	2023 Q3
✓	NI-VISA .NET Development Support Installs the local assembly for development of NI-VISA applications using the standard .NET API specified by the IVI Foundation.	2023 Q4
✓	NI-VISA .NET Runtime Enables you to run applications that use the NI-VISA .NET API.	2023 Q4
✓	NI-VISA C Examples Provides examples for programming using the C API.	2023 Q4

You will need to accept any/all license agreements in order to proceed. Read carefully. If there's anything you do not agree with, then discontinue installation, delete the downloaded installer (likely housed in your Downloads folder), and discard this document. Otherwise, click **Next**.


Installing NI-VISA

Select | Agree | Review | Finish

You must accept the license agreements below to proceed.

NI | IVI | .NET 6.0

NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

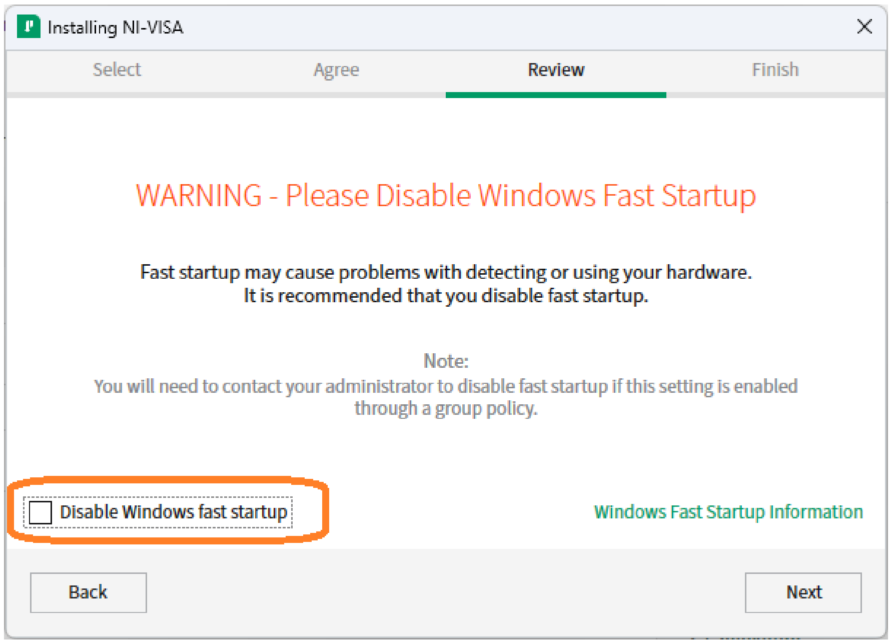
CAREFULLY READ THIS SOFTWARE LICENSE AGREEMENT ("**AGREEMENT**"). BY DOWNLOADING THE SOFTWARE AND/OR CLICKING THE APPLICABLE BUTTON TO COMPLETE THE INSTALLATION PROCESS, OR BY OTHERWISE EXECUTING THE APPLICABLE QUOTE (DEFINED BELOW), YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT WISH TO BECOME A PARTY TO THIS AGREEMENT AND BE BOUND BY ITS TERMS AND CONDITIONS, DO NOT INSTALL OR USE THE SOFTWARE, AND RETURN THE SOFTWARE (WITH ALL ACCOMPANYING WRITTEN MATERIALS AND THEIR CONTAINERS) WITHIN THIRTY (30) DAYS OF RECEIPT. ALL RETURNS TO NI WILL BE SUBJECT TO NI'S THEN-CURRENT RETURN POLICY.

This license agreement applies to the following packages: NI Update Service, LabVIEW Runtime (32-bit), LabVIEW Runtime (64-bit), NI-VISA

○ I do not accept all the license agreements. ● I accept the above 3 license agreements.

Back | Next

You are then given the option to disable Windows Fast Startup. While it is recommended by NI, if you are installing on a computer controlled by an IT department, you may not have the option to do this. If in doubt, uncheck the option and click **Next** to continue.


Installing NI-VISA

Select | Agree | Review | Finish

WARNING - Please Disable Windows Fast Startup

Fast startup may cause problems with detecting or using your hardware.
It is recommended that you disable fast startup.

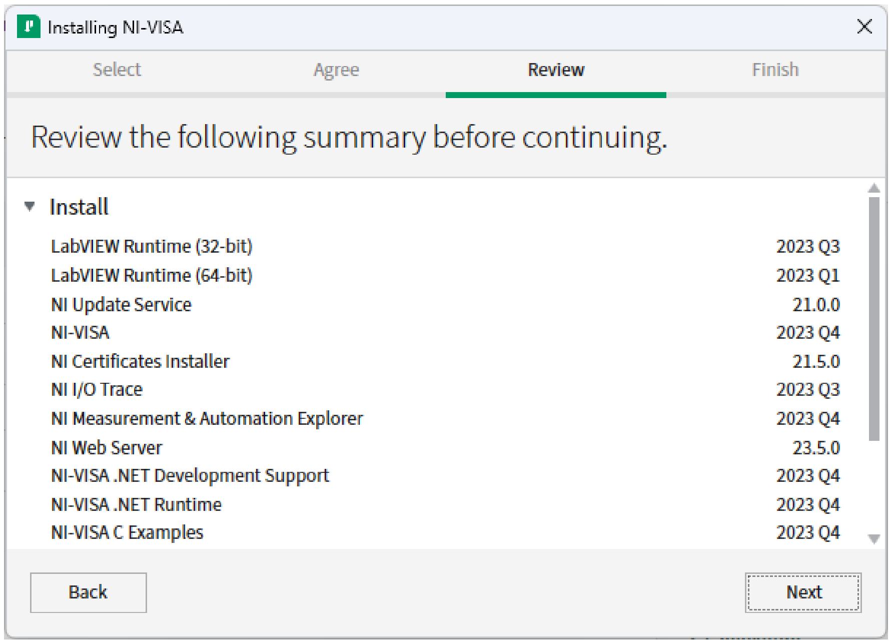
Note:
You will need to contact your administrator to disable fast startup if this setting is enabled through a group policy.

☐ Disable Windows fast startup

Windows Fast Startup Information

Back | Next

You will then be given the opportunity to review all the software options you have elected to install. Click **Next** to continue.

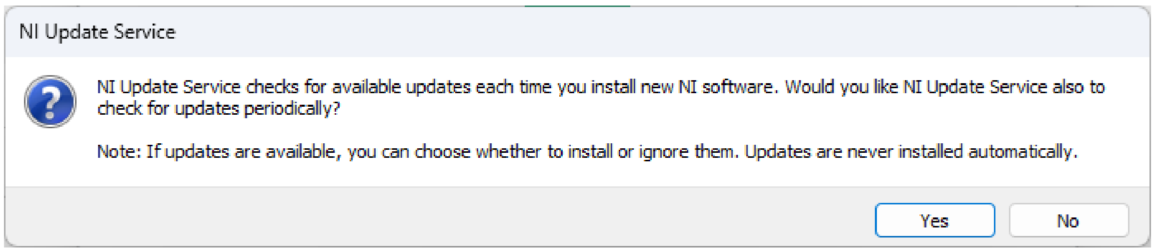

Installing NI-VISA

Review the following summary before continuing.

Install

LabVIEW Runtime (32-bit)	2023 Q3
LabVIEW Runtime (64-bit)	2023 Q1
NI Update Service	21.0.0
NI-VISA	2023 Q4
NI Certificates Installer	21.5.0
NI I/O Trace	2023 Q3
NI Measurement & Automation Explorer	2023 Q4
NI Web Server	23.5.0
NI-VISA .NET Development Support	2023 Q4
NI-VISA .NET Runtime	2023 Q4
NI-VISA C Examples	2023 Q4

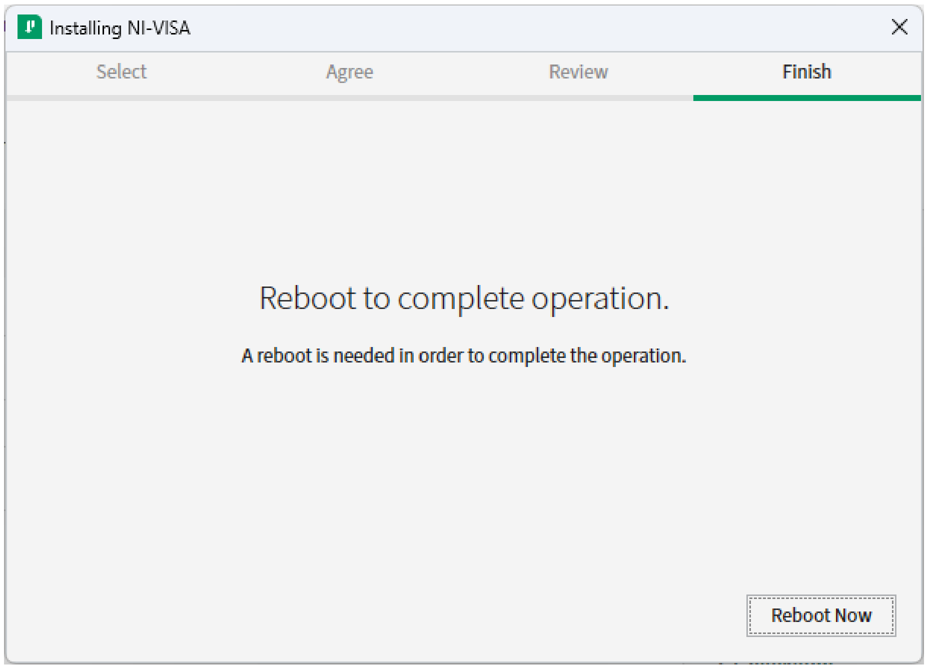
Near the end of the installation, you may be prompted to enable the NI Update Service which will run in the background after you power on your computer and periodically check for updates. You can select whether to apply updates automatically or be prompted to choose whether or not to install. If you are comfortable with the automatic update checking, click **Yes**, otherwise click **No**.


NI Update Service

NI Update Service checks for available updates each time you install new NI software. Would you like NI Update Service also to check for updates periodically?

Note: If updates are available, you can choose whether to install or ignore them. Updates are never installed automatically.

Finally, after successful installation of the VISA software you will be prompted to reboot your system. Click **Reboot Now**.



Reboot to complete operation.

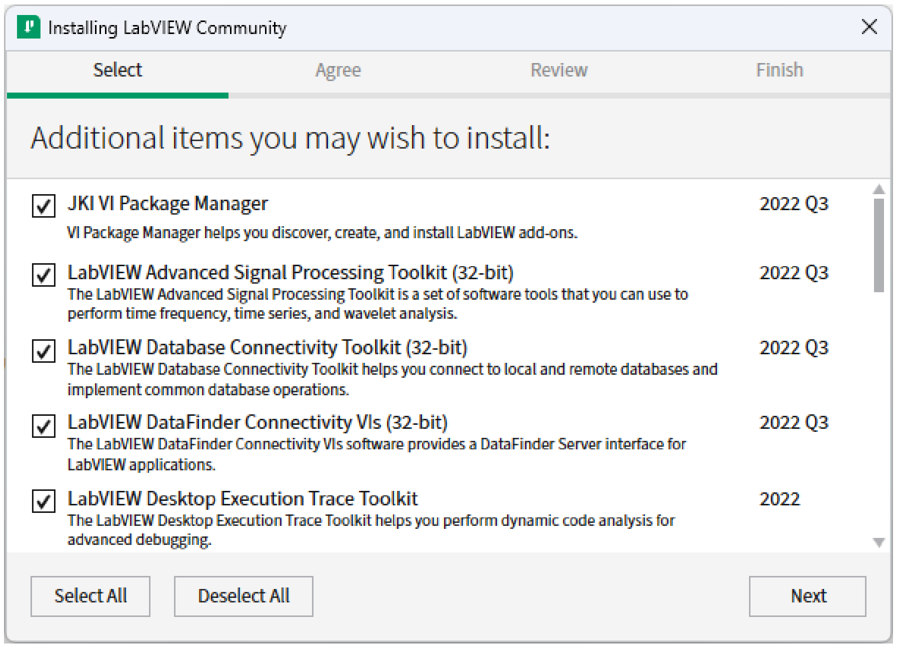
A reboot is needed in order to complete the operation.

Then install LabVIEW Community Edition

You can download the LV Community software from the NI website on this landing page.

The download consists of a large *.iso image – about 2.77 GB for the 2022 Q3 Community version. Double-clicking on the install option (found within the *.iso) will create a temporary drive from which the installer runs. *Fair warning: the install will require some patience on your part.*

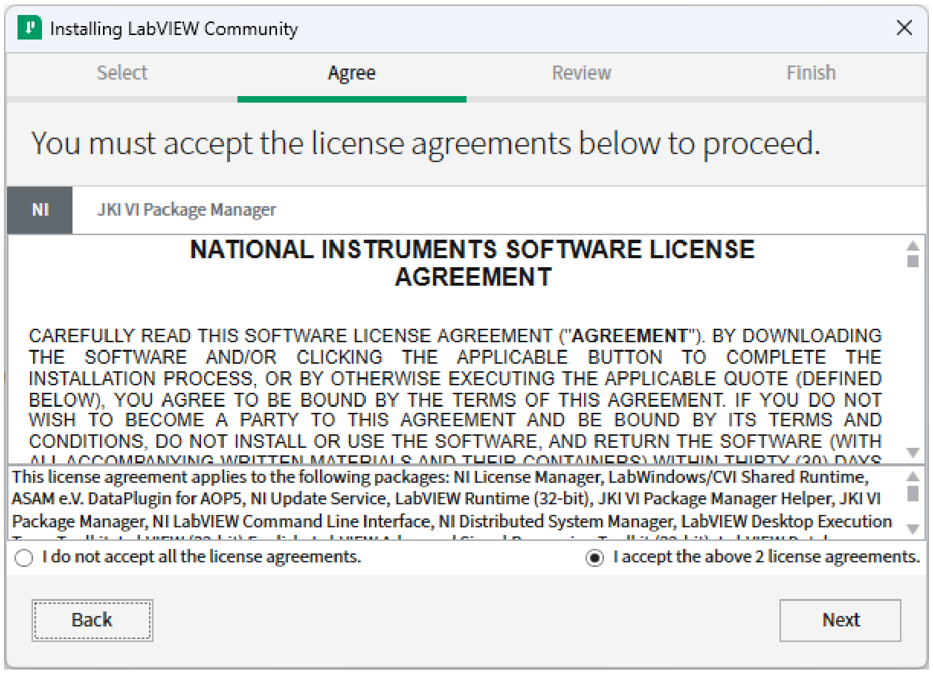
When the installer launches, it will offer you a few additional items. Go ahead and accept these by clicking **Next**.



Additional items you may wish to install:

✓	JKI VI Package Manager VI Package Manager helps you discover, create, and install LabVIEW add-ons.	2022 Q3
✓	LabVIEW Advanced Signal Processing Toolkit (32-bit) The LabVIEW Advanced Signal Processing Toolkit is a set of software tools that you can use to perform time frequency, time series, and wavelet analysis.	2022 Q3
✓	LabVIEW Database Connectivity Toolkit (32-bit) The LabVIEW Database Connectivity Toolkit helps you connect to local and remote databases and implement common database operations.	2022 Q3
✓	LabVIEW DataFinder Connectivity VIs (32-bit) The LabVIEW DataFinder Connectivity VIs software provides a DataFinder Server interface for LabVIEW applications.	2022 Q3
✓	LabVIEW Desktop Execution Trace Toolkit The LabVIEW Desktop Execution Trace Toolkit helps you perform dynamic code analysis for advanced debugging.	2022

You will need to accept any/all license agreements in order to proceed. Read carefully. If there's anything you do not agree with, then discontinue installation, delete the downloaded installer (likely housed in your Downloads folder), and discard this document. Otherwise, click **Next**.



You must accept the license agreements below to proceed.

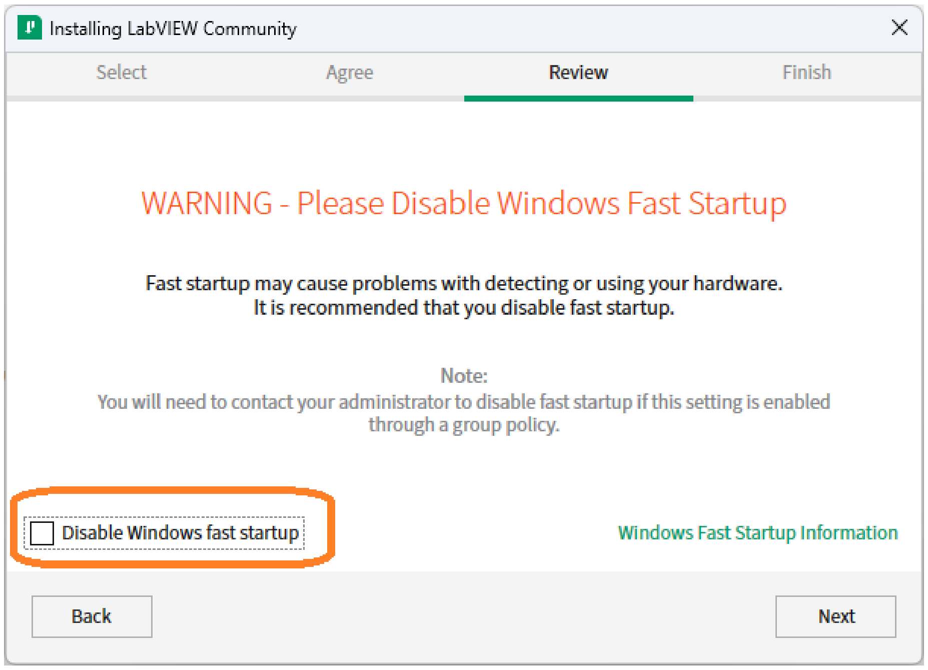
NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

CAREFULLY READ THIS SOFTWARE LICENSE AGREEMENT ("**AGREEMENT**"). BY DOWNLOADING THE SOFTWARE AND/OR CLICKING THE APPLICABLE BUTTON TO COMPLETE THE INSTALLATION PROCESS, OR BY OTHERWISE EXECUTING THE APPLICABLE QUOTE (DEFINED BELOW), YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT WISH TO BECOME A PARTY TO THIS AGREEMENT AND BE BOUND BY ITS TERMS AND CONDITIONS, DO NOT INSTALL OR USE THE SOFTWARE, AND RETURN THE SOFTWARE (WITH

This license agreement applies to the following packages: NI License Manager, LabWindows/CVI Shared Runtime, ASAM e.V. DataPlugin for AOP5, NI Update Service, LabVIEW Runtime (32-bit), JKI VI Package Manager Helper, JKI VI Package Manager, NI LabVIEW Command Line Interface, NI Distributed System Manager, LabVIEW Desktop Execution

I do not accept all the license agreements. [✓ I accept the above 2 license agreements.]

Again, you are then given the option to disable Windows Fast Startup. While it is recommended by NI, if you are installing on a computer controlled by an IT department, you may not have the option to do this. If in doubt, uncheck the option and click **Next** to continue.


Installing LabVIEW Community
Select
Agree
Review
Finish
Windows Fast Startup Information
Back
Next

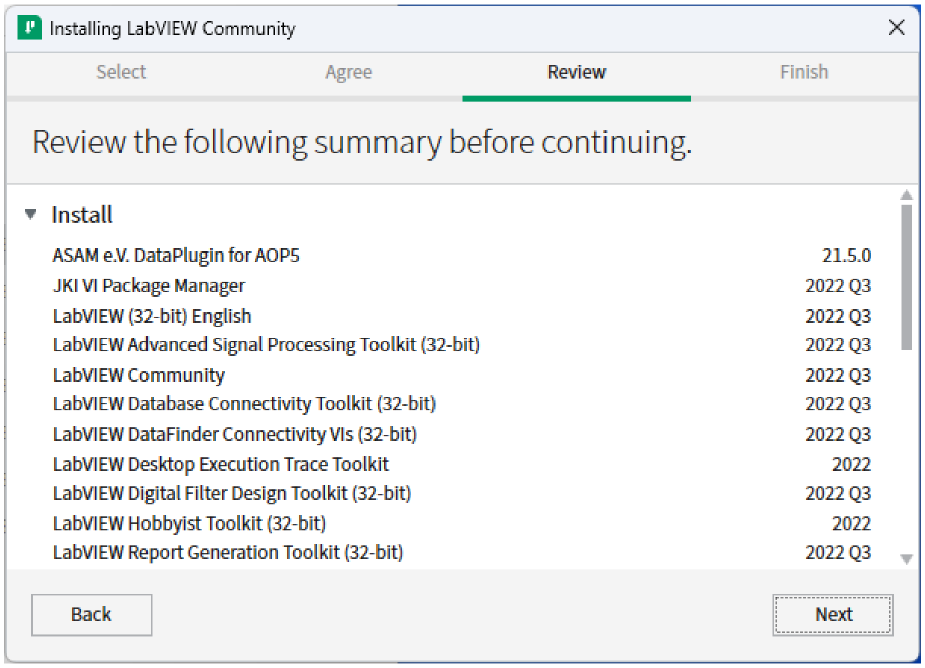
WARNING - Please Disable Windows Fast Startup

Fast startup may cause problems with detecting or using your hardware. It is recommended that you disable fast startup.

Note:
You will need to contact your administrator to disable fast startup if this setting is enabled through a group policy.

☐ Disable Windows fast startup

You will then be given the opportunity to review all the software options you have elected to install. Click **Next** to continue.

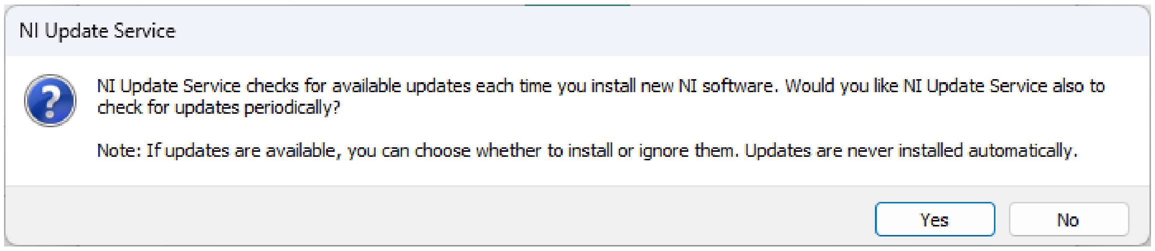


Review the following summary before continuing.

Install

ASAM e.V. DataPlugin for AOP5	21.5.0
JKI VI Package Manager	2022 Q3
LabVIEW (32-bit) English	2022 Q3
LabVIEW Advanced Signal Processing Toolkit (32-bit)	2022 Q3
LabVIEW Community	2022 Q3
LabVIEW Database Connectivity Toolkit (32-bit)	2022 Q3
LabVIEW DataFinder Connectivity VIs (32-bit)	2022 Q3
LabVIEW Desktop Execution Trace Toolkit	2022
LabVIEW Digital Filter Design Toolkit (32-bit)	2022 Q3
LabVIEW Hobbyist Toolkit (32-bit)	2022
LabVIEW Report Generation Toolkit (32-bit)	2022 Q3

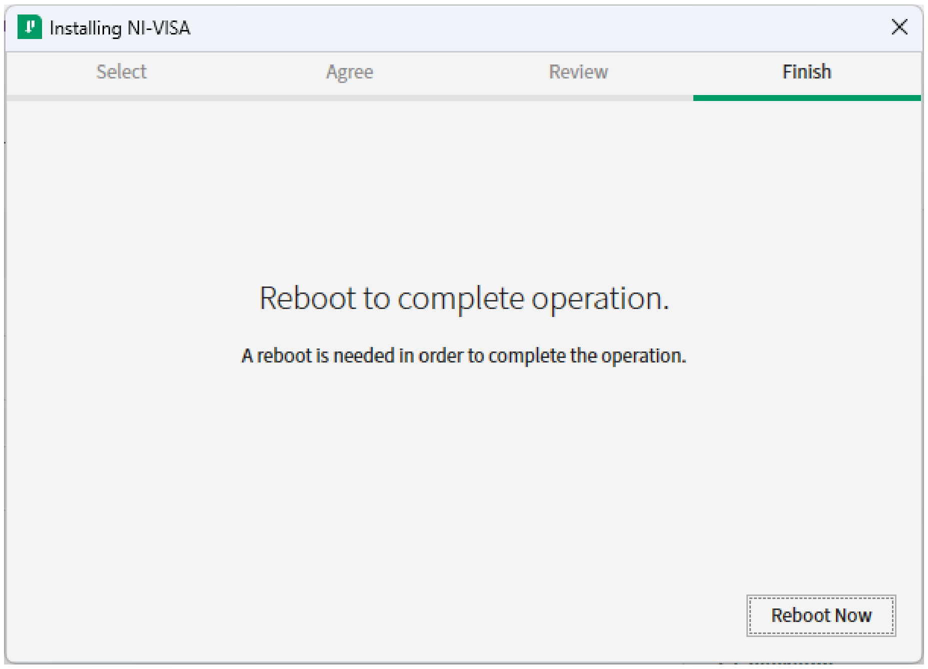
Near the end of the installation, you may be prompted to enable the NI Update Service which will run in the background after you power on your computer and periodically check for updates. You can select whether to apply updates automatically or be prompted to choose whether or not to install. If you are comfortable with the automatic update checking, click **Yes**, otherwise click **No**.



NI Update Service checks for available updates each time you install new NI software. Would you like NI Update Service also to check for updates periodically?

Note: If updates are available, you can choose whether to install or ignore them. Updates are never installed automatically.

Finally, after successful installation of the VISA software you will be prompted to reboot your system. Click **Reboot Now**.

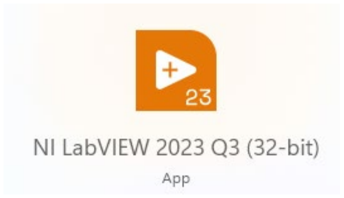


Reboot to complete operation.

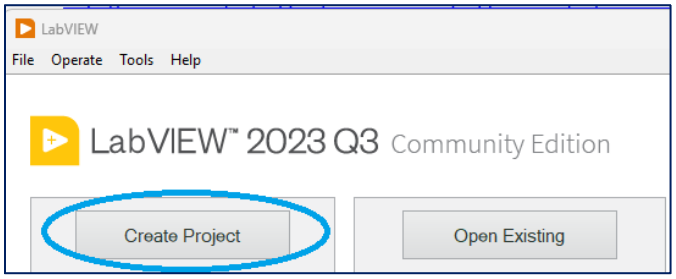
A reboot is needed in order to complete the operation.

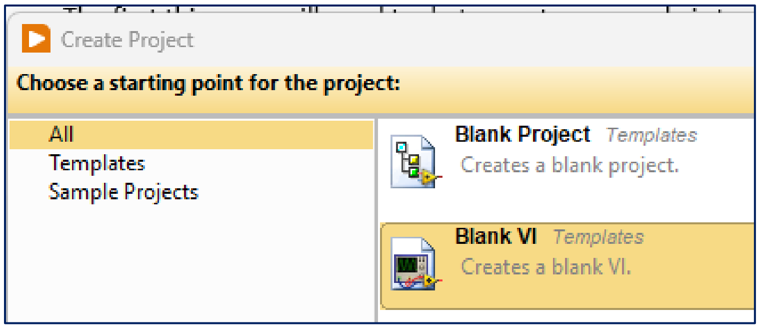
Create Code with Open, Write, Read, and Close Instrument Operations

The first thing you will need to do to create your code is to start LabVIEW which you can do from the Windows Start menu, locating and selecting the NI LabVIEW application from the list of options.



After you are presented with the LV starting dialog, choose **Create Project** from the two buttons provided, then select the **Blank VI** option.





Create Project

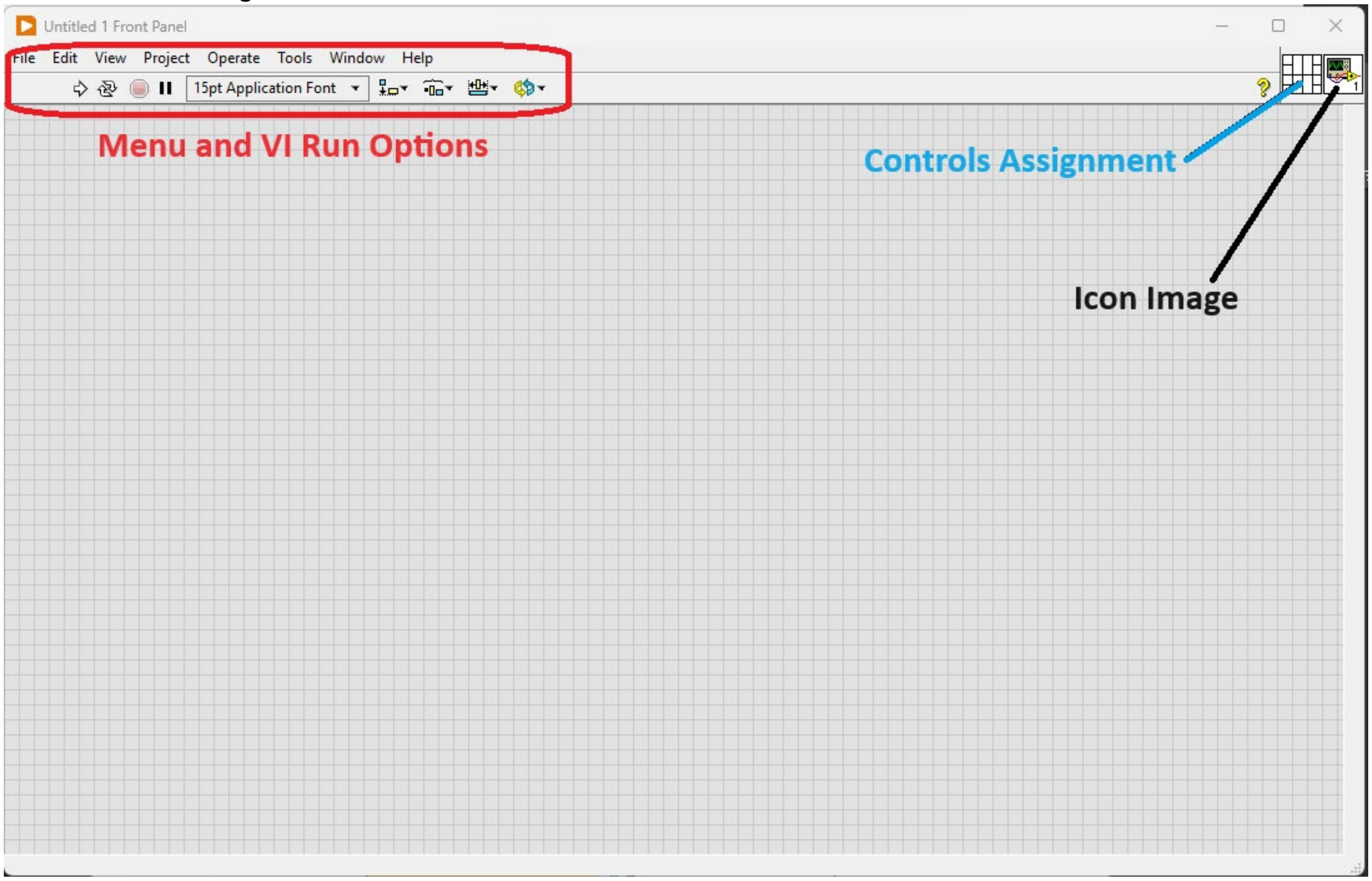
Choose a starting point for the project:

Blank Project *Templates* Creates a blank project.

Blank VI *Templates* Creates a blank VI.

You will be presented with two new and separate windows: a front panel and a block diagram. The front panel acts as your graphical canvas as you construct your code, hosting things like controls, indicators, icons, and text that help to establish the main interface that an operator will interact with. You are given a blank slate to work with until you start adding elements to your VI. The primary areas to be concerned with for this exercise are:

- The menu and run options
- The controls assignment widget (or pattern assignment tool)
- The icon image

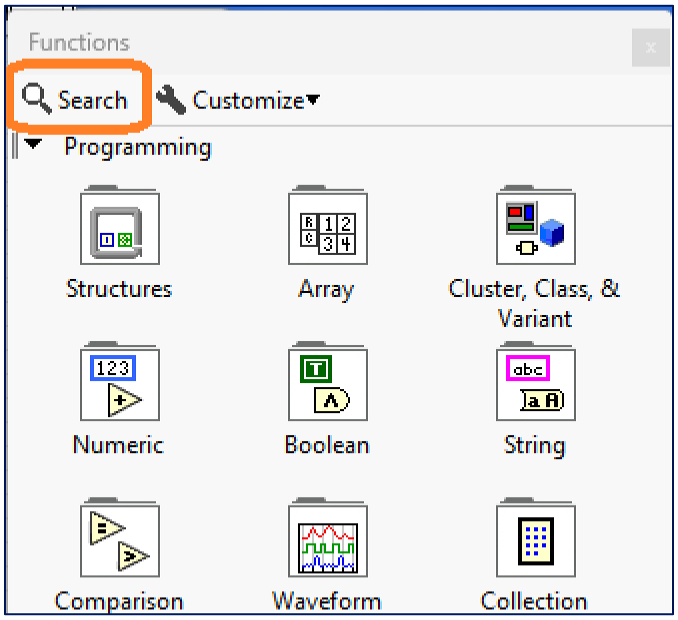


While this view shows a lot of promise on the numerous ways we might choose to interact with the end user, the logic to make it all work happens within the block diagram. To switch between these two associated VI views, you have two choices:

- Use the hotkey combination of **Ctrl+E**, or
- Use your mouse to navigate between them.

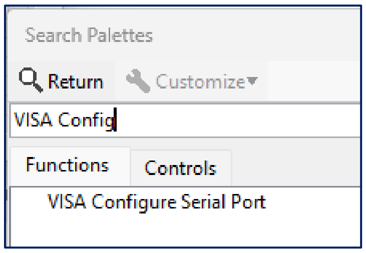
Switch to block diagram view.

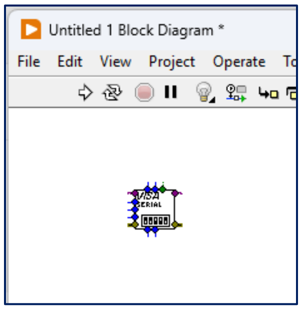
While working in the block diagram, a necessary tool to become familiar with is the Functions Pallet with its Search capability which should be presented by default. If it is hidden, you can enable it through the **View->Functions Pallet** selection.



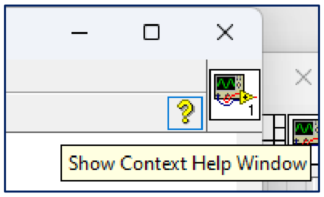
Your most reliable means for making connections to instrumentation will be using the virtual instrumentation software architecture (VISA) toolset. Note that NI-VISA[ii] is a software add-in which comes as part of the LabVIEW installation so it will already be part of your system.

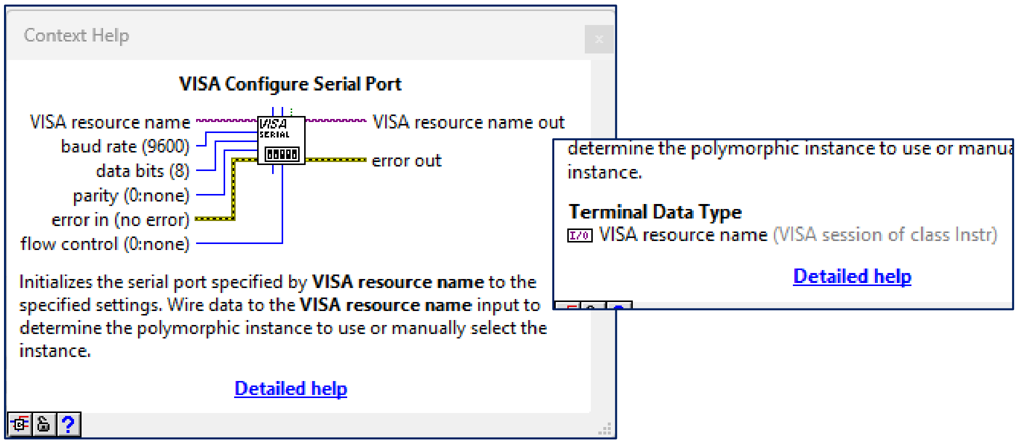
Use the search feature to find “VISA Configure Serial Port” then drag and drop the text for this result into your block diagram. This exposes the option as its own VI, similar to a function or module in text-based programming languages. Hovering over top of the VI with your mouse cursor will expose the different colored nodes available for specifying necessary inputs and outputs to the VI.





To help understand the function of each of these different nodes, you can turn on **Context Help** by clicking on the yellow question mark icon in the top right corner of the block diagram interface. This will expose a small, generic dialog. However, to see its full potential, you can hover your mouse cursor over the "VISA Configure Serial Port" VI again.





VISA Configure Serial Port

Initializes the serial port specified by **VISA resource name** to the specified settings. Wire data to the **VISA resource name** input to determine the polymorphic instance to use or manually select the instance.

Detailed help

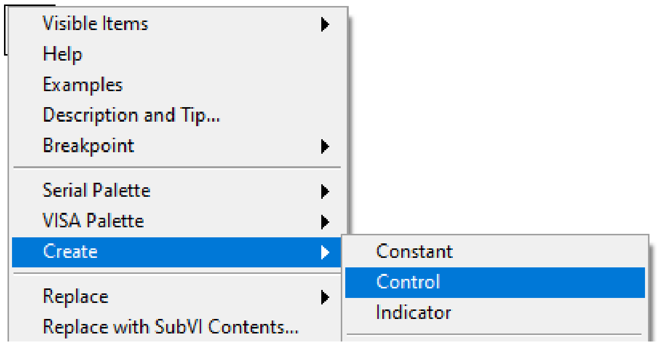
determine the polymorphic instance to use or manua
instance.

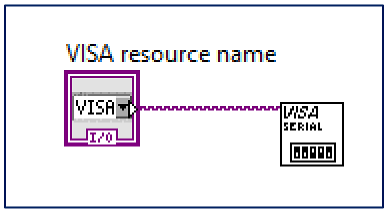
Terminal Data Type
VISA resource name (VISA session of class Instr)

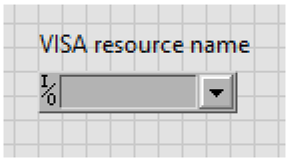
Detailed help

This will provide you with details on the VI, what each specific node purpose is, and, depending on the node/wire color, the data type associated with the node. Additionally, as you use your mouse to hover over each individual node, you will see a description for that node's purpose as well.

Hover your cursor just above the top left (purple) node of the "VISA Configure Serial Port" VI so that the cursor image changes into a wire spool and the node has a strobing black dot above it. Right-click on this node and, from the pop-up menu options that are presented, choose **Create->Control**. This will not just provide you with a control element on your block diagram, but it will also populate a corresponding user control on your front panel. (Remember, you can use the **Ctrl+E** hotkey combination to switch between these two user interfaces.)

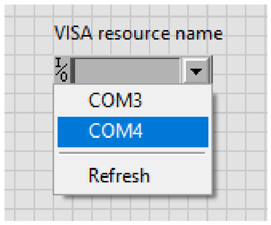






Switch to the front panel.

With the Bird 4421A connected to your PC via RS-232, the *VISA resource name* control will be updated to expose any valid communications options available to you. Clicking on the drop-down of *VISA resource name* may expose something like the following, where, in this case, the **COM4** serial option is that of the connected 4421A, and the one used in this example.

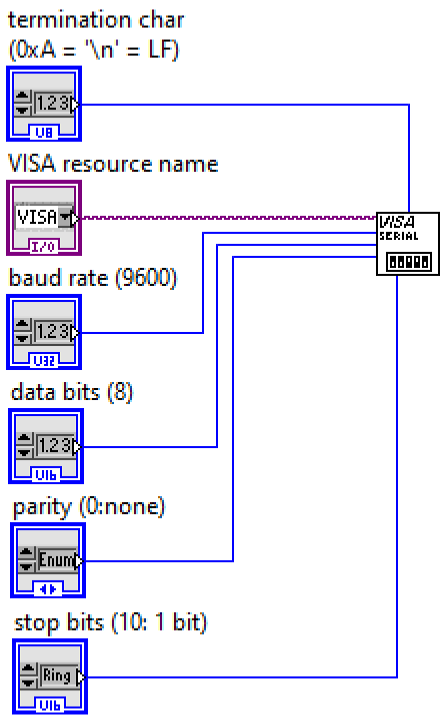


Switch to the block diagram.

From the 4421A User Manual, you will find that the following are the additional necessities you need to provide settings for:

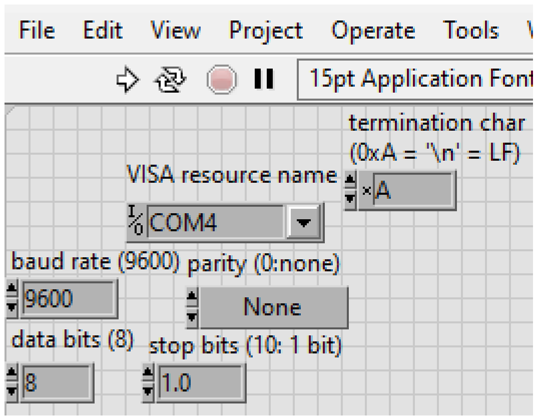
- Baud rate
- Data bits
- Stop bits
- Parity
- Termination character

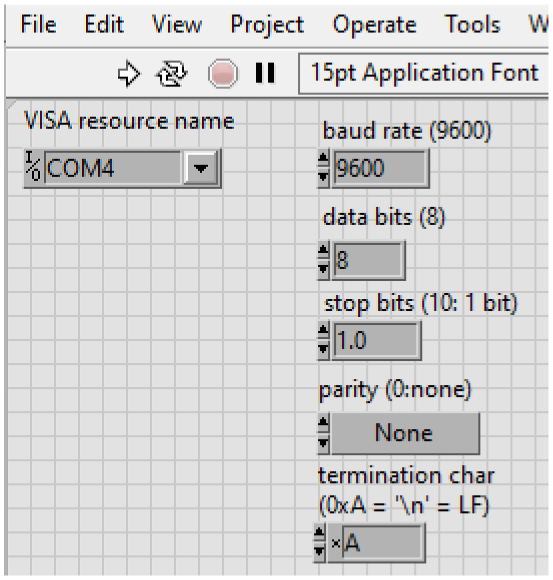
At this point, you can add controls for each of these the same way you did when you added the *VISA resource name*. Note that you can use your cursor to drag the controls and wire positions to whatever suits you. When completed, your block diagram may look something like the following.



Switch back to the front panel.

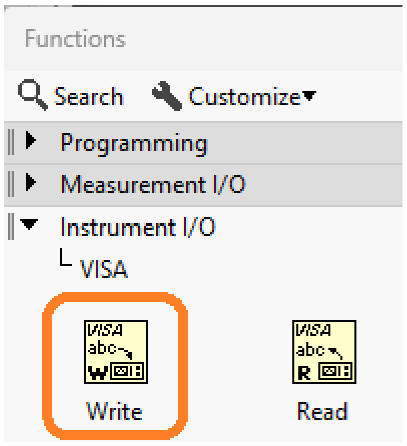
Notice all the new, additional controls have been added but they are in random order. Like you did on the block diagram, you can rearrange these to your liking. The transition may appear something like the following.

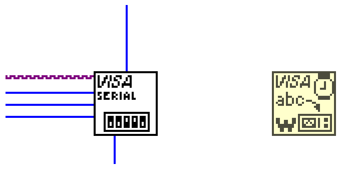




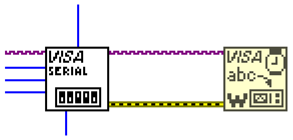
Switch back to the block diagram.

To be able to control your device, you will need to send it commands. To do so, use the Function pallet to search for “VISA Write”. When the function is shown, drag and drop the VI into your block diagram.

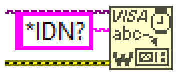




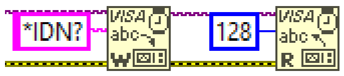
Using your Context Help and hovering above the nodes, you will find that this “VISA Write” VI is expecting three inputs and yields three outputs. You can use the outputs from your “VISA Configure Serial Port” VI as inputs to your “VISA Write” VI, daisy-chaining them together and making the latter dependent on the former.



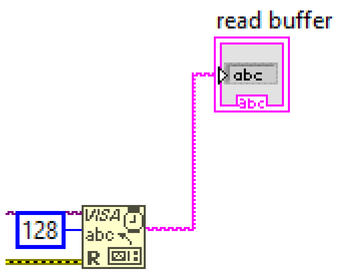
Similar to how you created controls, right-click on the *write buffer* node of the "VISA Write" VI but choose to **Create->Constant**. This will provide a blank field in which you can type your command. In this case, you will direct the 4421A to report its instrument identification string. To do so, you will type ***IDN?** into the available box.

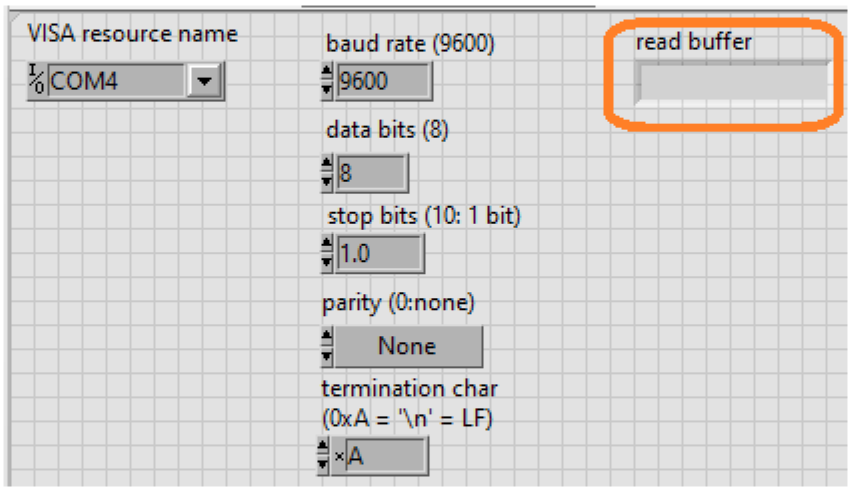


To receive information back from the 4421A, you will need to add a "VISA Read" VI. To do so, you search for this VI with your Functions Pallet, then drag and drop the VI into your block diagram. Again, you can leverage outputs from your previous VI to feed inputs of the "VISA Read" VI. The additional input you will need to provide here is a *byte count* number representative of the number of characters you anticipate receiving back from the instrument in its identification string. Here, we are okay to pass in 128 after creating a constant to hold the value.

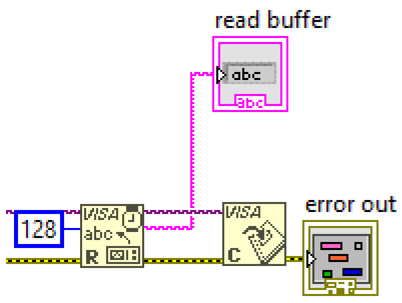


One last item you will want to add is something to present the readback information to the operator as they are viewing the front panel. Hover above the *read buffer* output node of the "VISA Read" VI, right-click and this time choose **Create->Indicator**. Like a control, an indicator will be added to both your block diagram and your front panel, and you will want to organize to meet your liking.

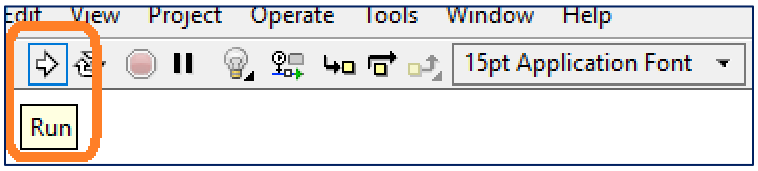




One last thing to add in your block diagram is the "VISA Close" VI. This close operation typically is added at the point in your program where no further communication between the PC and the instrument is necessary. As you did before, you will wire necessary outputs from your previous VI into the "VISA Close" VI. For a means of diagnostics, also create an indicator for the *error out* output node of your "VISA Close" VI.



If the identification string is all we want from the 4421A at this point, we can run the project from the block diagram view using the controls at the top of the screen.

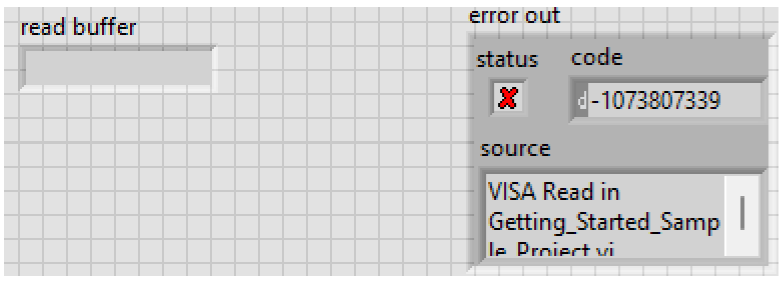


What you will experience at this point is that it will seem as though your program locks up for about 5 to 10 seconds.

Navigate to the front panel.

Notice two important things:

1. There is no response text in the *read buffer* indicator.
2. The *error out* fields are giving you an error code and messaging that helps point you to the problem spot.


read buffer

error out

status: ✗

code: -1073807339

source: VISA Read in Getting_Started_Sample Project.vi

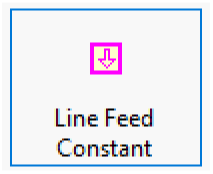
We will not go deep into troubleshooting in this document (you can save that for your future studies) but will simply point you to the problem and provide resolution.

The 4421A anticipates a line feed character at the end of each "VISA Write" so that it knows when to stop listening for the command instruction and move to acting on it.

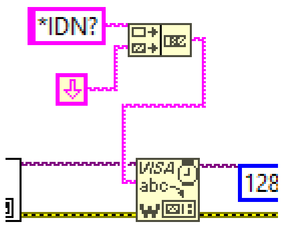
Switch back to the block diagram.

Using the Functions Pallet, search for "Concatenate Strings" VI then drag and drop this near the "VISA Write".

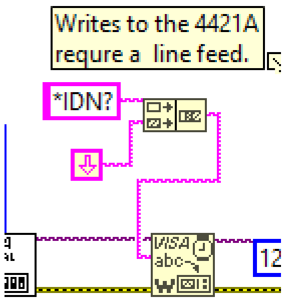
Use the Functions Pallet again, this time navigating through its standard tree features to **Programming->String->Line Feed Constant.**



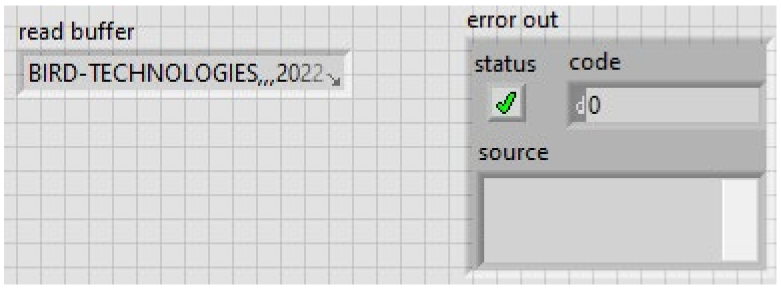
Drag and drop this near the other two VIs, and re-wire the inputs to the "VISA Write" as follows:



If you need a reminder about how the writes are required to operate, you can use your left mouse button to double-click on the open whitespace of your block diagram and leave yourself a note.



Running the code again, you will notice that the completion time is almost instantaneous. Further, navigating back to the front panel will show that information is now available in the *read buffer* indicator and the *error out* indicator fields are cleared of the error.

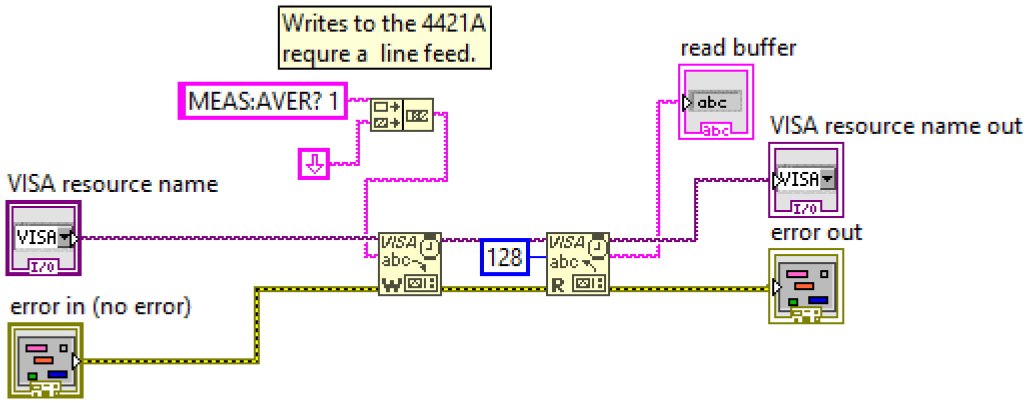


Creating Reusable VIs

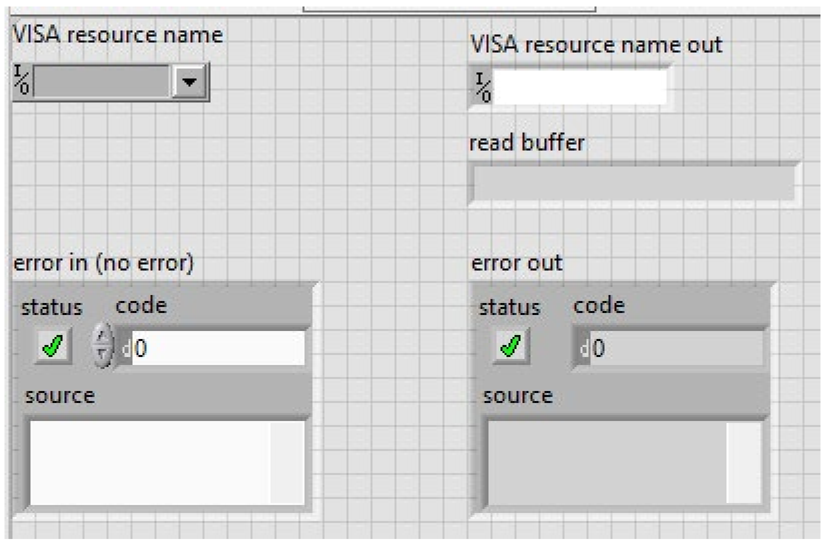
Over time you may find yourself repeating the same or similar steps that you would normally build into a function that accepts arguments and provides return values. As you may have guessed, you can construct your own VIs to accomplish this.

In either your existing front panel or block diagram, use the menu options to select **File->New VI** and save this with the name "**Measure Forward Power**". _Note: Do not close your original VI_.

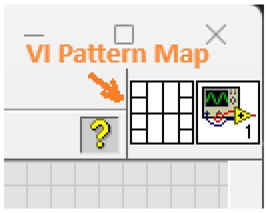
Similar to what you did with the previous VI, you will add "VISA Write" and "VISA Read" VIs to the block diagram. You will want to add controls to pass in the *VISA resource name* and *error in* arguments directly to the "VISA Write" block. Additionally, this block will need to send the string "MEAS:AVER? 1" to tell the 4421A to return the measured forward power to the caller. Further, you will want to add indicators for *VISA resource name* out and *error out* to the "VISA Read" block as well as an indicator for *read buffer*. Your final block code should appear like the following:



Switch to the front panel and organize your controls and indicators similar to the following:



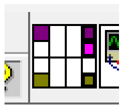
To make this usable within another VI, locate the VI pattern map in the upper right corner of the front panel.



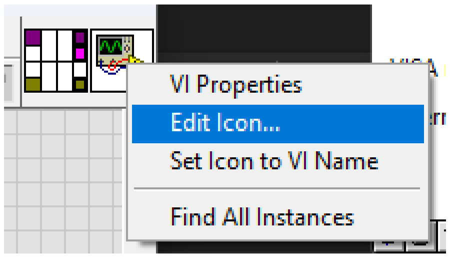
Use your mouse cursor to click on the pattern's individual cells one at a time, each time associating the cell with one of the front panel elements. The recommended assignment is as follows; note how inputs are to the left and outputs are to the right:

- Far left column, top cell – *VISA resource name*
- Far left column, bottom cell – *error in*
- Far right column, top cell – *VISA resource name out*
- Far right column, second cell down – *read buffer*
- Far right column, bottom cell – *error out*

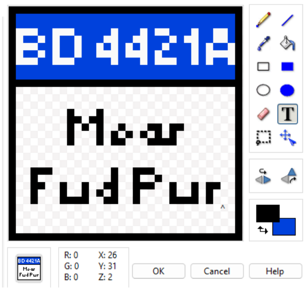
You will notice that each of the cells you assign will change color to match the underlying variable type used on the block diagram. Your updated VI Pattern Map should appear as follows:



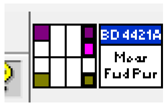
As you continue to build a greater number of VIs that may be associated with 4421A, you will want to add some additional visual distinction to each to make them better identifiable as they are added to your code. To do this, right-click on the VI icon (adjacent to the pattern map) and select Edit Icon.



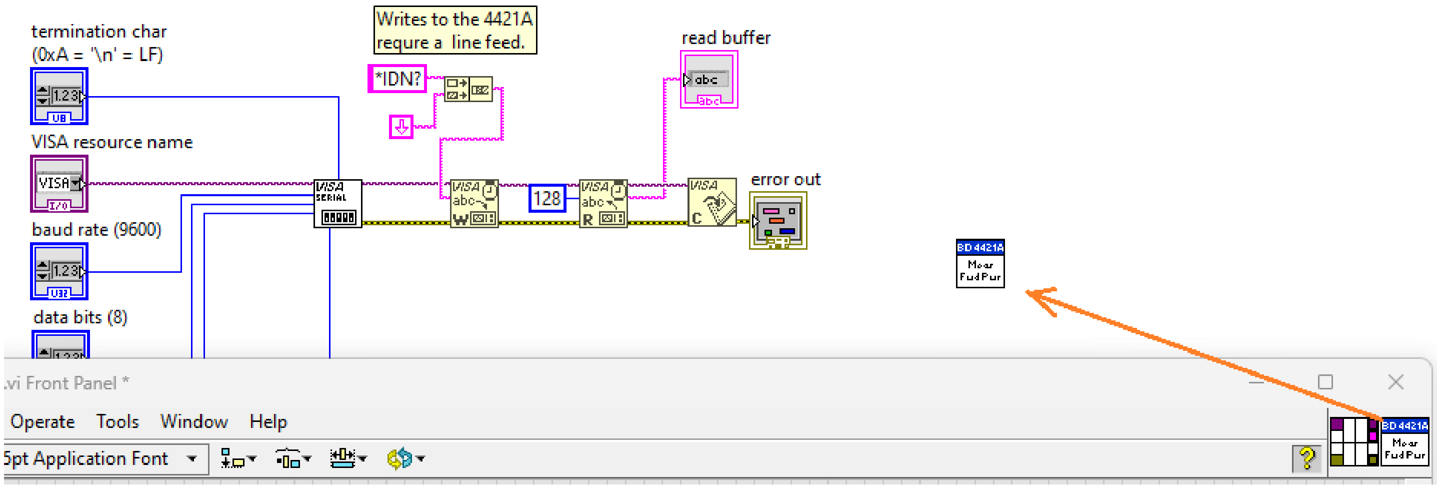
Use the editor tools to try to come up with something like the following:



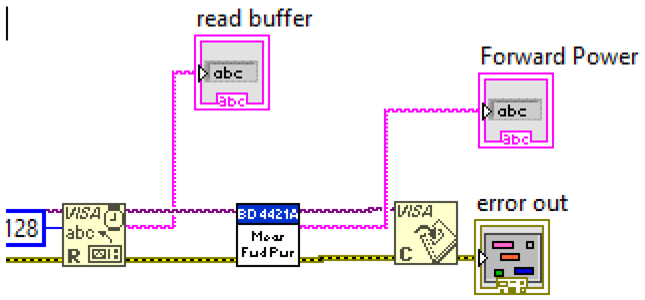
Click on OK then notice how the icon in the upper right corner of the front panel reflects your changes.



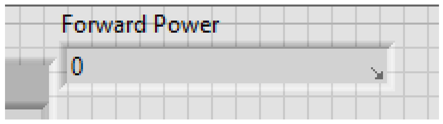
You now have a reusable VI that you can include in any of your other coding efforts. To emphasize this, expose the block diagram from your original VI then drag and drop your newly created icon into it:



You can now insert this VI between the "VISA Read" and "VISA Close" blocks, rewiring inputs and outputs as needed as well as adding an indicator to show the measured forward power. The result should look something like the following:



Running your code will give you the measured forward power as reported by the 4421A, in this case "0" since no transmitter or load were attached to the sensor connected to the 4421A.



A Word on Instrument Drivers

The exercises provided above are used to give you an idea on how you might start to establish a base of code to reuse and share with your test development team. Bear in mind that many instrument manufacturers often take this type of development support as a necessity and build a library of VIs specific to a particular model of instrument then make available for public use. This is what we refer to as an instrument "driver" and they are often made available in a variety of programming languages. Because of its popularity, it is likely that you can find a LabVIEW instrument driver readily available for your immediate needs through NI's Instrument Driver Network (IDNet)iii.

If you want or need to build your own LabVIEW instrument driver, there are resources on NI's website to provide you guidance on how to do so properly.iv

Conclusion

LabVIEW is a popular and powerful programming language that enables engineers and programmers with a unique, visual approach to creating control code and user interfaces. The software can be used for learning and sharing without fee, but also offers price plans for those who intend to use it to server their own paying customers and those teaching at degree-granting institutions.

You now know where to get the LabVIEW software and how to install it. Moreover, you now have the knowledge on how to use basic VISA VI blocks to connect with, write to, read from, and properly close out of a connected instrument communication session. You should also be able to build your own custom, reusable Vis.

Instrument drivers provide a great starting point for instrument control, cutting down on the amount of code you might need to create yourself. When you cannot find a driver you are looking for – either through NI or directly from your instrument's manufacturer – you can always create your own.

---

[i] What is LabVIEW - https://www.ni.com/en-us/shop/labview.html
[ii] NI-VISA Overview - https://www.ni.com/en/support/documentation/supplemental/06/ni-visa-overview.html
[iii] Instrument Driver Network - https://www.ni.com/en/support/downloads/instrument-drivers.html
[iv] Developing LabVIEW Plug and Play Instrument Drivers - https://www.ni.com/en/support/documentation/supplemental/21/developing-labview-plug-and-play-instrument-drivers.html