

# **Computer Interfacing Using LabView**

Physics 258

Last revised September 25, 2005 by Ed Eyer

## ***Purpose:***

To write a simple LabView program that digitizes data using an ADC on a data acquisition card, under computer control.

## ***Note:***

It is especially important to read the LabView tutorial ahead of time, so that you will be prepared when the lab begins.

## ***Equipment:***

The usual stuff found in room P303, plus:

1. Computers with LabView and National Instruments Data Acquisition (DAQ) cards.
2. National Instruments SC-2075 interface boards, or screw-terminal interface blocks together with an adjustable 0-10V power supply.

# ***Introduction to National Instruments LabVIEW and Data Acquisition (DAQ)***

Written by Danial J. Neebel, Joseph R. Blandino, and David J. Lawrence,  
College of Integrated Science and Technology  
James Madison University

Modified by Ed Eyler, University of Connecticut Physics Dept.

## **Introduction**

In this lab, you will write a LabVIEW virtual instrument (VI) to read voltages and display them. The idea is to build a VI that works like a graphing digital multimeter. Your multimeter will be able to set the range of values for the readings (resolution) and display the reading on a digital readout and a chart.

## **Objective**

- To navigate the LabVIEW graphical programming language environment.
- Take voltage measurements using computer-based instrumentation.

## **Theory**

This lab covers two very broad areas of study in instrumentation: graphical computer programming and analog-to-digital conversion. Both areas require an entire book to thoroughly study the topics. However, the following short introduction to graphical programming will allow students to take simple measurements. You should already be familiar with analog-to-digital conversion from the textbook by Wheeler and Ganji, *Introduction to Engineering Experimentation*, and the lecture periods.

### **Graphical Computer Programming**

Traditional computer programming involves setting down a list of tasks for the computer to execute in the given sequential order. Each instruction is executed in the order of appearance in the list. Often, the availability of data determines the order given to these instructions. For example, instruction 3 in Figure 1 requires data calculated in instruction 2.

Therefore, instruction 2 must execute before instruction 3. Instruction 3 has a *data dependency* on instruction 2. Note that instruction 2 has a data dependency on instruction 1. Because instruction 4 does not require the result from any other instruction in the sequence, it has no data dependencies on instructions 1, 2, or 3. Because instruction 4 is data independent with all the other instructions, it does not matter when it executes.

1. Add A to B
2. Add C to the Sum of A and B
3. Divide Sum of A,B, and C by 3
4. Subtract A from C

Figure 1. A Sequence of Instructions

This discussion of data dependency leads to a new way of programming. If you specify the operations and the data dependencies, the computer can execute the instructions in any order that protects the data dependencies. Now you need a way of easily specifying data dependencies. If you can draw a block for each operation and connect the blocks to show the dependencies, you can program the computer by drawing a picture. For most people, pictures are much easier to understand than a list of instructions.

LabVIEW programming consists of drawing pictures that specify data dependencies. The LabVIEW programming environment includes a large set of blocks to specify operations and a Wiring tool to connect them together. As an example, Figure 2 shows the operation of multiplying two numbers and displaying the result.

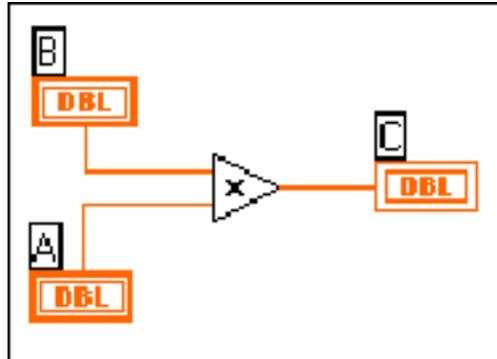


Figure 2. Program for Multiplying Two Numbers and Displaying the Result

A LabVIEW program, called a virtual instrument (VI), is a two-window system. The code is in one window and the user interface (inputs and outputs) appears in a separate window. The program window is the block diagram window, and the user inputs and outputs are in the front panel window. Figure 2 shows a sample program that would appear in the block diagram window. The numbers are entered into the computer and displayed in the front panel window shown in Figure 3. The two boxes on the left (labeled **A** and **B**) are controls, and the box on the right (labeled **C**) is the output or indicator. (The **X** and **=** are only displays showing the operation of the VI and not inputs or outputs.) The three boxes are

associated with like labeled boxes in the diagram window shown in Figure 2.

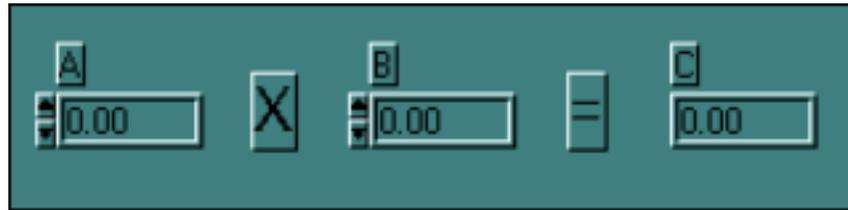


Figure 3. Front Panel for a Two-Number Multiplication Program

Figure 4 shows a more complicated program. This program reads a voltage and adds it to a chart. The gray box around the program is a While Loop. The program elements inside the While Loop will execute repeatedly as long as the input to the condition terminal is false. That is, as long as the variable **Stop** is false. **Stop** is the button on the front panel shown in Figure 5. When the user presses the button, **Stop** becomes true, and the While Loop stops executing. When the While Loop in this example stops, there are no other program elements to execute, so the entire program stops running.

In the center of the While Loop, you see the “work” being done in the loop. The block labeled **AI ONE PT**, which is actually **AI Sample Channel.vi**, performs the operation of getting a voltage from the channel specified by **Channel** and the data acquisition device specified by **Device**. **AI ONE PT** reads one voltage reading from the specified channel on the specified data acquisition device. The output of **AI ONE PT** is a voltage. Each time the While Loop runs, **AI ONE PT** outputs one voltage value to the terminal labeled **Voltage Display**. The terminal labeled **Voltage Display** is the connection point for the chart in the front panel shown in Figure 5. The update of the chart is such that every time a new number is input to it, the new number is plotted along with all the previous numbers.

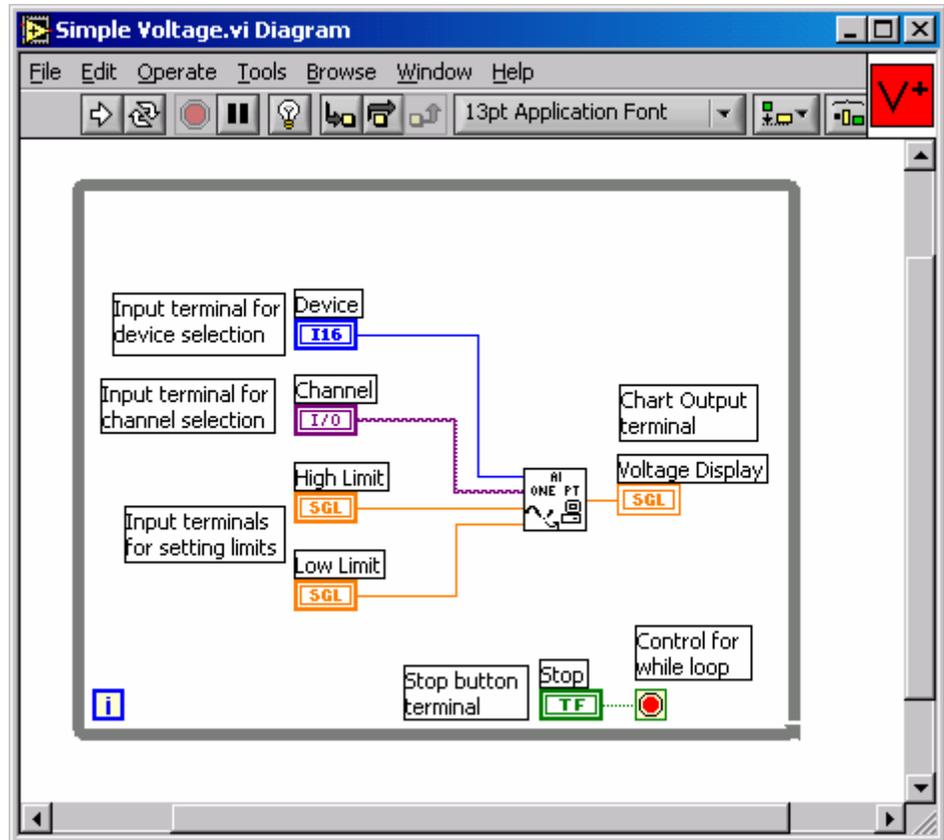


Figure 4. LabVIEW Program to Read a Voltage from a Single Channel

The **AI ONE PT** block in Figure 4 performs the data acquisition procedure. National Instruments has completed a major portion of the programming by writing software that drives the DAQ system. That means that all you need to do is include the **AI ONE PT** block in the program to take voltage measurements with your DAQ device.

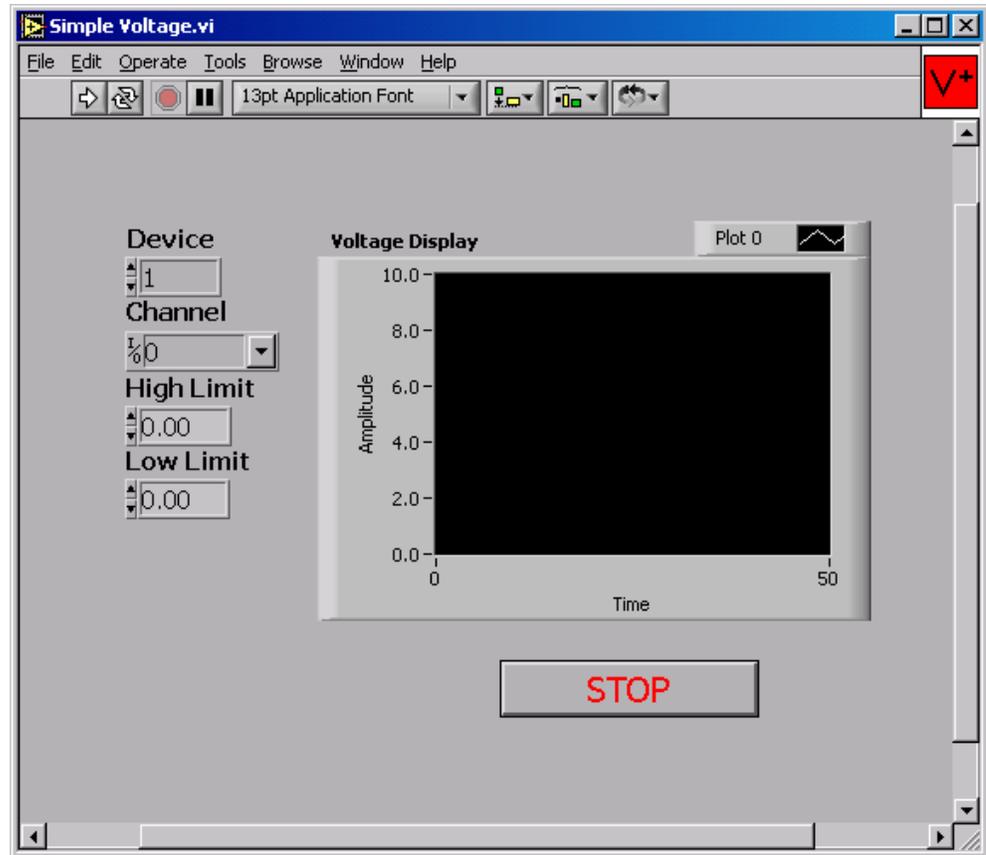


Figure 5. Front Panel of a Program that Reads and Displays a Voltage Waveform from Device 1 and Channel 0.

### Pre-Lab Preparation

- Read through the theory and lab procedure for this experiment. Come prepared to execute the lab exercise.
- Bring the following to lab with you:
  - This experiment.
  - Your lab notebook and pencil.

### Workstation Details

Your workstation should have the following items:

- Computer with National Instruments LabVIEW software.
- National Instruments DAQ board (inside the computer).

- National Instruments SC-2075 Breadboard Connector (outside the computer), or a screw-type terminal block (less desirable).
- Standard wires.
- Variable voltage source, if not using a SC-2075 board.

## Lab Procedure

### Part 1

#### A LabVIEW Tutorial: Displaying Scaled Random Numbers on a Chart

The following tutorial will help you learn the basics of LabVIEW programming. Read each step completely before executing the step. By the end of the tutorial, you will have written a VI that displays scaled random numbers on a chart. After completing this tutorial, you should be able to complete the rest of the lab. Be sure that everyone in the group gets a chance at the computer during the tutorial.

Table 1. Some Useful Commands and Tools

Command/Tool	Purpose	Used When	Picture
Delete key	Deletes selected objects	There are unwanted objects in the program	<Delete>
Ctrl-S	Saves files	You want to save your changes	<Ctrl-S>
Positioning tool	Moves and selects objects	You need to be move or delete program elements or insert new ones	
Wiring tool	Connects objects together	Program elements must be connected to allow data to flow between them	
Ctrl-B	Removes all broken wires	There are several unwanted wires in the program; use with caution	<Ctrl-B>
Operating tool	Changes values	You need to change a value in a front panel object	
Text tool	Edits text	You need to change a label or a comment	

1. Setup:
  - a. Insert your *blank* floppy disk into the disk drive. Check the disk directory to make sure your disk is formatted and is free of any viruses. A simple way to achieve this is to format the disk, but you will lose any information stored on the disk.
  - b. Launch LabVIEW from the **LabVIEW** group in the **Start** menu of the task bar.
  - c. When prompted to open an existing or new VI, select **New VI**.
  - d. When the new VI windows appear, select **Windows»Show Tools Palette** to display the **Tools** palette.
  - e. From the **Tools** palette, select the Positioning tool, shown in Table 1.
  - f. From the **File** menu, select **Save As** and save the file to your floppy disk (drive a :) under a suitable name. The file extension must be \*.vi. It is a good idea to save the file every few minutes during the development process. Save the file after making a change you want to keep.
  - g. Review the commands and tools in Table 1.
2. Virtual instrument programming:
  - a. Click on the Block Diagram window (the window with the white background) to bring it to the front.
  - b. Insert a While Loop into the diagram window. First, open the **Functions** palette by clicking the right mouse button with the cursor in the block diagram window. Then move the pointer down to the **Structures** palette button (upper left button). When the cursor reaches the button, a palette of program elements will appear. Click on the While Loop (icon on the far right in the top row).

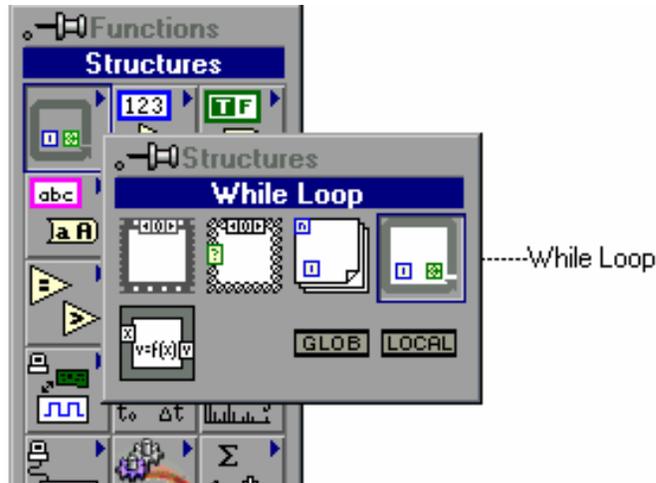


Figure 6. Palette showing location of While Loop

- c. The While Loop first appears in the Block Diagram window as a box-shaped cursor. Insert the loop by placing the cursor in the upper left corner of the block diagram window and clicking and dragging the icon to the lower right corner. Make the While Loop almost as large as the window, but don't fill the entire window.
- d. Insert the **Random Number Generator** function into the While Loop. Click the right mouse button as before. This time, select the **Numeric** button by moving the cursor to the **Numeric** palette button (the center button in the top row). Choose the icon that looks like a pair of dice to add the **Random Number Generator** function to your block diagram.
- e. Press <Ctrl-H> to open the Help window. Move the cursor to the pair of dice and click once. Read the information in the Help window. This help feature can be useful when determining what connections need to be made to a VI.
- f. Click on the Front Panel window (the window with the gray background).
- g. Insert a Waveform Chart. Right-click in the Front Panel window to bring up the **Controls** palette. Click on the **Graph** button (the right button in the second row) in the **Controls** palette. Choose **Waveform Chart** from the palette, move the cursor back to the Front Panel window, and click to insert the chart wherever you want.
- h. Name the chart *Scaled Data* by typing the name and clicking on the Enter icon of the tool bar. You should see the text appear in a box near the upper left corner of the chart. If not, try pointing the cursor at the chart and clicking the right mouse button. In the

menu that appears, select **Show»Label** from the submenu and type the title.

- i. Point at the chart and click and hold the right mouse button. Select **Show»Digital Display** from the submenu.
  - j. Point the cursor at the chart and click and hold the right mouse button. Select **Find Terminal** in the pop-up menu and release the button. This should bring up the Block Diagram window, and the terminal for the chart will be highlighted with dashed lines. Make sure the chart terminal is inside the while loop. If it isn't, use the positioning tool to drag it into the while loop.
  - k. Connect the Random Number Generator to the chart terminal. Select the Wiring tool from the **Tools** palette. (It looks like a spool of thread.) Use the Wiring tool to connect the output of the dice to the terminal for the Scaled Data chart by pointing the tool at the dice and clicking once. Move the tool to the indicator terminal (a small rectangle with DBL inside) and click once more. An orange line should appear.
  - l. Click on the Front Panel window. You will now insert a stop button to control the While Loop execution. You can find the button palette by choosing the Boolean controls from the **Controls** palette. You may select any button, as long as it is a button and not an LED, light, or switch. Type *STOP* as a label for the button and click the Enter button (check mark in the top left) on the Toolbar.
  - m. When you have the button in place, point at it with the Positioning tool. Click and hold on the right mouse button and select Find Terminal to bring up the Block Diagram window. Make sure the boolean terminal is inside the while loop. If it isn't, use the positioning tool to drag it into the while loop.
  - n. In the Block Diagram window, add a logical inverter (**Functions»Boolean»Not**), and place it near the STOP terminal you created in the previous step.
  - o. Use the Wiring tool to connect the STOP terminal (small rectangle with TF written inside) to the inverter input, and then connect the inverter output to the conditional terminal that controls the While Loop. The conditional terminal should be in the lower right corner of the While Loop.
3. Click on the Front Panel window. Now you can test your VI. Make sure that the STOP button is in the 'OFF' position (i.e., not stopped); if it isn't, switch it using the operating tool. Click on the Run button, the single arrow in the upper left corner. To stop the execution, press the STOP button you put on the front panel. Run the VI several times.

Does the VI run? How do you know? How could you determine the number of times the while loop executes each time you run the program? Hint: What does the other small square with an “i” do in the bottom left corner of the While Loop?

4. Have your instructor check your progress. You may want to use the Positioning tool to rearrange some of the icons to make the program clearer. In general, it is best to place input terminals on the left and output terminals on the right. Also, the wires in between should not cross unless absolutely necessary.
5. Save the VI to a floppy disk, if you wish.
6. In the next few steps, you will add parameters to your program to provide a scaled random number between user-defined values of X and Y.
  - a. Go to the Front Panel window and insert two digital controls (under the **Controls**»**Numeric** palette). Name one of the controls “upper limit” and the other “lower limit.”
  - b. Use the Positioning tool to arrange the controls to make your front panel look presentable.
  - c. Switch to the Block Diagram window. Make sure the terminals for the new controls are inside the While Loop. If they aren’t, use the positioning tool to drag them into the while loop.

Before continuing, you need some background on mathematical operations in LabVIEW. Math operations are programmed using triangular icons. All operations used here are binary, meaning they have two inputs. For example, the subtract icon in Figure 7 has two inputs on the left and one output on the right. To subtract Y from X (as shown below), you connect X to the upper left corner and Y to the lower left corner. The difference (answer) is then available at the right corner.

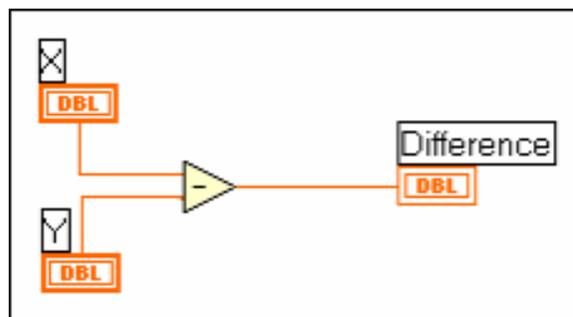


Figure 7. Subtract Operation in LabVIEW

7. Insert a **Subtract**, a **Multiply**, and an **Add** function from the **Numeric** palette. Use the Wiring tool to connect these three arithmetic functions together with the **Random Number Generator** function and the chart terminal to provide the following function. The operators you need are shown below the equation. (Hint: Set up the program so that the subtraction operation takes place first on the left of the screen, then the output of the subtraction is multiplied by a random number. **Note:** This is not the order shown below!)

$$\text{Output} = \text{Random} \times (\text{Upper} - \text{Lower}) + \text{Lower}$$



Figure 8. Multiply, Subtract, Add functions as shown in LabVIEW, respectively.

8. Save the VI to your floppy disk. Run the VI with five different sets of values of Upper Limit and Lower Limit. (Use the Operating tool to change the values.) Record a few of the values you get after pressing the stop button each time. Did you get any values outside the limits? Write the numbers and your answer on your data sheet.

## Part 2

Build a graphing digital voltmeter. This part of the lab requires you to do some experimenting and use what you have learned from Part 1.

### Building a Voltmeter with LabVIEW

1. Now you will modify your program from Part 1 so it looks like Figures 4 and 5. First, save the program under a different and still meaningful name, such as **Graphing Digital Voltmeter.vi**.
2. The VI used to take voltage measurements is **AI Sample Channel** (located in **Data Acquisition»Analog Input»AI Sample Channel**). Remember that you find this menu by pointing the cursor where you want the icon to appear and clicking the right mouse button.
3. Open the Help window <Ctrl-H> if it is not already open.
4. Use a digital control or numeric constant to set the device number for **AI Sample Channel** to 1. For the channel number, use the default value of 0. (If you wish to change this, use a string control to set the channel number for **AI Sample Channel**. String controls work the same as digital controls, except they use a different data type.)
5. Use the upper and lower limit controls of the tutorial to set the upper and lower limits of the voltmeter. The smaller the range of limits that

you set, the higher the gain of the amplifier in the DAQ equipment. In other words, the closer the range is set to the actual range of values, the more precise your voltmeter will be. The gain settings are rather coarse, so you may find that small changes in the upper and lower limits have no effect.

6. Remember to save your VI after every few changes, so you can return to a known state if necessary.

### **Lab Measurements**

1. Connect the CH0- and CH0+ terminals on the National Instruments SC-2075 Breadboard Connector to an adjustable 0-5V supply, available on the terminals labeled GND and 0-5V+.
2. Have your instructor check your system and program.
3. Run your VI.

### **Extra Credit Items**

- Make your VI read and display the voltage of more than one channel.
- Make your VI read several samples at one time but using a subVI designed to acquire a waveform. A sub VI is another VI called inside of your main VI.
- Use your imagination to create a new application for your VI.

## **Lab Report**

- The completed VI will be considered an informal report. Just demonstrate your working voltmeter to your TA.