

Experiment-0

Data Acquisition Using LabVIEW

Introduction The objectives of this experiment are to become acquainted with using computer-controlled instrumentation for data acquisition. LabVIEW, a program developed by National Instruments, is the industry standard for programming computer-controlled instruments, and it will be used in this experiment as well as others to measure and record sensor readings and to characterize various electrical systems and devices.

LabVIEW is a graphical programming environment. Unlike C/C++ where programs are written in text, in LabVIEW creates Virtual Instruments (VIs) by graphically composing them from different elements and structures that are placed like the blocks of a block diagram and interconnected with wires to indicate the intended signal flow paths. This is referred to as “G-code” for the graphical language that it uses. The most important aspect of understanding LabVIEW virtual instruments (VIs) is that they are data-driven, meaning that the execution of the block diagram proceeds along the same path as which the data propagates through the block diagram. A new block is not executed until the new data arrives at its input. This is quite different from the event-driven programming of Windows, or the more familiar procedure-driven programming of FORTRAN, Pascal, or C/C++. A complete tutorial for programming in LabVIEW will not be presented in this laboratory handbook since other excellent references exist for this purpose.

This experiment will introduce opening and running virtual instruments in LabVIEW and using it to control data acquisition (DAQ) hardware for making electrical measurements. In addition some simple modifications to the virtual instruments will be performed to gain some experience with using the LabVIEW graphical interface and programming language. These basic operation skills will be useful starting points for developing more complex data acquisition instruments in later experiments, and will form the basis for automated measurement of semiconductor device characteristics. One of the important advantages of computer-based instruments is that recording measurement data becomes very easy, and some of this will be introduced in this experiment also.

Experiment-0

Procedure 1 Temperature measurement using LabVIEW and DAQ hardware

Comment The goal of this procedure is to get LabVIEW up and running and open an existing VI which can be used to measure and log the ambient temperature using an LM35DZ integrated circuit temperature sensor.

Set-Up The first step is to set up the data acquisition (DAQ) hardware. All of these laboratories will utilize the National Instruments NI-USB-6009 DAQ which provides 8 single-ended or 4 differential analog inputs with 14-bit resolution and up to 48 kS/s sampling, 2 analog outputs with 12-bit resolution and up to 150 S/s data rate, a 12-bit multifunction digital I/O block, and a built-in 32-bit counter module. This DAQ is very versatile and completely powered through the USB connection. To connect the DAQ to the computer, simply use a standard USB cable to any USB 2.0 port on the computer. Once connected, the DAQ will be recognized by the computer, and if a driver has not already been installed, the Windows operating system will automatically install the driver for it. Connect the NI-USB-6009 DAQ to the computer using a standard USB cable if it is not already connected.

Inputs and outputs to the DAQ are from 16 screw terminals on each side of the DAQ. The most convenient method for using the NI-USB-6009 in these laboratories is to use a solderless breadboard, also known as “superstrip”, and make connections to the DAQ using some solid wire jumpers, as shown below in Fig. E0.1a. The best wires to use are 8 inch long, solid, #22 gauge, with different insulation colors to distinguish the different pins, as also shown in the figure. This system then allows subcircuits to be easily constructed on the superstrip and allows the DAQ to easily connect to any tie points on the superstrip for signal acquisition or injection.

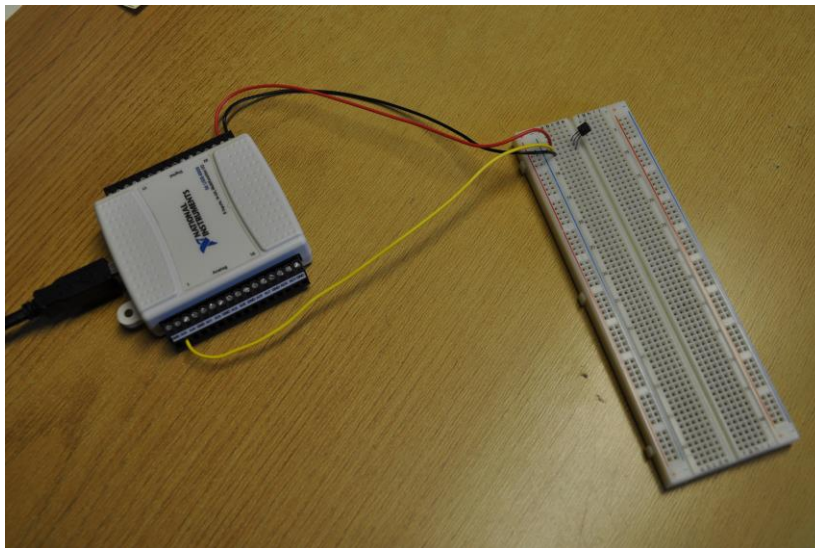


Figure E0.1a

Experiment-0

The ambient temperature is measured by an LM35DZ temperature sensor which is a small 3-lead part in a TO-92 case. A DC voltage of 4~20 Volts is supplied between the VS and GND pins, and the device will output a DC voltage between the VOUT and GND pins which is proportional to the ambient temperature in degrees Celsius. The conversion factor for the LM35DZ is 100 mV/°C, and it is calibrated so that 0°C will produce 0.0 Volts output. Looking at the LM35DZ with its labeled flat side facing you and its leads pointing downward, the pin on the left is VS, the pin in the middle is VOUT, and the pin on the right is GND. A +5 Volt DC power supply can be obtained from the NI-USB-6009 DAQ itself, using the +5V terminal, pin #31, and the digital ground terminal GND, pin #32. These two leads can be used to power up the LM35DZ temperature sensor. For a stable output, the LM35DZ should have a 200 Ω load resistor connected between the VOUT and GND pins. The output voltage is then taken across this 200 Ω load, as shown in Fig. E0.1b below.

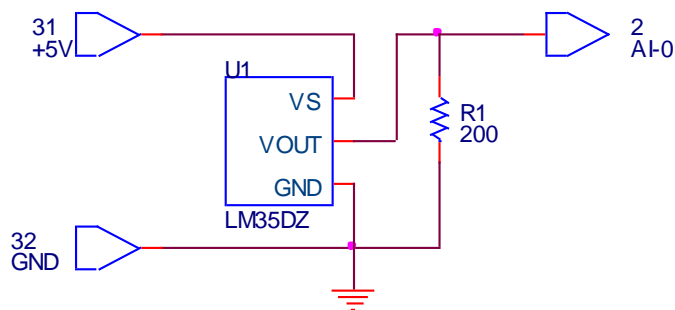


Figure E0.1b

As shown below in Figs. E0.1c and E0.1d, use a superstrip and some solid insulated wire jumpers to connect an LM35DZ temperature sensor to the NI-USB-6009 DAQ. Connect a wire from analog input 0 (ai0, pin 2) on the NI-USB-6009 DAQ to VOUT lead (center) of the LM35DZ, shown in yellow in the figures below. Connect a wire from ground (GND, pin 32) on the NI-USB-6009 DAQ to GND lead of the LM35DZ, shown in black in the figures. Connect a wire from the +5.0 Volt power supply (+5V, pin 31) on the NI-USB-6009 DAQ to VS lead of the LM35DZ, shown in red in the figures. Finally, connect a 200 Ω 1/4 Watt resistor between the VOUT and GND leads of the LM35DZ. The two black screw-terminal blocks on the DAQ should have some decals attached which identify the terminals. If not, see the TA or lab manager for some replacement decals. The terminal blocks can also be removed and changed from the DAQ, if needed as well.

Experiment-0

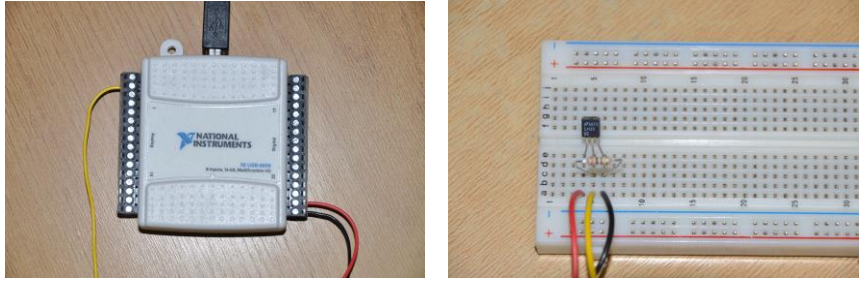


Figure E0.1c,d

A very simple and quick method to test that the DAQ card is connected and working properly is to use the National Instruments Measurement and Automation Explorer (NI-MAX). From Windows, launch the Measurement and Automation Explorer (NI-MAX) from the Start Menu by clicking on Start > All Programs > NI-MAX, or alternatively, there may already be a shortcut for NI-MAX on the desktop. After NI-MAX opens, on the left hand side of the window is a configuration panel. Click on the expand button [+] beside Devices and Interfaces. Select the NI USB-6009 DAQ, and the Settings and External Calibration should appear in the central panel. Above the central panel, click on a toolbar button called “Self-Test.” This should return a small message window saying that the device has passed its self test. This indicates that the DAQ is properly connected to the computer, that Windows has properly recognized the device and has loaded its drivers, and that LabVIEW has properly registered the device so that it can be accessed by various VIs that call it. This self-test only tests the DAQ computer connection, not anything that may be connected to the DAQ I/O pins.

If you wish to test the system further, the toolbar button to the right of “Self-Test” is “Test Panels...” and this provides a more detailed set of commands which directly control the DAQ card and can be used to insure that the entire system is working properly. Click on “Test Panels...” and select the Analog Input tab. Make sure that the Channel Name is set to Dev1/ai0 and the Mode is set to On Demand. Change the Max Input Limit to 5 and Min Input Limit to 0, and change the Input Configuration to RSE (referenced, single-ended). Click on Start, and the chart should show a streaming set of analog measurements from the ai0 channel of the DAQ in the range of 0.25 to 0.35 (Volts). This validates that the LM35DZ is properly connected to the DAQ and that the DAQ is properly sampling the voltage on the center output lead of the device. Click Stop to halt the data streaming and close the Test Panels window.

The National Instruments device driver system allows multiple USB hardware devices to be installed simultaneously. Each are uniquely identified by their hardware model number and serial number, and each are given a name which allows them to be more conveniently referred to in various VIs. The present name for the NI-USB-6009 appears in the box to the right of “Name” and it probably says something like Dev1 or Dev2. All of the laboratory VIs have

Experiment-0

been written for the NI-USB-6009 DAQ as “Dev1.” If the NI-USB-6009 is not listed as “Dev1” you can change this by typing in “Dev1” in the name box and clicking on the Save button at the left top of the central panel. If it responds saying that Dev1 is already in use, make the change anyway. Do this now if your DAQ is not already listed as “Dev1.” If the DAQ has the wrong name assigned to it, none of the VIs will properly recognize it. Once you have verified that the DAQ is properly connected and named Dev1, you can EXIT MAX at this point.

The next step is to launch LabVIEW and start up a VI which has already been written for making a simple temperature measurement. From the Windows Start Menu, click on Start > All Programs > National Instruments LabVIEW, or use the desktop shortcut if it is present. All of the VIs are written in LabVIEW version 13.0, and will run on any version of LabVIEW that is 13.0 or more recent. The main LabVIEW window should appear. Use the Open Existing button to navigate to the EE-332 laboratory VIs and select the TempSensorReadout.vi. LabVIEW will first open the front panel window for this VI which is from where the virtual instrument is controlled. If you wish to view the internal G-code for this instrument, click on Window > Show Block Diagram (or type Ctrl+E). This is a relatively simple VI, and the block diagram shows how the voltage reading from the DAQ card is first multiplied by a factor of 100 and then sent to a waveform chart for display. The temperature readings are taken each 500 milliseconds, and a STOP button is set up to end the program. LabVIEW virtual instruments (VIs) always consist of two parts: a front panel and a block diagram. Ctrl+E switches between the two. Only the front panel needs to be open in order to run the VI.

Measurement-1 To start the temperature measurement VI, make the front panel window active by either clicking on it somewhere or select it from the Window menu. Click on the run button, which is shaped like a right-pointing arrow on the toolbar. The front panel appearance will change from the grid pattern to a solid pattern, and the stop sign will turn to brighter shade of red. The waveform chart will start scanning and you should soon see a signal waveform appear that represents the running output of the LM35DZ temperature sensor being sampled every 500 milliseconds.

After the measurement has been running for a few seconds, press your finger tip onto the top of the LM35DZ temperature sensor and you should see the measured temperature rise by a few degrees Celsius. After removing your finger tip, you should similarly see the measured temperature fall back to close to its original value. Once you have finished running the VI, you can stop it by simply clicking on the rectangular STOP button on the front panel. If for some reason this does not work, you can stop the VI by clicking on the red stop sign button in the toolbar. The results of a rising temperature are shown in the screen shot of Figure. E0.1e.

Experiment-0

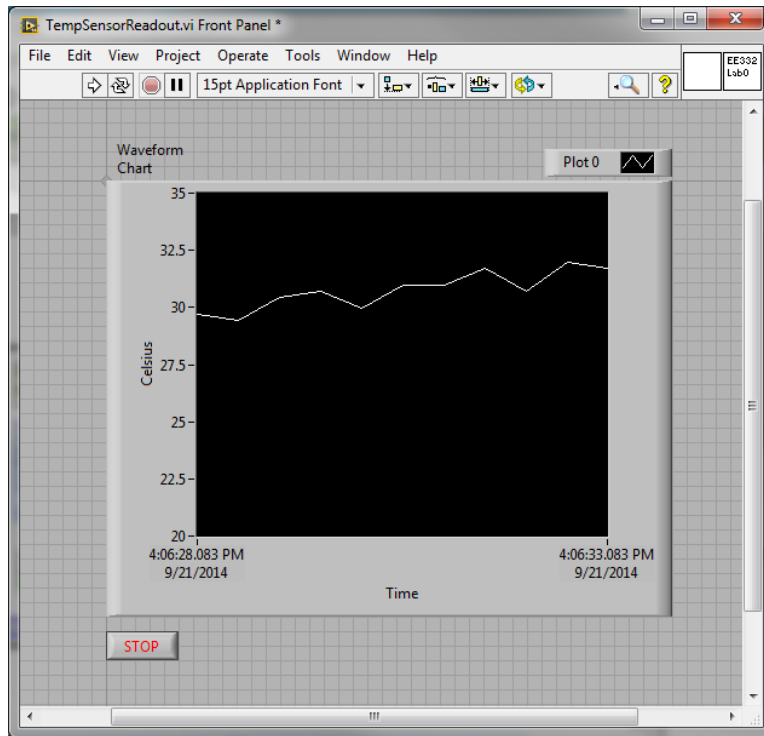


Figure E0.1e

Comment

Generally, it is always best to stop a running VI by using the STOP button that is part of the VI front panel. Using the red stop sign button to stop the VI is a more drastic measure which sometimes leaves LabVIEW in a less predictable state. The red stop sign is really a program abort, which should be used as a last resort.

Question-1

Explain the time dependent behavior of the measured voltage that you observed in the above. Speculate on what factors determine how fast the temperature sensor will react to a change in its case temperature. Suggest some ways of speeding this up, or for slowing it down.

The waveform chart attempts to display the data from the temperature sensor as a “real-time” signal that moves across the display. From the sampling at 500 ms intervals, you should notice some corners in the waveform as the chart connects the data points by straight lines. Describe how the sampling rate affects (or does not affect) the response time of the temperature measurement system.

Procedure 2 Adding a Celsius to Fahrenheit conversion

Comment This next procedure will modify the previous VI to add a Celsius to Fahrenheit conversion, allowing the measured result to be displayed simultaneously on both temperature scales.

Set-up If it is not already running, launch LabVIEW and open the previous TempSensorReadout.vi. Open the block diagram for this VI by clicking on Window > Show Block Diagram, or typing Ctrl+E.

Return to the Front Panel window and resize it to make some room for a second waveform chart. Copy and paste a second waveform chart, either below or to the right of the first. A second waveform chart could have also been added by clicking and dragging from the Modern > Graph section of the Controls Palette, but copying and pasting retains all of the settings of the prior chart, and is more efficient.

Go to the Block Diagram window, and first move the new Waveform Chart 2 to inside of the while loop, placing it above the first Waveform Chart. To convert the temperature in Celsius to Fahrenheit, the data needs to be multiplied by 1.8 and 32 degrees added to it. From the Programming > Numeric section of the Functions Palette, first place a multiply and add operator inside the while loop by clicking on them in the Palette, and then clicking on where they are to be placed in the block diagram. Hover the cursor, which should look like the bobbin wiring tool, over one of the inputs of the multiply operator, right click, and then select Create > Constant. Type “1.8” into the box and this should create the proper multiplication function. Similarly, add a numeric constant of “32” to the add operator. Finally, wire the operators together using the bobbin wiring tool to create the block diagram shown in Fig. E0.2a. Click on a blank area of the block diagram and add a comment “Convert C to F.”

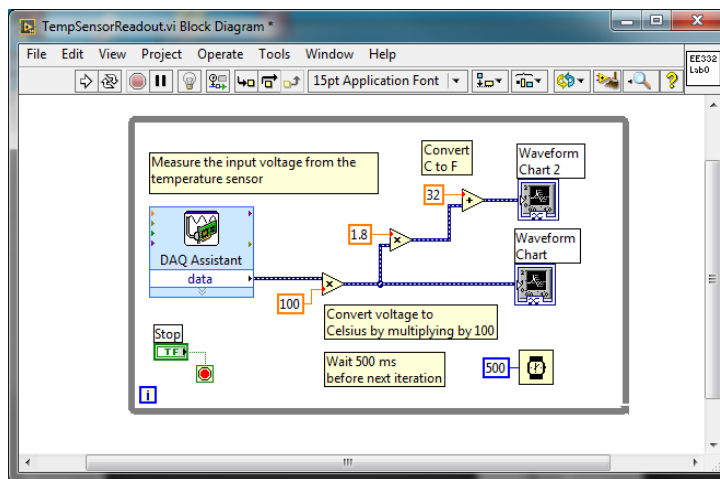


Figure E0.2a

Experiment-0

Switch to the front panel window and make it active. Click on the second Waveform Chart 2 and right click to select Properties. Under the Scales tab, select the Y-Axis from the drop-down menu, and change the Name to “Fahrenheit.” Next, change the minimum and maximum to 60 and 90, respectively. Click on OK, and the chart should now have an updated Y-axis.

Using File > Save As ... , save the modified VI with a new name “Experiment 0 Procedure 2.vi” in your own directory.

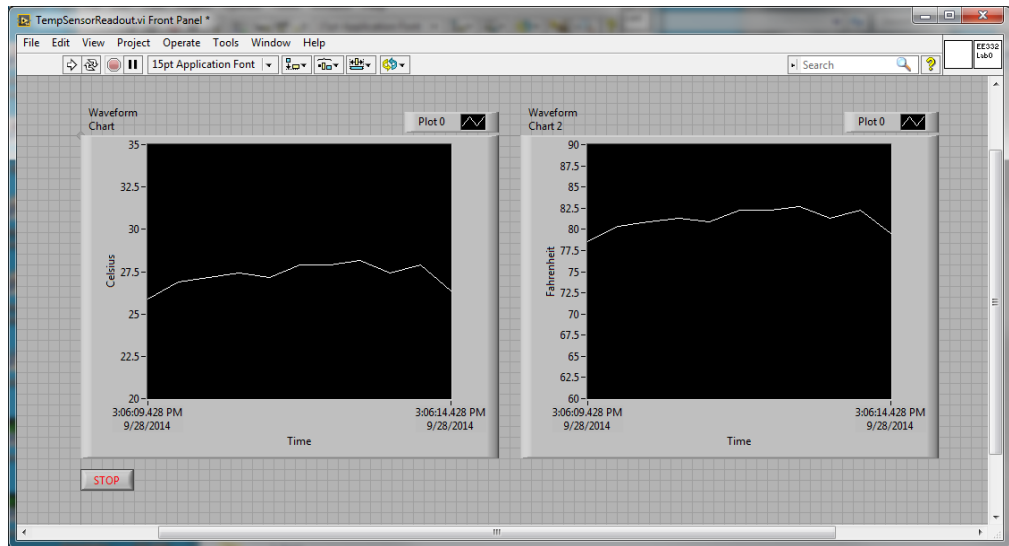


Figure E0.2b

Measurement-2 Switch back again to the front panel window. Click the run button on the front panel and observe that both waveform charts show the correct temperature behavior, similar to that shown in Fig. E0.2b. Click on the STOP button to halt the program.

Question-2 In the block diagram, you might notice that after wiring the multiply and add operators into the data flow path, the wires automatically changed to reflect the proper data structure type. The blue/white wire is termed “dynamic data” and it contains both the floating point values and a time stamp for when the data was collected. A simple orange line is a floating point variable without any time stamp. You might also notice that the multiply and add operators each have a small red dot where the floating point constant enters. Suggest what that red dot indicates and what it does in a programming sense.

Procedure 3 Saving measurement results to spreadsheet files

Comment This next procedure will modify the temperature measurement VI once more to add the capability to store the measurement results in a spreadsheet file.

Set-Up If it is not already running, launch LabVIEW and open the previous Experiment 0 Procedure 2.vi. Open the block diagram for this VI by clicking on Window > Show Block Diagram, or typing Ctrl+E.

From the Functions Palette, select All Functions > Programming > File I/O > Write To Spreadsheet File, and use the mouse to drop this function into the block diagram to the right of, and outside of, the while loop. Change the cursor to the bobbin wiring tool, hover over the terminal at the bottom of the Write to Spreadsheet File labelled “transpose?”, right click, and select create constant. A Boolean False constant (the default) will then appear connected to this terminal. Use the finger tool to click on the Boolean constant and change it to a True value. This transpose operation will make the final data in the spreadsheet appear as columns instead of rows.

Next, from the Functions Palette, select All Functions > Express > Signal Manipulation > From DDT, and drop the dynamic data type converter into the block diagram, inside the while loop. A window will open allowing you to specify what type of data type should be the result of the conversion. Select single scalar value. Use the wiring tool to connect the input of the data converter to the blue/white signal line that runs into the first waveform chart. Use the wiring tool to connect the output of the data converter to the right edge of the while loop. A small box will appear at this termination. Use the wiring tool to click on this small box and extend the wire to the 1D Data input of the Write To Spreadsheet File function. Right click on the small box to pop up the options, and select Tunnel Mode > Indexing. This should make the small box appear with braces ([]) inside it. The orange wire inside the while loop should be a thin one, indicating a simple double precision value, and the orange wire outside the while loop should be a thick one, indicating that the output from the while loop is now an array. Once the STOP button is pressed, the while loop will end and all of the measurements that have been taken up to this point will then be passed to the Write To Spreadsheet File function as one array of values.

Using File > Save As ... , save the modified VI with a new name “Experiment 0 Procedure 3.vi” in your own directory.

The block diagram should now look like that shown in Fig. E0.3a below.

Experiment-0

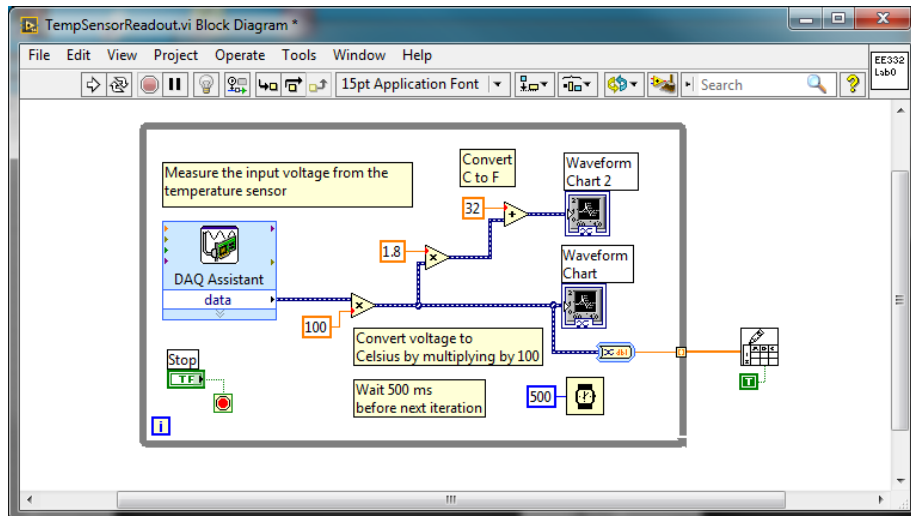


Figure E0.3a

Measurement-3 Make the front panel window for the new modified VI active and click on the run button to start its operation. The Celsius and Fahrenheit waveform charts should both begin displaying the running temperature data. After a few minutes, click on the STOP button. A Save As ... dialog window will open in which you can specify the location to where the new Excel (.xls) file will be written. Enter a new file name, such as "Experiment 0 Procedure 3.xls" and click on "OK." After the file has been written, use Excel to open this file and verify that the correct data values have been written there in the first column of the worksheet. You might also create a plot within Excel and compare this to what you saw displayed on the waveform chart for the Fahrenheit temperatures.

Question-3 If the new VI is kept running for 5 minutes, how many Celsius temperature readings will be stored in the spreadsheet file? Suggest how you might modify the existing VI so that both Fahrenheit and Celsius readings can be stored in the spreadsheet as two columns. You do not need to construct this new VI, simply suggest how the VI needs to be modified to accomplish this.

Procedure 4 Measurement of diode I-V characteristics using LabVIEW

Comment This procedure is the same as that performed for the EE-331 lab. It is included here as a refresher on taking I-V measurements, and provides a good starting point for the measurements of bipolar junction transistors that will follow in Experiment 1.

Computer-controlled automatic measurements are commonly used to gather data for the purpose of characterizing or testing a device or system. In this experiment, a LabVIEW curve tracer will be used to capture the characteristic I-V curve for a pn-junction diode. This procedure will also use the NI-USB-6009 DAQ to both create the excitation voltages and to measure the resulting test voltage responses. No other external bench instruments are needed other than the computer, a USB cable, the NI-USB-6009 DAQ, a superstrip solderless breadboard, and a few solid jumper wires to connect the DAQ to the superstrip.

The NI-USB-6009 DAQ does have one awkward shortcoming which is that the analog output can only produce voltages in the range of 0.0 to +5.0 Volts. This limits the DAQ curve tracer to only being capable of tracing the forward characteristics. The reverse characteristics can be obtained by reversing the diode polarity in the test circuit.

Set-Up First, launch LabVIEW. Click on Open Existing File and navigate to the EE-332 LabVIEW VIs directory. Open the VI named “DiodeCurveTracer.vi.” For this VI to open correctly, three sub-VIs named “DiodeStepGenerator.vi,” “DiodeMeasurement.vi” and “RemoveArrayDuplicates.vi” must also exist in the same directory as “DiodeCurveTracer.vi.”

The front panel window is shown in Fig. E0.4a below.

Experiment-0

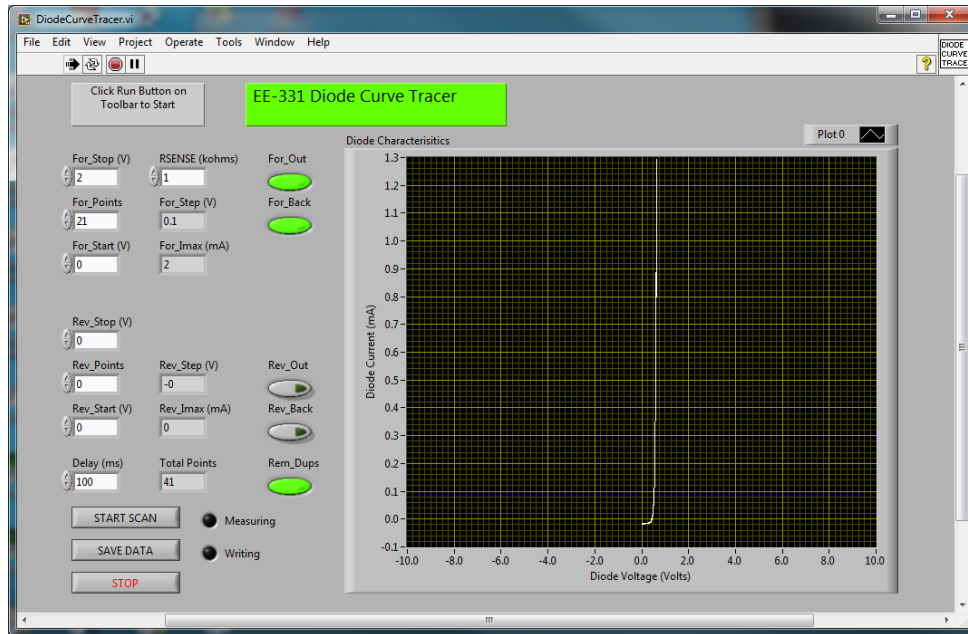


Figure E0.4a

This diode curve tracer has been designed to allow different excitation voltage scans in the forward and reverse directions. The forward and reverse parts of the scan are set up independently according to their initial value (Start), their final value (Stop), and the number of points used for each (Points). Positive start and stop values are used for the forward scan, while negative start and stop values are used for the reverse scan. After these values are entered, the VI computes the voltage increment which is added to go from one point to the next (Step). The delay between when a new excitation voltage is output and when the device response is measured is entered in milliseconds in the Delay input box. Usually a delay of 50-100 ms gives the device plenty of time to stabilize between measurement points.

When the START SCAN button is clicked, this sequence of excitation voltages is passed to the analog output on the DAQ (AO-0) which first steps out (upward) in the forward direction, then back down to zero, then steps out (downward) in the reverse direction, and then back up to zero, making one complete cycle through the applied bias range for which the diode is to be tested. Each of these four segments can be independently included or excluded from the scan using the four green pushbutton switches on the front panel (For_Out, For_Back, Rev_Out, Rev_Back). When the pushbutton is illuminated in green, that part of the cycle will be included in the scan. When these different segments of the scan are concatenated, some duplicate voltage points will be generated. Clicking on the Rem_Dups button will remove these duplicate points from the scan when it is enabled in its illuminated green state. The total number of excitation voltage points in the scan is displayed in the box at the bottom of the front panel.

Experiment-0

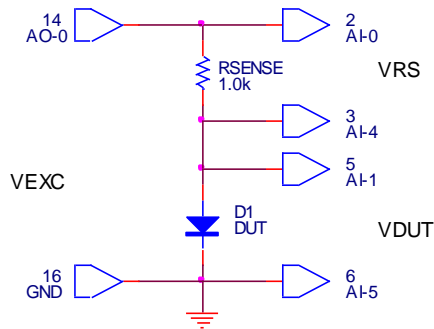


Figure E0.5b

The excitation voltage is applied across a series combination of the device under test (DUT) and a current sensing resistor (RSENSE), as shown in Fig. E0.5b. Thus, $V_{EXC} = V_{RS} + V_{DUT}$, as shown in the schematic. The value of RSENSE is entered into the box at the top of the front panel in units of k Ω . This value is used to calibrate the vertical axis of the diode characteristics graph, and it is also used to compute the maximum current in milliamperes that can flow through the device under test, based upon the forward and reverse stop values (For_Imax, Rev_Imax).

The cathode end of the diode under test (the end with the bar) is grounded, so that when the excitation voltage is positive, a positive current flows downward through the current sense resistor and the diode in the conventional direction. The voltage across the diode (VDUT) is used to create the x-values for the I-V curves, and the y-values of diode current (IDUT) are obtained by dividing the voltage across the sense resistor (VRS) by the value of the resistor. This is typically how one accomplishes current sampling with a data acquisition system. The diode voltage and current are then plotted as (x,y) pairs in the chart. After the scan is complete, the SAVE DATA button can be clicked to write the data to a spreadsheet file. A window will pop open allowing the user to specify the filename for the data to be written into. The output spreadsheet file will consist of four columns of data with one row for each excitation voltage. The columns are: VEXC, VRS, VDUT, and $IDUT = VRS/RSENSE$.

The excitation output (VEXC) and the two measured voltages (VRS, VDUT) are implemented through channels on the NI-USB-6009 DAQ. Analog Output channel # 0 (AO-0) is used to create the excitation output voltage, using terminals AO-0 (#14) and GND (#16) on the DAQ analog connector block. The diode voltage VDUT is measured by Analog Input channel # 1 (AI-1), which is set up as a differential input using terminals AI-1 (#5) and AI-5 (#6) on the connector block. Similarly, the voltage across the current sensing resistor (VRS) is measured by Analog Input channel # 0 (AI-0), which is also set up as a differential input using terminals AI-0 (#2) and AI-4 (#3) on the connector block.

Experiment-0

The device under test (DUT) and current sensing resistor (RSENSE) are inserted into a superstrip solderless breadboard, and connections from these tie points to the DAQ analog screw terminal connector block are achieved with lengths of solid insulated hookup wire, as shown in Fig. E0.5c below.

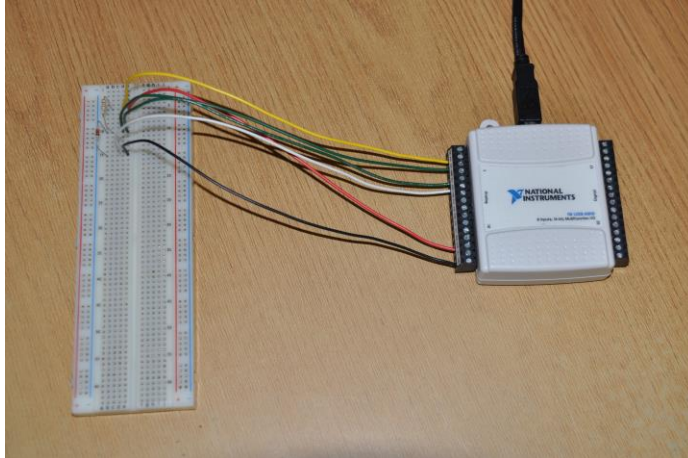
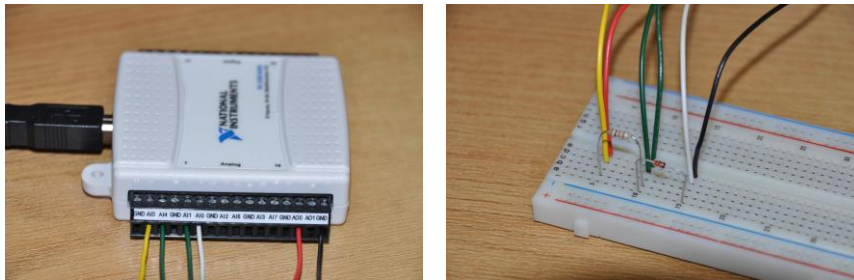


Figure E0.5c

Details of the connections to the DAQ and to the superstrip are shown below in Figs. E0.5d and E0.5e:



Figures E0.5d,e

For this procedure, use a current sensing resistor of $R_{SENSE} = 1.0 \text{ k}\Omega$ and a type 1N4148 test diode, as shown in Fig. E0.5e.

Measurement-5 From the front panel window, click on the run button to start the diode curve tracer VI. Use the following settings for the forward and reverse bias scan ranges: forward bias: 21 points from 0.0 Volts to +2.0 Volts, and reverse bias: 0 points from 0.0 Volts to 0.0 Volts, since the NI-USB-6009 DAQ analog output cannot create negative output voltages. Enable forward out and forward back segments of the scan, and remove the duplicate points for a total of 41 points. Enter a delay of 100 ms and an RSENSE value of 1 kohms. After rechecking all of the connections, click on the START SCAN button, which should start the measurement sequence and then display the resulting diode I-V characteristics on the x-y chart, similar to those shown in Fig. E0.5a.

Experiment-0

Once you are happy with the measurement, click on the SAVE DATA button to save the measured diode I-V characteristic data in an Excel spreadsheet format. A Save As ... dialog window will open, and you can type in the name of the file that you want the data written to, for example, "Experiment 1 Procedure 4.xls." Click on OK to write the file. Once you have saved the data, click on the STOP button to halt the measurement VI.

You might open this newly created file with Excel to verify that the data was properly written to the file. If everything was working properly, the first column should show the sequence of excitation voltages (VEXC) in units of Volts, the second column should show the voltage across the current sensing resistor (VRS) in units of Volts, the third column should show the voltage across the diode (VDUT) in units of Volts, and the fourth column should show the current through the diode (IDUT) in units of milliamperes, computed as $IDUT = VRS/RSENSE$. If you were to create an x-y graph in Excel using the third and fourth columns, you should obtain the same graph as which is shown on the front panel of the VI.

Comment It is generally a good idea to halt any running VI when you are done with it. If you wish to use other Windows programs, such as Excel, or Internet Explorer, or Windows Explorer, you will find that these programs will run rather slowly while any VIs are running at the same time.

Question-5 (a) If the diode were reversed in its polarity (connecting its anode to ground), what would be the expected I-V curve?
(b) If the diode were replaced by another 1.0 k Ω resistor, what would be the expected I-V curve? What would the slope of the resulting I-V curve correspond to?

Comment You might wish to view the internal structure of the diode curve tracer by opening the block diagram window for this VI. This is a fairly complicated VI that uses a number of control structures and employs two other sub-VIs: DiodeStepGenerator.vi and DiodeMeasurement.vi. The DiodeStepGenerator.vi calls yet another sub-VI, RemoveArrayDuplicates.vi. You should try to locate these in the block diagram. If you double click on either of these sub-VIs, they will open and you will be able to then view their internal structure from their block diagrams. Try this, and open the block diagram for DiodeMeasurement.vi. This sub-VI consists simply of a flat sequence control structure whose borders look like a piece of photographic film. This consists of 4 frames [0...3] which are executed in sequence. By clicking on the left and right arrows at the top of the film boundary, you can sequence through the 4 frames. In this case, the #0 frame sends the excitation voltage VEXC to the DAQ card which then outputs it as Analog Output channel # 0 (AO-0). In frame #1, the system waits for a specified delay (in ms) to allow the effects of this new excitation voltage to the diode and resistor

Experiment-0

to settle out. In frame #2, the DAQ card reads Analog Input channel # 0 (AI-0) and sends this measurement value out as VRS. In frame #3, the DAQ card reads Analog Input channel # 1 (AI-1) and sends this measurement value out as VDUT. This sequence of 4 frames is executed each time for each new value of the excitation voltage and is fairly typical of the core of an automated measurement procedure.