Labview Exercises

for Labview 7.0

Installation of Labview:

1. Install the Labview 7.0 software and drivers onto your computer. These files can be found by mapping a network drive to \\poohbah\labview, and by running the 'autorun' file in the 'Labview 7' folder. The serial number for the program is in a word document in this folder. When asked for the location of the Labview drivers during the installation, direct the install program to the 'Labview 7 Drivers' folder.

2. Copy the files LVBASICS.LLB and CEM834.llb to the Program Files/National Instruments/Labview 7.0/user.lib folder on your computer hard drive. These files can be also found at \\poohbah\labview.

3. Change the name of the CEM834.llb file to Yourname.llb. (Highlight the file and then click on the name. You can then type in a new name.) **IT IS VERY IMPORTANT** to have the .llb extension.

4. Remove the "Read-only" restriction on Yourname.llb. (Highlight the file again and select Properties from the File menu. Deselect Read-only.)

5. Start Labview by clicking the LabVIEW icon in the Program Files/National Instruments/Labview 7.0 folder.

Getting Started:

NOTE:

In order to have the Advanced Palettes available when using Labview, click on New, then double click on blank.vi. In the Front Panel window choose Tools > Options. Choose Control/Function Palettes from the menu. If necessary, change Palette View to Advanced and click OK.

All of the VI's that you will use and create in this lesson will be within your "library" file: Yourname.llb. At the end of the Labview lessons, you will give a copy of your library file to me for grading.

IMPORTANT TERMS:

Pop-up: means select with the RIGHT mouse button.

Click on: means select with the LEFT mouse button.

HINTS (they'll make more sense once you get started)**:**

- It is useful to have the "Context Help" window activated while you are learning Labview. To do this, select **Show Context Help** from the Help menu on the toolbar. Whenever the curser is on an icon, the Help window will display a description of the function and show and identify its terminals.

- You do not have to have the Control and Function palettes on your desktop at all times. The Control palette is displayed if you pop-up on the Front Panel, and the Function palette is displayed if you pop-up on the Diagram window.
- You will learn about Debugging VI's in the second exercise. Make a habit of using these tools to debug the VI's in these exercises – they'll save you lots of frustration when your VI doesn't work!

Labview Exercise List

Name: _____

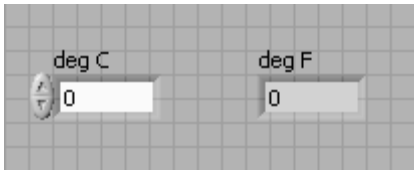
	TA Initials
Convert Celcius to Farenheit VI	
Debug Exercise VI	
Creating an Icon and Connector Exercise VI	
Thermometer VI	
Temperature Monitor VI	
Random Signal VI	
Auto Match VI	
Shift Register Example VI	
Temperature Running Average VI	
Random Average VI	
Array Exercise VI	
Graph Waveform Array VI	
Temperature Analysis VI	
Square Root VI	
Temperature Control VI	
Time to Match VI	
Formula Node VI	
Build String VI	
File Writer VI	
File Reader VI	
Temperature Logger VI	
Temperature Application VI (Optional)	
Spreadsheet Example VI	
Temperature System VI (Optional)	
Serial Write and Read VI	
GPIB Write and Read VI	
Build a Command String VI	

Convert Celcius to Farenheit VI

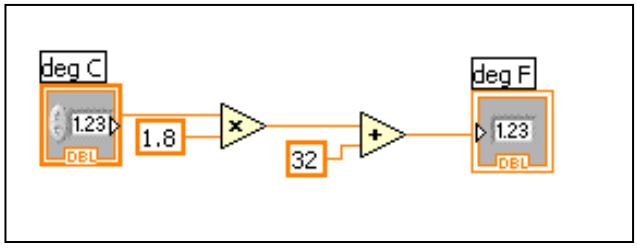
Objective: Learn to build, save, and run a VI.

You will create a VI that takes a number representing degrees Centigrade and converts it to a number representing degrees Fahrenheit.

1. Open a new front panel by selecting the **New VI** button from the startup window. Double click on "Blank VI" to open the window.
2. View the front panel and block diagram windows side by side by choosing **Tile Left and Right** from the **Window** menu.
3. You will now build the front panel and block diagram shown below.



Front Panel



Block Diagram

A. CREATE THE FRONT PANEL

1. Create the **Numeric Control**. You will use this control to enter the value for degrees Centigrade.

   When the left (front panel) window is active, the Tools palette and the Controls palette should be visible. To make them visible, select **Show Tools Palette** and/or **Show Controls Palette** from the **Windows** menu. (Alternatively, you can choose not to show the controls palette and right click on the control panel window to bring it up only when you need it. This also works for the Functions palette which will be discussed in the next section.).

   NOTE: "Pop-up" means click the right mouse button.

   a. Select **Numeric Control** from the **Numeric** subpalette of the **Controls** palette.

   b. Drag the Control to where you want it and release the mouse button.

   c. Type deg C inside the label and press the **Enter Button** (check mark) on the **Toolbar**. If you want to change the label text in the future, select the **Text Tool** on the **Tools Palette** and use it to highlight the text. Then type in the desired text in the box and press the **Enter Button**. You can make the label appear or disappear by popping up on the **Numeric Control** and selecting or deselecting **Label** from the **Visible Items** menu.

2. Create the **Numeric Indicator**. You will use this indicator to display the result from a calculation to convert temperature from degrees Centigrade to degrees Fahrenheit.

a. Select **Numeric Indicator** from the **Numeric** subpalette of the **Controls** palette. If the Controls palette is not visible, pop-up in an open area of the Panel window.

b. Drag the indicator to where you want it and then release the mouse button.

c. Type deg F inside the label and click outside the label when finished.

NOTE THAT:

- Each time you create a new control or indicator, LabVIEW automatically creates the corresponding terminal in the Diagram window. The terminal symbols suggest the data type of the control and indicator. For example, a DBL terminal represents a double-precision floating-point number.
- A control terminal has a thicker border than an indicator terminal on the block diagram.

B. CREATE THE BLOCK DIAGRAM

1. Make the Diagram window the active window by clicking anywhere on it.

   When the right (block diagram) window is active, the Tools palette and the Functions palette should be visible. To make them visible, select **Show Tools Palette** and/or **Show Functions Palette** from the **Windows** menu.

2. Select the **Multiply** and **Add** functions one at a time from the **Numeric** subpalette of the **Functions** palette and place them in the diagram window.

3. You can activate the help window by choosing **Show Context Help** from the **Help** menu. Placing any of the editing tools on a node displays the inputs and outputs of the function in the Help window when the diagram window is active. Use this to look at the inputs and outputs of the multiply and add function icons.

4. Select the two numeric constants one at a time from the **Numeric** subpalette of the **Functions** palette. When you first place the numeric constant on the Diagram window, it is highlighted so you can type a value into it. Type 1.8 into one constant and 32.0 into the other one.

   If you moved the constants before you typed a value into them, you can use the **Text Tool** (**Tools** palette) to enter the values.

5. Using the **Wiring Tool** from the **Tools** palette, wire the icons as shown in the block diagram above.

   To wire from one terminal to another, click the Wiring tool on the first terminal, move the tool to the second terminal, and click on the second terminal. It does not matter at which terminal you start.

   To aid in wiring:

   - Tack down wires by clicking on the diagram.
   - Pop-up on the **Multiply** and **Add** functions and choose **Visible Items » Terminals**. Return to the icons after wiring by popping-up on the functions, choosing **Show Panel**, and unchecking **Terminals** from the menu.

C. SAVE THE VI.

a. Select **Save** from the **File** menu. Select your VI library **Z:\YOURNAME.LLB** (it's on the Z: drive).

b. A dialog box will open. Type in the name of the VI: Convert C to F.vi

c. Click on OK. This will save the VI to your library.

D. RUN THE VI.

a. Using the **Operate Tool**, double-click in the **Numeric Control** and type in a new number.

b. Run the VI by clicking on the **Run Arrow** on the **Toolbar**.

c. Try several different numbers.

E. CLOSE THE VI.

Debug Exercise VI

Objective: To become familiar with LabVIEW debugging features.

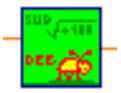
You will load a nonexecutable VI and correct the error. You also will use the single-step and execution highlighting modes to step through the VI.

1. Open the Debug Exercise VI by choosing Open from the File menu, and selecting your VI library: Z:\Yourname.LLB. Double click on Debug Exercise (Main) VI. View the front panel and block diagram windows side by side by choosing **Tile Left and Right** from the **Window** menu.
2. Notice the broken Run button in the Toolbar, indicating the VI is not executable. The following icons are shown the block diagram window:

Random Number (0-1) function (**Numeric** subpalette). This function returns a random number between zero and one.

Multiply function (**Numeric** subpalette). In this exercise, this function multiplies the random number by 10.0.

Numeric Constant (**Numeric** subpalette). This constant specifies the constant in the block diagram.



Debug Exercise (Sub) VI. This VI adds 100 and then calculates the square root of the value.

3. To find the object reporting the error:
    a. Click on the broken Run button. A dialog box listing one error will appear.
    b. Double click on the error in the dialog box. In the block diagram, a dashed line highlights the **Multiply** function. The **Multiply** function contains an unwired terminal.
4. Wire the numeric constant (10.0) to the lower-left terminal of the **Multiply** function. If you correctly wired the numeric constant, the arrow symbol in the Run button looks normal.
5. Save the VI.
6. Run the VI several times by clicking on the Run button.

A good way to debug a VI is to single-step through the VI and animate the flow of data through the block diagram. As data passes from one node to another, the movement of data is marked by bubbles moving along the wires. In addition, in single stepping, the next node to be executed blinks rapidly.

7. In the Diagram window enable the execution-highlighting mode by clicking on the execution highlighting button (light bulb) on the Toolbar.
8. Enable the single-step mode by clicking on the **Step Into** button or the **Step Over** button on the Toolbar (initially, both of these buttons are labelled **Start Single Stepping**). You will see the data flow from the numeric constant to the input of the **Multiply** function, and the **Random Number** generator function blinks rapidly.
9. The Run button becomes black to indicate the VI is running.
    a. Step through the entire block diagram by clicking on the **Step Over** button after each node. By clicking the **Step Over** button, you will execute the current node and pause at the next node, which is ready to execute.
    b. When the outline of the Debug Exercise (Sub) VI blinks, click on the **Step Out** button to complete execution of the Debug Exercise (Sub) VI.
    c. When the outline of the whole block diagram blinks, click on the **Step Out** button to complete execution of the Debug Exercise (Main) VI.

    Notice that the data appears on the front panel as you step through the program. First, the VI generates the random number and then multiplies it by 10.0. Finally, the subVI adds 100.0 and takes the square root of the multiplication result.
10. Single step through the VI again, but this time you also will single step through the Debug Exercise (Sub) VI subVI.
    a. Activate the Debug Exercise (Main) VI Diagram window and begin single stepping by clicking on the **Step Into** button or the **Step Over** button.
    b. Click on the **Step Into** button when the Debug Exercise (Sub) VI is blinking. The diagram for the subroutine is displayed on top of the calling VI.
    c. Click on the Debug Exercise (Main) VI (calling VI) Diagram window to activate the window and notice the green arrow on the subVI icon, depicting it in single-step mode.
    d. Click on the Debug Exercise (Sub) VI Diagram window and click on the **Step Out** button twice to complete the subVI block diagram execution and then the subVI execution.

e. The Debug Exercise (Main) VI Diagram window becomes active. Click on the **Step Out** button to complete the VI execution.

LabVIEW also contains a probe to view the data as it flows through a wire.

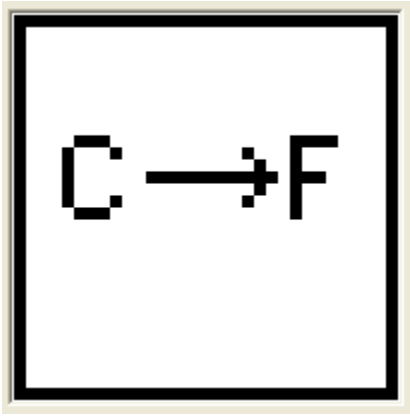
11. Enable the probe by selecting the **Probe** tool from the **Tools** palette and then clicking on any object.
12. Single step through the VI again. (Remember: when the sub VI is executing, press the **Step Over** button *in* the sub VI diagram window.) The **Probe** windows should display the data as it flows through that segment.
13. LabVIEW can halt execution of a VI at any location on its diagram.
14. Set breakpoints by selecting the **Breakpoint** tool from the **Tools** palette and clicking anywhere in the circuit.
15. Run the VI by clicking on the Run button. The VI will pause at the breakpoints. To continue VI execution, click on the **Pause/Continue** button. To clear breakpoints, click on the set breakpoints with the **Breakpoint** tool.
16. Turn off execution highlighting by clicking on it.
17. Close the VI and all open windows by selecting Close from the File menu.

Creating an Icon and Connector VI

Objective: To learn to create your own VI icons. These icons are essentially sub-VI's (subroutines) that can be used within other VI's. In order to be used, icons must have terminals for wiring them into a VI. In this lesson, you will create an icon for the Convert C to F VI and assign the terminals of the icon to the digital control and indicator. You will use this VI again later.

Creating an Icon and Connector - Front Panel

1. Open the Convert C to F VI.
2. Open the Icon Editor by popping up in the **Icon Pane** (upper right corner) and choosing **Edit Icon** from the pop-up menu.
3. Erase the default icon by double-clicking on the **Select** tool (broken rectangle) and pressing <delete>. Redraw the border by double-clicking on the **Rectangle** tool (unbroken rectangle).
4. Create the icon shown on the top of the next page. Create the text with the **Text** tool (A). Double-click on the **Text** tool to change the font. Create the arrow using the **Pencil** tool. Shift-dragging (holding down <shift> while dragging the mouse) with the Pencil tool draws horizontal or vertical straight lines.



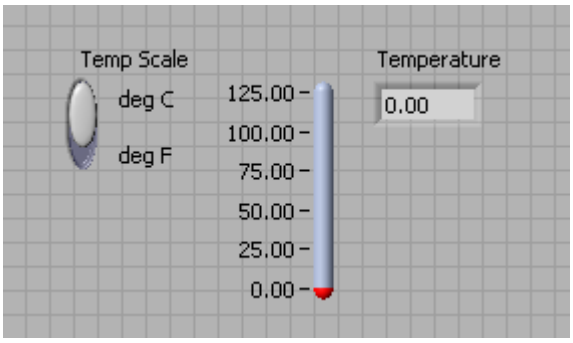
5. Close the Icon Editor by clicking on OK when your icon is complete. The icon appears in the Icon Pane in the upper-right corner of the Panel or Diagram window. The icon represents the VI in the block diagram of other VIs. An icon can be a pictorial representation of the VI's purpose, or it can be a textual description of the VI or its terminals.
6. In the Front Panel window (note: this will not work in the Diagram window), define the connector terminal pattern by popping up in the **Icon pane** and choosing **Show Connector** from the pop-up menu .
7. LabVIEW will select a terminal pattern based on the number of controls and indicators on the front panel. In this example, there are two terminals—the deg C digital control and the deg F digital indicator.
8. Assign the terminals to the digital control and digital indicator using the **Wiring** tool. Click on the left terminal in the connector and then on the deg C control. The left terminal should turn a dark orange color. Now click on the right terminal in the connector and then on the deg F indicator. Now the right terminal will turn dark orange.
9. When you click in an open area of the panel, both terminals on the connector should be orange. This shows that both terminals are connected to floating-point values.
10. A common LabVIEW convention is that the terminals connected to front panel controls are located at the left side of the connector pane, while the terminals connected to front panel indicators are located at the right side of the connector pane. In other words, your input terminals are at the left on the connector pane, and your output terminals are at the right on the connector pane.
11. Save the VI under the same name.
12. Close the VI.

Thermometer VI

Objective: You will build a VI that measures temperature using the temperature sensor on the DAQ Signal Accessory. The sensor outputs a voltage proportional to temperature. For example, if the temperature is 23° C, the sensor output voltage is 0.23 V. The VI also will have the option to display the temperature in degrees Fahrenheit rather than degrees Celsius.

NOTE: Use the (Demo) Read Voltage VI instead of Read Voltage VI. This VI simulates a temperature sensor.

A. Begin by building this front panel (instructions follow).



1. Open a new panel by selecting New from the File menu.
2. Place the thermometer indicator in the Panel window.
   a. Pop-up in an open area of the Panel window and choose Thermometer from the pop-up Numeric subpalette.
   b. Type Temperature inside the highlighted text box and click the mouse button or the Enter button on the toolbar.
   c. Pop-up on the Thermometer and select **Visible Items >>Digital Display**.
3. Rescale the thermometer control to display the temperature between 0 and 125. Using the Labeling tool, double-click on 100 in thermometer scale, type 125, and click the enter button on the toolbar. Change the precision of the displayed temperatures by popping up on one of the numbers and selecting Format & Precision from the dropdown menu. Choose Automatic formatting and change 6 and Significant figures to 2 and Digits of precision, respectively. The precision of the Digital Display can also be changed from this dialog box by selecting Digital Display 0 instead of Scale in the dropdown menu in the upper left corner.
4. Place the vertical slide switch in the Panel window.
   a. Pop-up in an open area of the Panel window and choose Vertical Slide Switch from the pop-up Boolean subpalette. Type Temp Scale inside the text box and click the mouse button or the enter button on the toolbar.
   b. Using the Labeling tool, place a free label, deg C, next to the true condition of the switch. Place a free label, deg F, next to the false condition of the switch.

B. Create on-line documentation for all front panel objects and for the on-line help.

1. Document the VI by choosing **VI Properties…** from the **File** menu. Select Documentation from Category menu. Type the description of the VI in the dialog box:

   "This VI measures temperature using the temperature sensor simulator."

   Click on OK. You can recall the description by again choosing **VI Properties…** from the **File** menu.

2. You can also document the objects on the front panel (or their respective terminals on the block diagram) by popping up on the object and choosing **Description and Tip** from the object pop-up menu. Document the thermometer indicator and switch control.

   a. Pop-up on the thermometer indicator and choose **Description and Tip** from the pop-up menu.

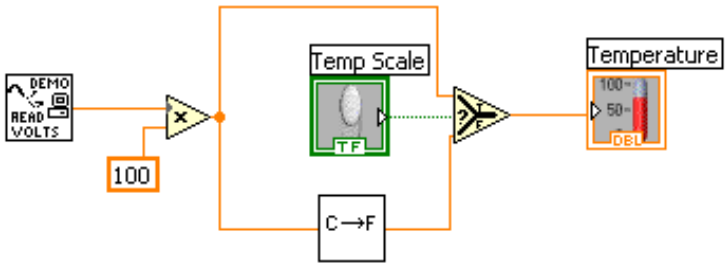
   b. Type the description: "Displays the temperature measurement." and click OK.

   c. Pop up on the vertical switch control and choose **Description and Tip** from the pop-up menu.

   d. Type the description: "Determines the scale (Fahrenheit or Celsius) to use for the temperature measurement." and click OK.

3. Show the descriptions you created by again selecting **Description and Tip** from the indicator and control pop-up menu.

C. Now build this block diagram using the instructions that follow.



1. Select the block diagram objects.

(Demo) Read Voltage VI (Functions Palette » User Libraries » Basics Course subpalette). This VI simulates the Read Voltage VI operation.

Numeric Constant (Numeric subpalette). To insert a new value, double-click inside the numeric with the Labeling tool and type the new value.

Multiply function (Numeric subpalette). In this exercise, this function multiplies the voltage that the Read Voltage VI returns by 100.0 to obtain the Celsius temperature.

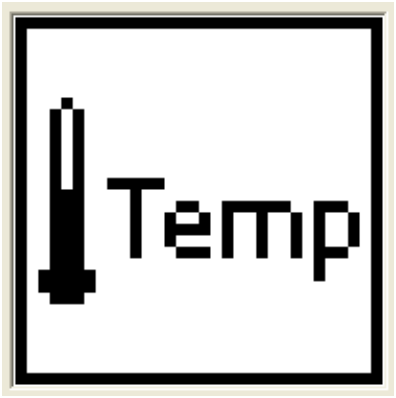
Convert C to F VI (Select a VI… subpalette). Open Yourname.llb and select the VI you wrote: Convert C to F. The icon you created can then be placed in the diagram as a sub-routine.

Select function (Comparison subpalette). Depending on the value of the Temp Scale switch, the function outputs either the Fahrenheit (False) or Celsius (True) temperature value.

2. Using the **Positioning** tool, place the icons as illustrated on the diagram and wire them together with the Wiring tool.

   Remember, if you need to see icon terminals, pop up on the icon and choose **Visible Items >>Terminals** from the pop-up menu. You also can show the Help window by choosing **Show Context Help** from the **Help** menu.

3. Run the VI several times. Place the VI in the free-run mode by clicking on the **Continuous Run** button.
4. Turn off the continuous-run mode by clicking on the **Continuous Run** button.
5. Create the icon as in the previous lesson.
   a. Invoke the Icon Editor and erase the default icon.
   b. Draw an icon that represents the thermometer.
6. Create the Connector as in the previous lesson.
   a. Pop-up on the icon pane in the control panel and choose **Show Connector**.
   b. Assign the terminals to the switch and the thermometer using the wiring tool.



D. Save the VI in your library and name it: Thermometer.vi.

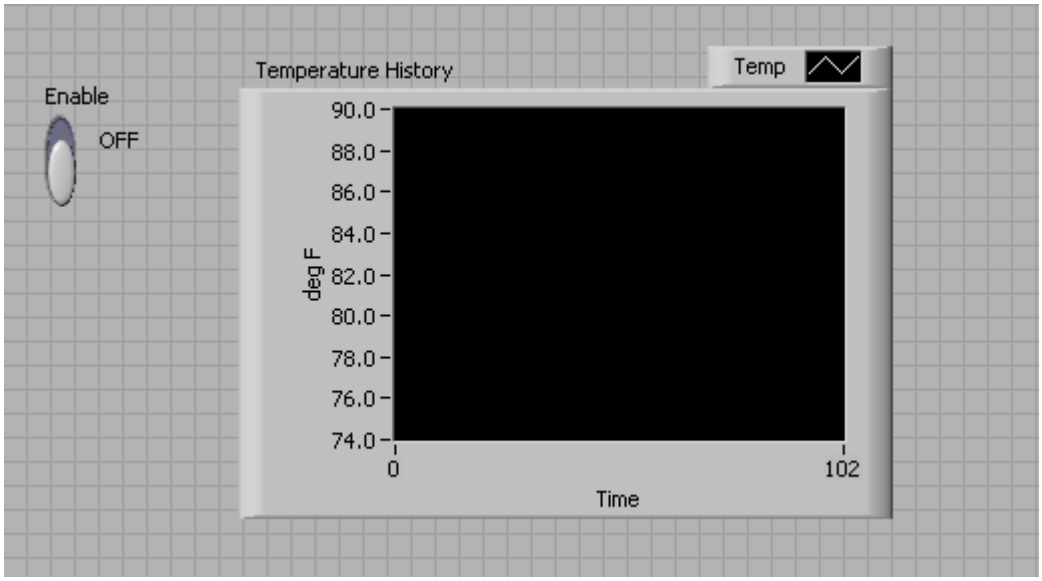
E. This VI is now complete and ready for use as a subVI in other VIs. The icon represents the VI in the block diagram of the calling VI. The connector (with two terminals) outputs the temperature. Close the VI by choosing Close from the File menu.

Temperature Monitor VI

Objective: To build a VI to measure temperature and display it on the waveform chart. This VI will measure the temperature using the Thermometer VI you built in the previous lesson as a subVI.

You will use this VI later, so be sure to save it as the instructions describe.

A. Build the front panel below using the instructions that follow.

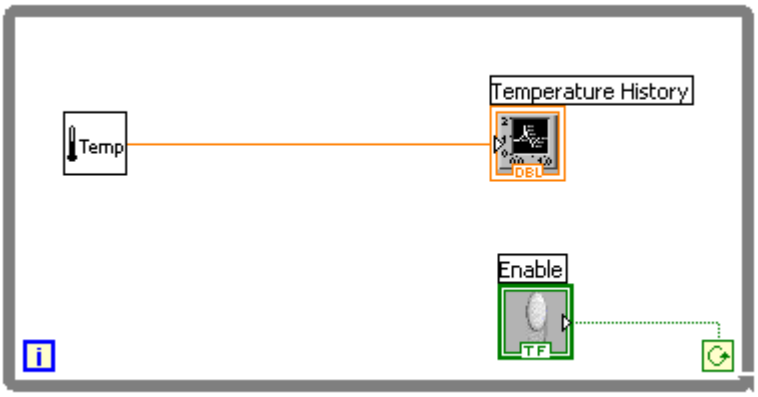


1. Open a new panel and place a vertical slide switch (**Boolean** subpalette) in the Panel window. Label the switch Enable. You will use the switch to stop the acquisition. Popup on the switch and choose Boolean Text from the Visible Items. Use the Positioning tool to move the text to the right of the switch.
2. Place a Waveform *Chart** (**Graph** subpalette) in the Panel window. Label the waveform chart Temperature History. The waveform chart will display the temperature in real time.

   *NOTE: A Waveform *Chart* is different from a Waveform *Graph* – make sure that you select the correct one. A *chart* displays data in real time as it is collected, like a chart recorder. A *graph* displays the results only after the data collection is complete.
3. Because the waveform chart legend labels the plot Plot 0 by default, relabel the legend appropriately. Using the Labeling tool, double-click on Plot 0 in the chart legend, type Temp, and click outside the text area. The click enters the change. You also can select the Enter button in the toolbar to input the change.
4. Because the temperature sensor measures room temperature, rescale the waveform chart to display the temperature. Using the Labeling tool, double-click on 10 in the waveform chart scale, type 90, and either click outside the text area or press <enter>. Change -10 to 70 in the same way.

5. To change the precision of the numbers on the y-axis, pop-up on one of the numbers and select Formatting… Change the formatting from 6 Significant digits to 1 Digits of precision.
6. Change the y-axis label by clicking on it with the Labeling Tool and typing deg F.

B. Create the block diagram using the instructions that follow.



1. Select a **While Loop** from the **Structures** subpalette; then click in the diagram and drag the loop around the enable switch and waveform chart icons. The loop can be enlarged by dragging a corner with the Positioning tool.
2. Wire the "Enable" switch to the Stop icon in the corner of the loop. Popup on the Stop icon and select Continue if True. The icon will now appear as it does in the diagram above.
3. Place your Thermometer VI (**Select a VI...** subpalette) inside the loop and wire it to the temperature monitor.

C. Save the VI in your library. Name it: Temperature Monitor.vi.

D. Modification of the graph format.

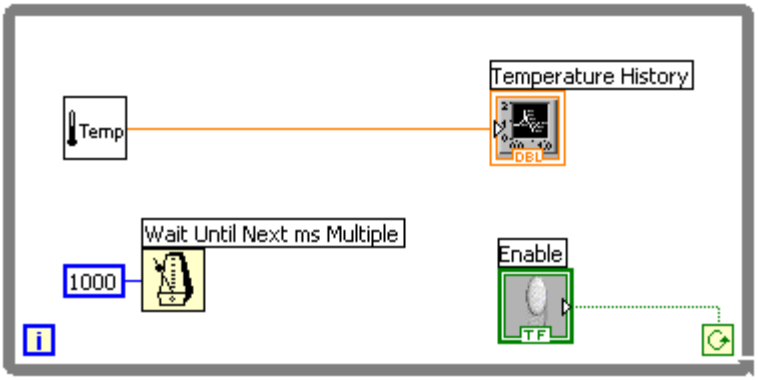
1. Return to the front panel and turn on the vertical switch by clicking on it with the Operating tool. Run the VI.

   The **While Loop** is an indefinite looping structure. The diagram within its border will execute as long as the specified condition is true. In this example, as long as the switch is ON (TRUE), the Thermometer VI will take and return a new measurement and display it on the waveform chart.
2. Stop the acquisition by clicking on the vertical switch. This action causes the loop condition to be FALSE (OFF) and the loop ends.
3. You can format and customize the X and Y scales of the waveform chart to suit your display preferences and data. Pop up on the chart and select **Y Scale » Formatting** from the pop-up menu.

   Experiment with different X- and Y-axis grid options by clicking on the grid style selector and choosing different styles for the axes from the sub-menu that appears. From this window you also can experiment with scale styles, scaling factors, mapping mode, and the format and precision of the axis displays. When you finish exploring these options, return the values to the ones shown above and click on OK or Cancel.

4. Clear the display buffer and reset the waveform chart by popping up on the waveform chart and choose **Data Operations » Clear Chart** from the pop-up menu. If the VI is running, select **Clear Chart** from the pop-up menu.
5. Notice that each time you run the VI, you first must turn on the vertical switch and then click on the Run button.
6. Modify the vertical switch so that you need not turn on the switch each time you run the VI.
   a. Stop the VI if it is running.
   b. Return the vertical switch to the ON position.
   c. Pop-up on the switch and choose **Data Operations » Make Current Value Default** from the pop-up menu. This will make the ON position the default value.
   d. Pop-up on the switch and choose **Mechanical Action » Latch When Pressed** from the pop-up menu.
   e. Run the VI. Click on the vertical switch to stop the acquisition. The switch will move to the OFF position and return to the ON position after the While Loop condition terminal reads the value.
7. Now add timing to the loop so that it takes a data point every second. When you ran the VI, the **While Loop** executed as quickly as possible. However, you may want to take data at certain intervals, such as once per second or once per minute.

   You can control loop timing using the **Wait Until Next ms Multiple** function (**Time & Dialog** subpalette). This function ensures that no iteration is shorter than the specified number of milliseconds.
8. Modify the VI as shown below, to take a temperature measurement once every second.

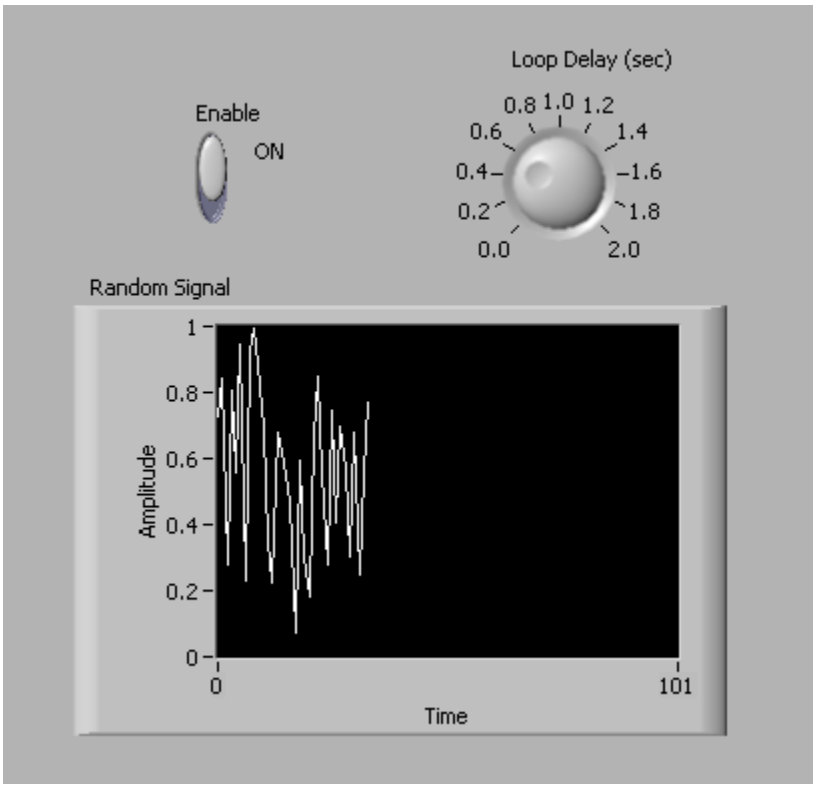


9. The Numeric Constant wired to the **Wait Until Next ms Multiple** function specifies a wait of 1000 ms (one second). Thus, the loop executes once every second.
10. Save the VI again under its current name.
11. Run the VI. Try different values for the number of milliseconds.
12. Close the VI.

Random Signal VI

Objective: To implement timing of a data display by using a numeric control and waveform chart.

Problem: Build a VI that generates random data and displays it on a waveform chart in scope update mode. The VI should have a knob control on the front panel to adjust the **While Loop** rate between 0 and 2 seconds. The panel also should have a switch to stop the VI. You should not need to turn on the switch each time you run the VI. Build a front panel similar to the one below.



Hints:

Hide the waveform chart's legend using the **Visible Items » Legend** option.

Use the **Random Number (0-1)** function (**Numeric** subpalette) to generate the data.

Multiply the knob terminal by 1,000 to convert the seconds to milliseconds. Use this value as the input to the **Wait Until Next ms Multiple** function (**Time & Dialog** subpalette).

When you have completed the VI:

1. Save the VI and name it Random Signal.vi.
2. Run it.
3. Close the VI.

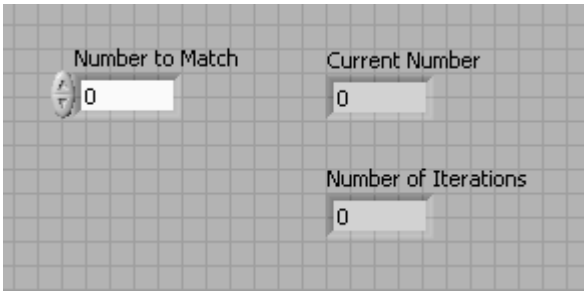
Auto Match VI

Objective: To pass data out of a **While Loop** through a tunnel.

You will build a VI that generates random numbers until the number generated matches the specified number. The loop count terminal keeps track of the number of iterations before a match occurs.

You will use this VI later, so be sure to save your work as Auto Match.vi.

A. Build the Front Panel by creating one digital control and two digital indicators and labeling as shown below.



1. Set the data range on the front panel controls.

   The Data Range option prevents you from setting a value that is not compatible with a preset range or increment. Your options are to ignore the error or coerce it to within range. A solid red border frames the control that is out of range. Set the range between 0 and 100 with an increment of 1 by popping-up on the digital control and choosing **Data Range** from the pop-up menu. Uncheck the Use Default Range box by clicking on it, and fill in the desired Minimum, Maximum, and Increment values.

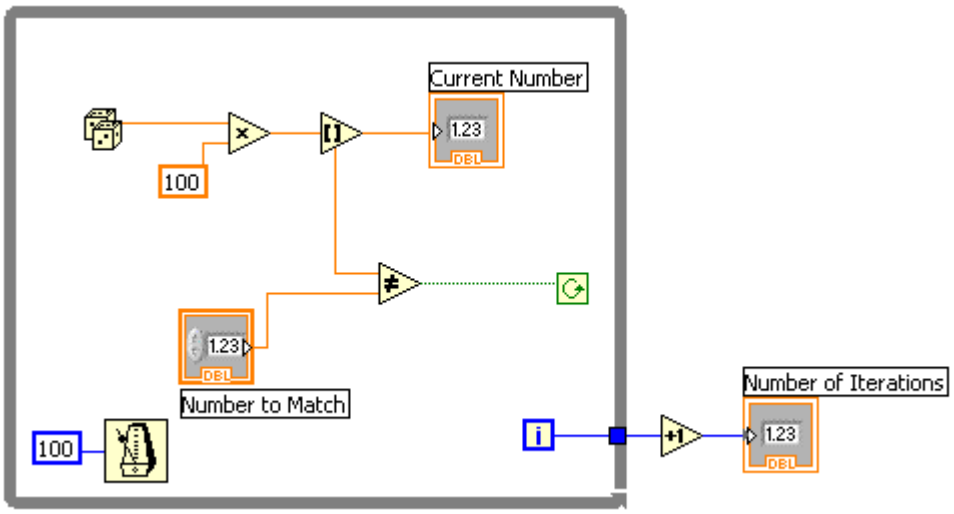
2. Set the precision of the control and indicators on the panel.

   You can use the **Format & Precision** option to change the precision or to display the numeric controls and indicators in scientific, engineering, or hour/minute/second notation. To change the precision to zero:

   a. Pop up on each control and digital indicator and choose **Format & Precision** from the pop-up menu.

   b. Select Floating Point, enter a 0 for Digits of Precision, and click on OK.

B. Create the block diagram to make this VI run.



Some new functions:

Round To Nearest function (**Numeric** subpalette). In this exercise, this function rounds the random number between 0 and 100 to the nearest whole number.

Not Equal? function (**Comparison** subpalette). In this exercise, this function compares the random number with the number specified in the front panel and returns a TRUE if the numbers are not equal; otherwise, it returns a FALSE.

Increment function (**Numeric** subpalette). In this exercise, this function increments the While Loop count by one.

Tunnels

The black square that appears on the **While Loop** border is called a tunnel. Through tunnels, data flows into or out of a looping structure. Data passes out of a loop after the loop terminates. When a tunnel passes data into a loop, the loop executes only after data arrives at the tunnel.

While Loop Operation

The loop in this exercise will execute as long as no match exists. That is, the **Not Equal?** function will return a TRUE as long as the two numbers do not match. Each time the loop executes, the iteration terminal automatically increments by one. The iteration count passes out of the loop upon completion. This value increments by one outside the loop because the count starts at 0.

C. Save the VI. Name it Auto Match.vi.

D. Enter a number in the Number to Match control. Run the VI several times. Change the number and run the VI again.

1. Notice that the Current Number indicator updates at every iteration of the loop because it is inside the loop. The # of iterations indicator updates on completion because it is outside the loop.

2. From the Diagram Window, click on the **Highlight Execution** button to enable execution highlighting. This mode slows down the VI so you can see each number as it is generated.
3. Enter a number that is out of the data range into the Number to Match control. The data range was originally set to between 0 and 100 with an increment of one. Attempt to run the VI. Notice that the VI automatically replaces the number with the value of the closest value within the range.
4. Close the VI.

Shift Register Example VI

Objective: To demonstrate the use of shift registers to access values from previous iterations.

1. Open the Shift Register Example VI and choose **Tile Left and Right** from the **Window** menu.
2. The front panel has four digital indicators. The X(i) indicator will display the current value, which will shift to the left terminal at the beginning of the next iteration. The X(i-1) indicator will display the value one iteration ago, the X(i-2) indicator will display the value two iterations ago, and so on.
3. NOTE: The zero wired to the left terminals initializes the elements of the shift register to zero.
4. Enable execution highlighting by clicking on the **Highlight Execution** button.
5. Run the VI and carefully watch the bubbles. (If the bubbles are moving too fast, use the **Pause** button and **Step Over** button to slow the execution.)
6. Notice that in each iteration of the **While Loop**, the VI "funnels" the previous values through the left terminals of the shift register. Each iteration of the loop adds 5 to the current data, X(i). This value shifts to the left terminal, X(i-1), at the beginning of the next iteration. The values at the left terminal funnel downward through the terminals. In this example, the VI retains only the last three values. To retain more values, more elements can be added to the left terminal of the shift register.
7. Stop the execution by pressing the STOP button on the front panel. Add several more registers by popping up on the registers and selecting Add Element. Each time, a new register will be added.
7. Run the VI again.
8. Save the VI and close it.

Temperature Running Average VI

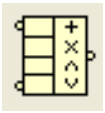
Objective: To use **Shift Registers** to perform a running average.

You will modify the Temperature Monitor VI to average the last three temperature measurements and display the average on a waveform chart.

1. Open the Temperature Monitor VI you created earlier.
2. Use the Save As option to rename the VI as Temperature Running Average.vi.

A. Modify the block diagram to calculate the running average temperature using last three temperature measurements. **NOTE:** Shift registers are created by popping up on the loop.

The following function may be useful in solving this problem.



Compound Arithmetic function (**Numeric** subpalette). This function returns the sums multiple inputs. Click the **Positioning** tool on the blue dot at the bottom of the function and drag to stretch the function into a three-input Add function.

B. Modify the block diagram to display simultaneously the current temperature reading and the running average.

The following function may be useful in solving this problem.

Bundle function (**Cluster** subpalette). This function can be used to "bundle" or group multiple inputs to a waveform chart. You can add additional elements by using the **Positioning** tool and dragging the blue dot on the bottom to resize the function.

C. Customize the waveform chart to match your data display requirements and to display more information. Features available for waveform charts include: a legend, a palette, a digital display, a scroll bar, and a buffer. By default, waveform charts have their legends showing when you first place them on a front panel.

1. If the scrollbar is present, hide it by popping up on the waveform chart and choosing **Visible Items » Scroll Bar** from the pop-up menu.
2. Now customize the Y axis:

   Use the **Labeling** tool to click on the minimum value in the Y scale. Type in 75.0 and press <enter>.

   Again using the **Labeling** tool, click on the second number from the bottom on the Y axis. Change this number to 80.0 or something other than the current number. This number determines the numerical spacing of the Y axis divisions.

   For example, if the number above 75.0 is 77.5, indicating a Y axis division of 2.5, changing the 77.5 to 80.0 will reformat the Y axis to multiples of 5.0 (75.0, 80.0, 85.0...).

   The waveform chart size has a direct effect on the display of axis scales. Increase the waveform chart size if you have trouble customizing the axis.

3. Move the legend and relabel it.

   You may place the legend anywhere relative to the waveform chart. Stretch the legend to include two plots using the Positioning tool. Change "Temp" to "Running Avg" by clicking on the label with the **Labeling** tool and typing in the new text. You can change "Plot 1" to "Current Temp" in the same way. If the text disappears, enlarge the legend

text box by resizing from the left corner of the legend with the **Positioning** tool. (The **Positioning** tool will change to a frame corner when you can resize the legend.)

You can set the plot line style and the point style by popping up on the plot in the legend. You also can color the traces by popping up on the legend and choosing the Color menu.

4. Run the VI. While the VI is running, use the buttons from the palette to modify the waveform chart. The single fit buttons activate X and Y axis autoscaling. The scale format buttons reformat the X and Y axis scale markers. The zoom button provides options for zooming in on specified sections of the chart or on the whole chart. The pan button allows click-and-drag scrolling (panning) in the chart. The return to standard mode button deactivates panning and zooming and returns the mouse to standard mode.

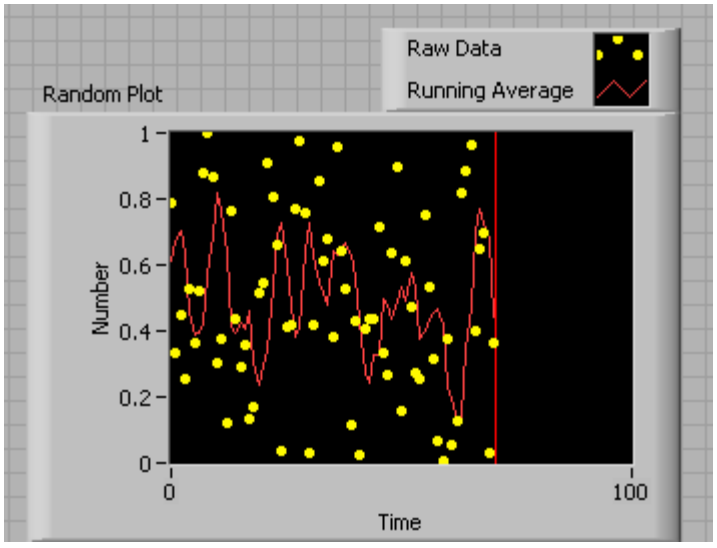
   Note: Modifying the axis text format often requires more physical space than was originally set aside for the axis. If you change the axis, the display may become larger than the maximum size that the VI can correctly present.

5. Stop the VI. Save any changes you have made and close the VI.

Random Average VI

Objective: To build a VI that displays two random plots on a waveform chart in sweep update mode. The plots should be a random plot and a running average of the last four points.

In this exercise, use a **For Loop (N = 200)** instead of a While Loop. Try to make your sweep chart look like this. You may also want to control the rate that this VI runs by using the **Wait Until the Next ms Multiple** function .



To change the mode of the Waveform chart, pop-up on the chart and select Properties. Click the Appearance tab and change mode to Sweep.

Use the **Random Number (0-1)** function (**Numeric** subpalette) to generate the data and average the last four data points.

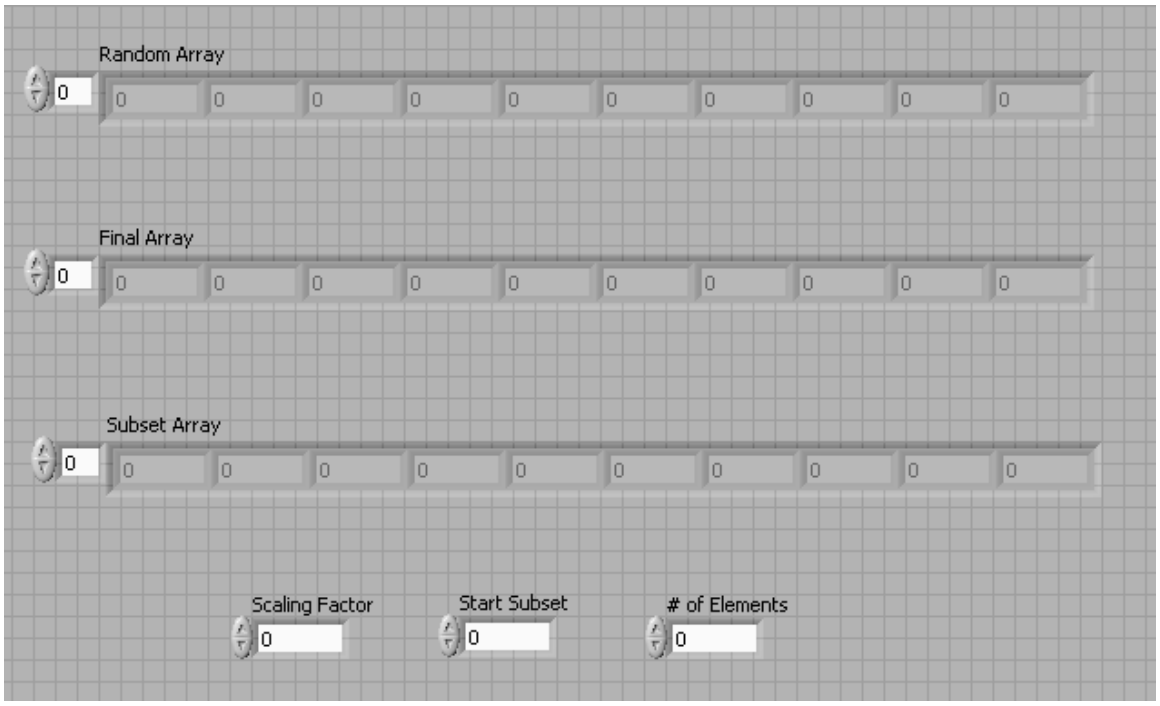
Save the VI. Name it Random Average.vi.

Array Exercise.VI

Objective: To create arrays and become familiar with array functions.

You will build a VI that creates an array of random numbers, scales the resulting array, and takes a subset of that final array.

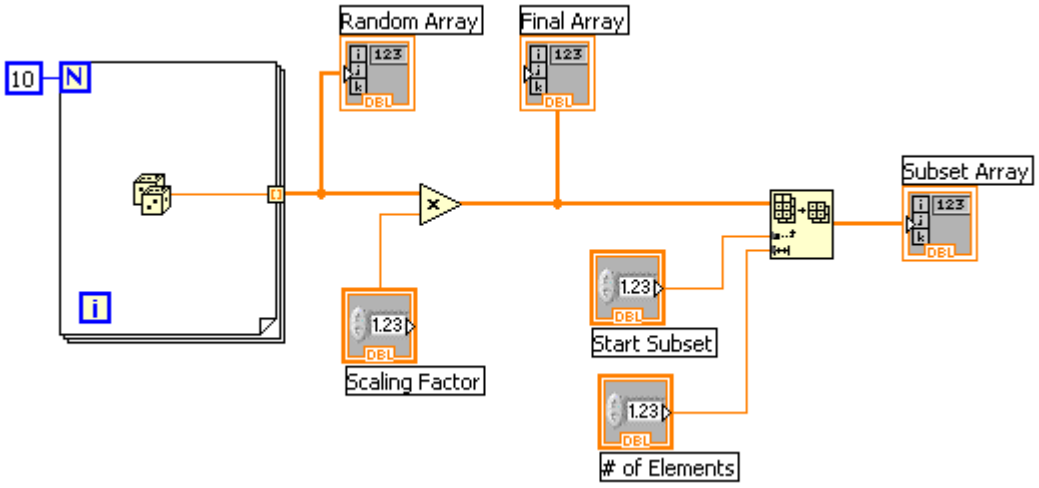
A. Open a new VI and follow the steps below to build the panel shown here:

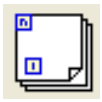


1. Create a digital indicator array.
   a. Place an **Array** shell (**Array & Cluster** subpalette) in the Panel window. Label the array shell Random Array.
   b. Place a numeric indicator (Numeric subpalette) inside the array shell using the pop-up menu. This indicator displays the array contents.
   c. Using the **Positioning** tool, drag a corner of the Array shell to contain 10 elements.
   d. Create two more numeric array indicators to display data in Final Array and Subset Array.
   e. Place three numeric controls to correspond to Scaling Factor, Start Subset, and # of Elements. Enter values into these controls.

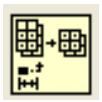
The VI will generate an array of 10 random numbers, scale them by the value in Scaling Factor, take a subset of that Final Array starting at Start Subset for # of Elements, and display the subset in Subset Array.

2. Build the block diagram shown below.





For Loop structure (**Structures** subpalette). This loop will accumulate an array of 10 random numbers at the tunnel as the wire leaves the loop. To set the loop to run 10 times, pop up on the N and choose Create Constant from the menu. Type 10 into the numeric constant.



Array Subset (**Array** subpalette). This VI removes a portion of an array starting where you specify and for a length you specify. The result will be displayed on the panel.

3. Save the VI as Array Exercise.vi.
4. Return to the front panel and run the VI a few times. From the diagram window, activate **Highlight Execution** and watch the execution of the VI.

The **For Loop** runs for 10 iterations. Each iteration generates a random number and stores it at the loop boundary. The Random Array is created at the tunnel when the **For Loop** completes. Then each value in the Random Array is multiplied by Scaling Factor to create Final Array. Lastly, a portion of the Scaled Array is displayed after the Subset Array function removes # of Elements starting at Start Subset.

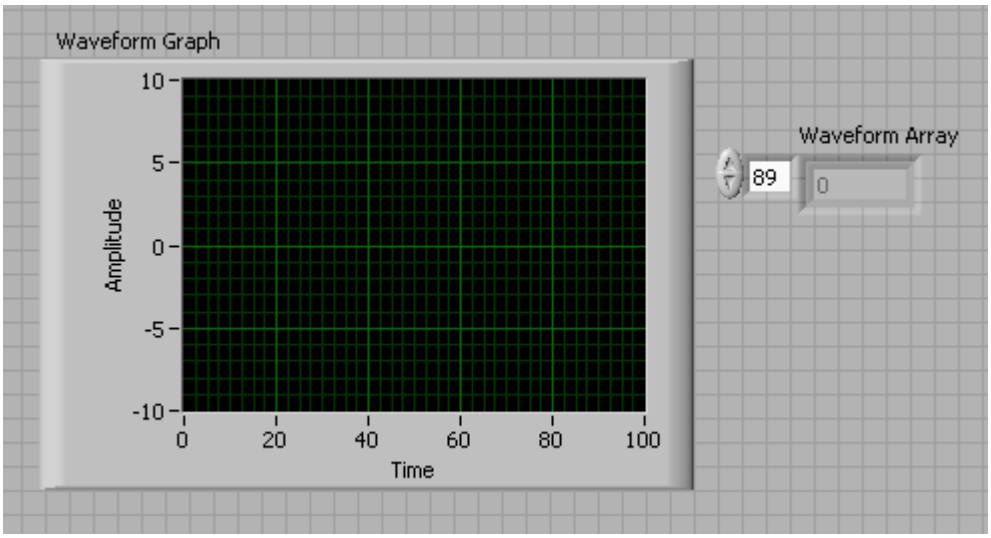
5. Close the VI.

Graph Waveform Array VI

Objective: To create an array using the auto-indexing feature of a **For Loop** and plot the array in a waveform graph.

You will build a VI that generates an array using the **Process Monitor VI** and plots the array in a waveform graph. You also will modify the VI to graph multiple plots.

A. Build this front panel. **NOTE: Waveform <u>Graph</u> is a different icon than Waveform <u>Chart</u>.**



1. Use a For Loop (N=100) and generate the data using:

   Process Monitor VI (**User Libraries » Basics** subpalette). This VI outputs simulated experimental data. In this exercise, this VI returns one point of simulated temperature data during each **For Loop** iteration.

2. Wire the index of the **Process Monitor VI** to the index (i) of the **For Loop**. The data is stored as an array at the border of the **For Loop** until all 100 executions are completed. Each element in the array has an index number associated with it. This is called "Auto-indexing" the array.

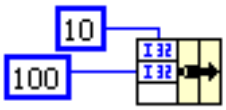
3. Wire the Temperature output of the **Process Monitor VI** to the right side of the loop. Then wire the output of the **For Loop** to the Digital Indicator Array. DO NOT wire your graph yet.

4. Save the VI in your library. Name it Graph Waveform Array I.vi.

5. Run the VI. You can view any element in the **Waveform Array** on the front panel s imply by entering the index of that element in the index display. If you enter a number greater than the array size, the display dims.

6. To view more than one element at a time, you can resize the array indicator. Place the **Positioning** tool on the lower-right corner of the array until the tool appears as a cross hatch and drag. The indicator now displays several elements in an ascending index order, beginning with the element corresponding to the specified index.

7. In previous VI's, you used the default value of the initial X and delta X value for the waveform. There are often cases where the initial X and delta X value will be a specific value. In these instances, you can use the **Bundle** function to specify an initial and delta X value for a waveform array. **Note:** The Bundle function should be outside of the loop.



Bundle function (**Cluster** subpalette) assembles the plot components into a single cluster. The components include the initial X value (10), the delta X value (100), and the Y array (waveform data). Use the **Positioning** tool to resize the function by dragging one of the corners. After the loop finishes execution, the **Bundle** function bundles the initial value of X (Xo), the delta value of X, and the array for plotting on the graph.

8. Use Save As and name it Waveform Array II.vi. Run the VI. The VI plots the auto-indexed waveform array on the waveform graph. The initial X value is 10 and the delta X value is 100.
9. Change the delta X value to 0.5 and the initial X value to 20.

   Notice that the graph now displays the same 100 points of data with a starting value of 20 and a delta X of 0.5 for each point (see the X axis). In a timed test, this graph would correspond to 50 seconds worth of data starting at 20 seconds. Experiment with several combinations for the initial and delta X values.

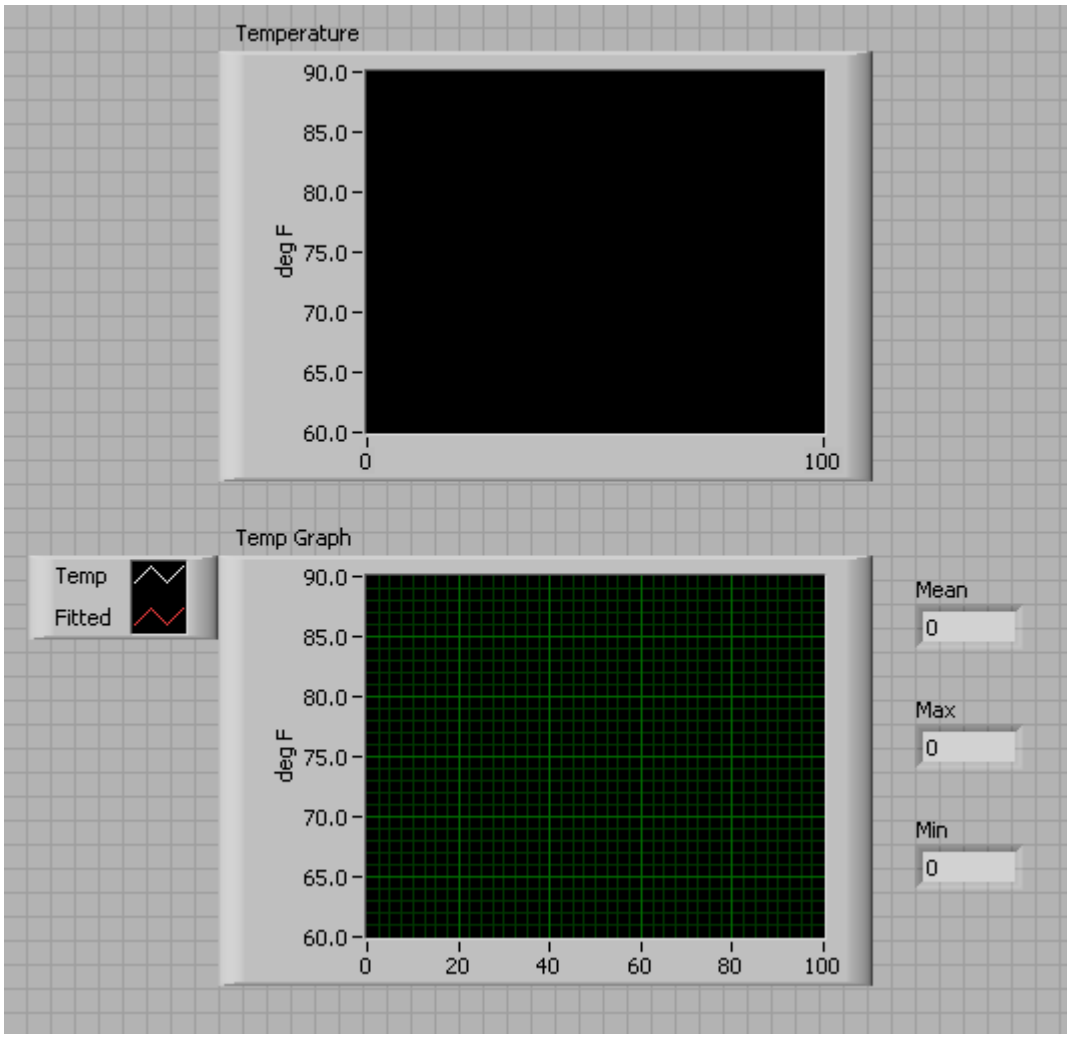
Temperature Analysis VI

Objective: To graph data and use the analysis VI's.

You will build a VI that measures temperature every 0.25 s for 10 s. During the acquisition, the VI displays the measurements in real time on a waveform *chart*. After the acquisition is complete, the VI plots the data on a *graph* and calculates the minimum, maximum, and average temperatures. The VI will display the best-fit of the temperature graph.

You will use this VI later, so be sure to save it as the instructions indicate.

A. Build this front panel.



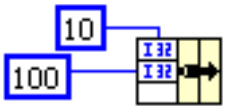
B. Build the block diagram using the following:

Thermometer VI (**Select a VI...** subpalette). Choose this VI from your library file.

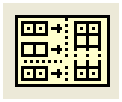
Array Max & Min function (**Array** subpalette). In this exercise, this function returns the maximum and minimum temperature measured during the acquisition.

Mean VI (**Analyze » Mathematics » Probability & Statistics** subpalette). In this exercise, this VI returns the average of the temperature measurements.

General Polynomial Fit VI (**Analyze » Mathematics » Curve Fitting** subpalette). In this exercise, this VI returns an array that is a polynomial fit to the temperature array. You will need three inputs to the left side of this VI: (1) the y-data, (2) the x-data (for this exercise, you need to create the x-data from a **For Loop** array), and (3) the order of the polynomial that you are trying to fit. This exercise uses five as the polynomial order. The General Polynomial Fit VI determines the best fit for the points in the temperature array.



As in the previous exercise, the **Bundle** function should be used to set the initial X and delta X values for the temperature data. This should also be used for the best fit data. The output of the two bundles will then be combined by the **Build Array** function before graphing.



Build Array function (**Array** subpalette). This function creates the proper data structure to plot two arrays on a waveform graph. Enlarge the **Build Array** function to include two inputs by dragging a corner with the **Positioning** tool. The actual temperature data and the best fit curve are the two arrays that should be displayed on the waveform graph.

C. Save the VI in your library. Name it Temperature Analysis.vi.

D. Run the VI. The graph should display the temperature data plot and “best fit” curve of the temperature waveform on the same graph. Try different values for the polynomial order constant (in the block diagram).

E. Change the appearance of your plot by modifying options such as plot styles and fill styles. You can create histogram graphs, general bar plots, or filled plots. The **Common Plots and Bar Plots** subpalette, in the legend pop-up menu, allows you to configure plot styles such as a scatter plot, a bar plot, or a fill to zero plot. You can configure the point, line, and fill styles in one step.

   a. Pop up on the Temp plot display in the legend of the Temp graph. Select **Common Plots** and select the **Scatter Plot** (top middle choice in **Common Plots**).

   b. Pop up on the **Fitted** plot display in the Legend of the Temp Graph and select the middle choice from **Bar Plots** in the legend pop-up menu.

F. Make sure your VI runs successfully, then save it as Temperature Analysis.vi.

G. Close the VI.

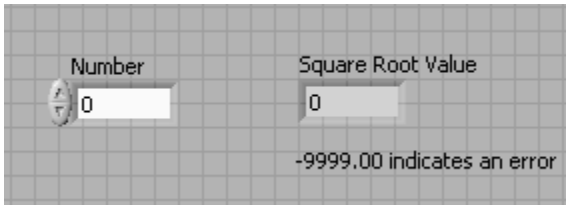
Square Root VI

Objective: To use the Case structure.

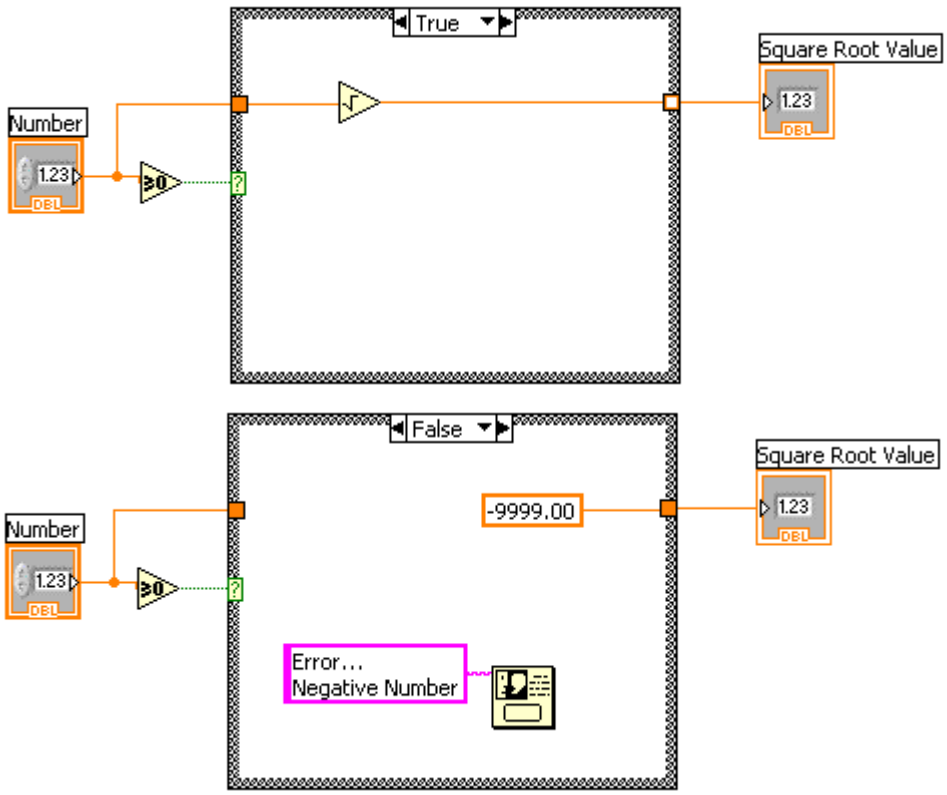
You will build a VI that checks a number to see if it is positive. If it is, the VI calculates the square root of the number; otherwise, the VI returns a message.

Warning: Do not run this VI continuously!

A. Open a new VI and build the front panel below. The Number digital control supplies the number. The Square Root Value indicator displays the square root of the number if Number is positive.



B. Build the corresponding block diagram shown below (instructions follow). Both the True and False cases are shown in the figure here but in actuality, only one case is displayed on the diagram at a time. IMPORTANT: To change cases, click on the arrows in the top border of the **Case** structure. First fill one case loop (ie TRUE), then change the case (ie FALSE) and build the other.



1. Select a **Case** structure (**Structures** subpalette). By default, the **Case** structure selection terminal is Boolean. It will automatically change to numeric if you wire a numeric control to the terminal. You can display only one case at a time.
2. Select the diagram objects shown below.

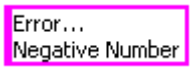
Greater or Equal to 0? function (**Comparison** subpalette). In this exercise, this function checks whether the number input is negative. The function returns a TRUE if the number input is greater than or equal to 0.

Square Root function (**Numeric** subpalette). In this exercise, this function returns the square root of the input number.

Numeric Constant (**Tunnel** pop-up menu). Place the **Wiring** tool on the white tunnel (the same one attached to the square root in the True window) and select **Create Constant**. Use the **Labeling** tool to type in the value into the constant. Pop up on the constant and select **Format & Precision....** Modify the numeric to have 1 digit of Precision and Floating Point Notation.

Notice that if both cases do not have data wired to the tunnel, the tunnel remains white. Ensure that a value is wired to the output tunnel from each case.

One Button Dialog function (**Time & Dialog** subpalette). In this exercise, this function displays a dialog box that contains the message “Error...Negative Number.”



String Constant (**Strings** subpalette). Enter text inside the box with the **Operating** tool.

In this exercise, the VI will execute either the True case or the False case. If the number is greater than or equal to zero, the VI will execute the True case. The True case returns the square root of the number. The False case outputs a -99999.0 and displays a dialog box with the message “Error...Negative Number.”

3. Save the VI. Name it Square Root.vi.
4. Return to the front panel and run the VI. Try numbers greater than zero and less than zero.
5. Close the VI.

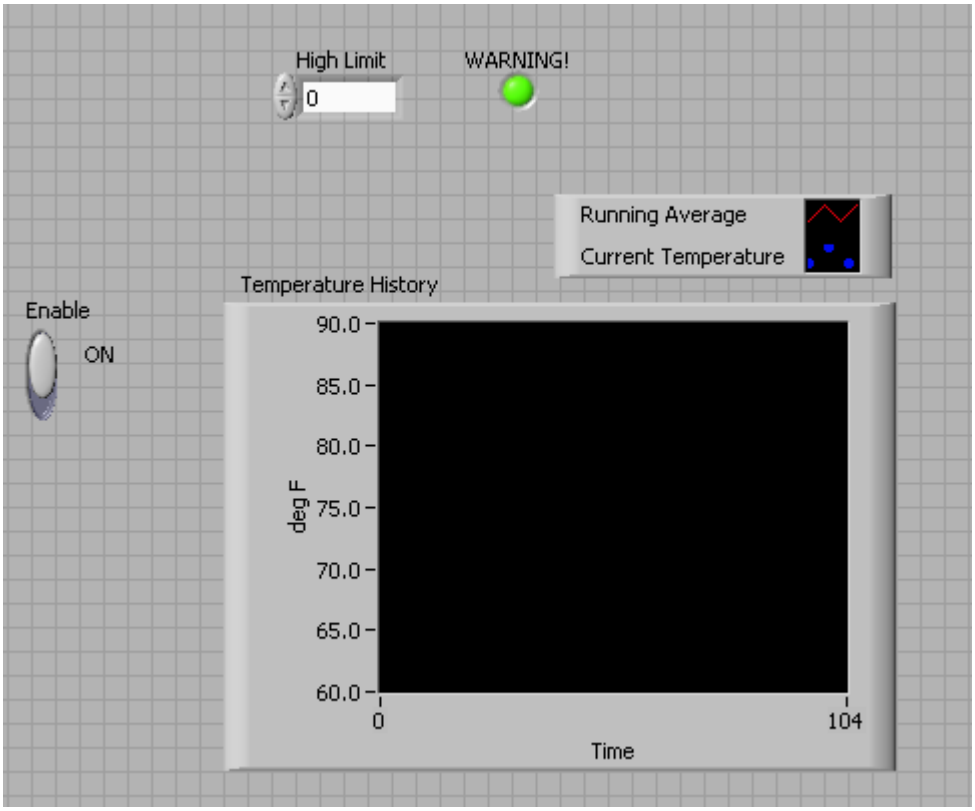
Temperature Control VI

Objective: To use the Case Structure.

You will modify the Temperature Running Average VI you created to detect when a temperature is out of range. If the temperature exceeds the set limit, a front panel LED will turn on and a beep will sound.

You will use this VI later, so be sure to save it as the instructions below describe.

1. Open the Temperature Running Average VI that you created earlier. Select Save As from the File menu, open your library files, and rename the VI Temperature Control.vi.
2. Modify the front panel so it looks like this:



The High Limit digital control specifies the upper temperature limit. The WARNING LED indicates if the temperature exceeds this limit. The Round LED can be found in the Boolean subpalette.

You create the numeric digital display values by popping up on the Temp History chart and selecting **Visible Items » Digital Display**.

3. Now modify the block diagram so that a beep will sound and a warning light will go on if the temperature limit is exceeded. Use the following functions.

Greater? function (**Comparison** subpalette). In this exercise, this function returns a TRUE if the temperature measured exceeds the temperature you specify in the High Limit control; otherwise, the function returns a FALSE.

Beep VI (**Graphics and Sound >> Sound** subpalette). In this exercise, this VI sounds a beep if the selection terminal of the **Case** structure receives a TRUE.

4. Save the VI.
5. Return to the front panel and enter 80 in the High Limit control. Run the VI.
6. Close the VI.

Time to Match VI

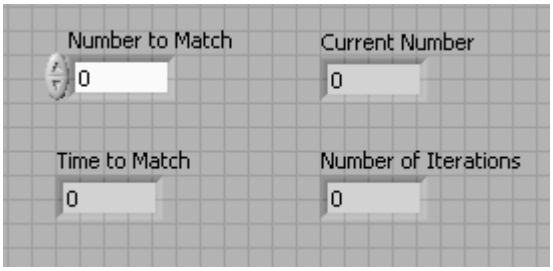
Objective: To use the **Sequence** structure.

You will build a VI that computes the time it takes to generate a random number that matches a number you specify. This VI uses the Auto Match VI you built earlier. The sequence structure performs the functions contained within them in sequential order, like the frames of a movie. As with the **Case** structure in the previous lesson, only one frame of the sequence structure is

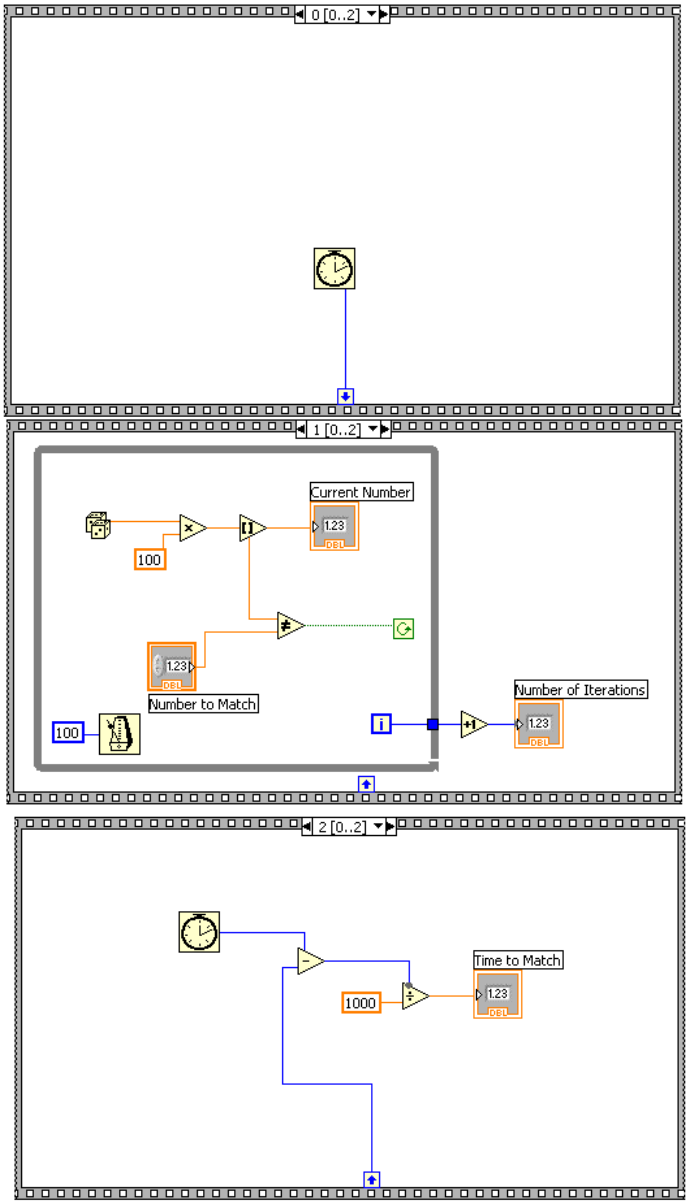
visible in the diagram at a time. Click on the arrows in the top border of the **Sequence** structure to move through the frames sequentially.

A. Open the Auto Match VI you created earlier and modify the front panel to look like the one below. Use the Save As command to save the VI as Time to Match.vi.

B. Modify the controls and indicators as depicted below.



C. Modify your block diagram so that it resembles the one shown below – as usual, instructions follow.



1. Choose a **Stacked Sequence** structure from the **Structures** subpalette and drag a selection area around the **While Loop**, **Increment** function, and # of iterations terminal.
2. Add a frame to the **Sequence** structure by popping up on the border of the frame and choosing **Add Frame After**. Repeat this step to add a second frame to the Sequence.
3. To place the **While Loop** in Frame 1: return to the frame that contains the While Loop, pop up on the frame border, and choose **Make This Frame»** 1.
4. A "sequence local" is a terminal used to pass data from one frame to the next. Create the sequence local by popping up on the bottom border of Frame 0 and choosing **Add Sequence Local** from the pop-up menu.

   The sequence local will appear as an empty square. The arrow inside the square will appear automatically when you wire to the sequence local.

5. Insert the  **Tick Count (ms)** function (**Time & Dialog** subpalette). This function reads the current value of the operating system's software timer and returns the value in milliseconds.

   In Frame 0, the **Tick Count (ms)** function reads the clock from the operating system and returns its value in milliseconds. In Frame 1, the VI executes the **While Loop** as long as the number specified does not match the number that the **Random Number (0-1)** function returns. In Frame 2, the **Tick Count (ms)** function again reads the operating system's software timer. The VI then subtracts the new value from the time read in Frame 0 and returns the elapsed time in seconds to the front panel.
6. Enter a number inside the Number to Match control. Save and run the VI. (Remember that you can use the shortcut keys to run the VI.)

NOTE: If **Time to Match** always reads 0.000, your VI may be running too quickly. To slow the VI, either run with Execution Highlighting enabled or increase the value on the diagram that is multiplied by the random number to a very large value such as 100,000.

7. Run the VI to make sure it works correctly.
8. Save the VI in your library and close it.

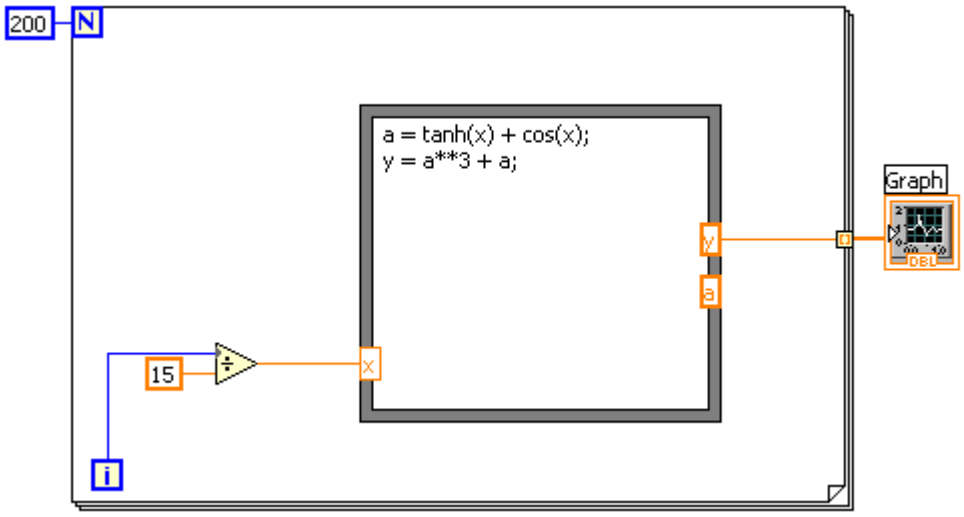
Formula Node VI

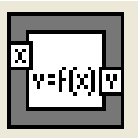
Objective: To use the **Formula Node**

You will build a VI that uses the **Formula Node** to evaluate a complex mathematical expression and graphs the results.

1. Open a new panel and put a Waveform graph on the front panel. This will be used to display the plot of the equation y = f(x)^3 + f(x), where f(x) = tanh(x) + cos(x).

2. Build this block diagram.





Formula Node (**Structures** subpalette). With this node, you can directly enter formulas. Create the input terminal by popping up on the border and choosing **Add Input** from the pop-up menu. You create the y output by choosing **Add Output** from the pop-up menu. You must also define the intermediate or "dummy" variable *a*.

NOTE:

- When you create an input or output terminal, you must give it a variable name that exactly matches the one in the formula. The names are case sensitive—if you use a lower case "r" to name the terminal, you must use a lower case "r" in the formula.
- A semicolon (;) terminates each formula statement.

During each iteration, the VI divides the iteration terminal value by 15.0. The quotient is wired to the input of the **Formula Node** *(x)*, which computes the function value. The output value *(y)* is then wired to the **For Loop** border. The VI then stores the result in an array at the **For Loop** border (auto-indexing). After the **For Loop** finishes executing, the VI plots the array.

2. Save the VI. Name it Formula Node Exercise.vi.
3. Return to the front panel and run the VI.
4. Close the VI.

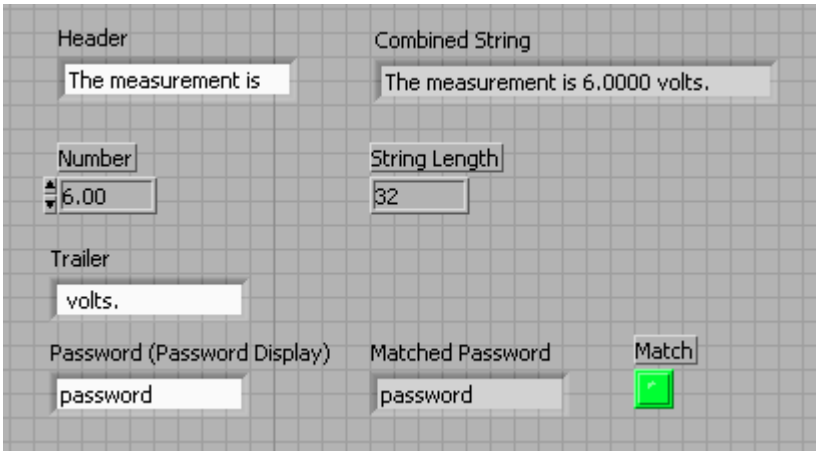
Build String VI

Objective: To create a SubVI utilizing the Format Into String, Concatenate Strings, and String Length functions

You will build a VI that converts a number to a string and concatenates the string to other strings to form a single output string. The VI also determines the output string length. The VI also tests if a password matches a given password.

You will use this VI later, so be sure to save it as the instructions describe.

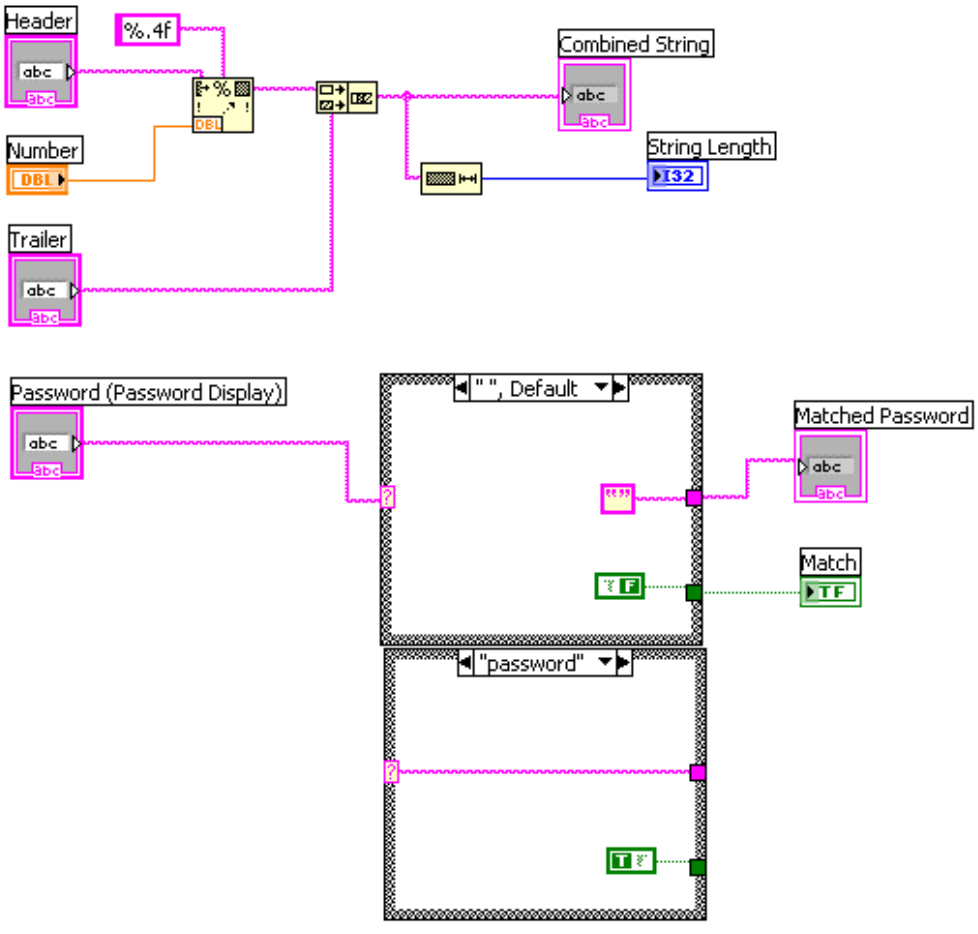
1. Open the Build String.vi in your library file.
2. Build the front panel shown. Be sure to modify the controls and indicators as depicted. The numerics and Boolean have been created for you.



This VI will concatenate the input from the two string controls and the digital control into a single output string and display the output in the string indicator. The digital indicator will display the string length.

The VI also will test whether the given string matches the input password string. The VI asserts a Boolean if there is a match, and a string indicator displays the matched string.

3. Build the diagram shown using the instructions that follow:



Format Into String function (**String** subpalette). In this exercise, this function converts the number you specify in the digital control, Number, to a string. Only strings can be input to the concatenate function.

To create the format string %.4f, pop up on the **Format Into String** function and select **Edit Format String**. From the **Edit Format String** dialog box, create the format string.

Concatenate Strings function (**String** subpalette). In this exercise, this function concatenates all input strings into a single output string. To increase the number of inputs, resize the function using the **Positioning** tool.

String Length function (**String** subpalette). This function returns the number of characters in the concatenated string.

4. In this exercise, the **Case** structure is used to see if the **Password** string matches your password. In the selector label for the default (False) case, use the **Labeling** Tool to replace "false" with an empty string (‘’ ‘’). In the selector label for the true case, type your password in place of "true". For this example, a password of the word "password" is used, but you can specify another password. Be sure to remember this password, as you will need it in a later exercise.
5. In the Front Panel, wire the connectors for the subVI. Pop up on the **Icon/Connector** and select **Show Connector**. Use the wiring tool to wire the input and output terminals to the front panel controls and indicators.

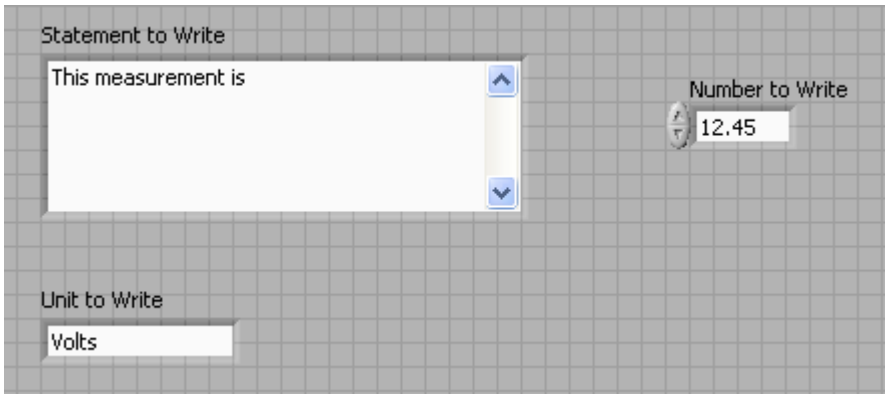
When the wiring is complete, select **Show Icon**.

6. Save the VI under its same name.
7. On the front panel, use the **Get Color** and **Set Color** tools in the Tool palette to make the button turn Red when the password does not match.
8. Type text inside the three string controls and a number inside the digital control. (Type "password" in the Password control.) Run the VI.
9. Type a different word in the Password control and run the VI.
10. Modify this diagram to add spaces automatically between Header, Number, and Trailer.
11. Save and close the VI.

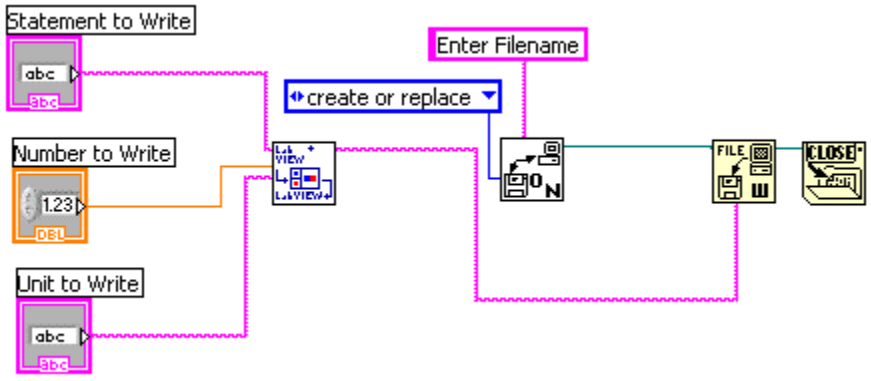
File Writer VI

Objective: To write data to a file.

1. Create this front panel.



2. The front panel contains two string controls with normal display and a numeric control. The string control can be expanded by dragging a corner with the positioning tool. Pop-up on the string control, select **Visible Items >> Scrollbar** to enable the scroll bar on the control.
3. The *Statement To Write* control will input the message written to the file. The *Number To Write* and *Unit to Write* controls will input their values and write them to the same file as the *Statement to Write* control.
4. Now create the block diagram using the icons and instructions that follow. Remember that the terminals and their names will be visible if the **Wiring** tool is placed over an icon. Other things that may help you wire icons together correctly are selecting **Show Context Help** from the Help menu or popping up on the icon and selecting **Visible Items >> Terminals** from the pop-up menu.



Build String.vi (**Select a VI...** subpalette). This is the subVI you created that concatenates the three input strings to one combined string. Wire the inputs to the appropriate terminals.

Open/Create/Replace File VI (**File I/O** subpalette). This VI displays an interactive file dialog box to open or create a file.

a. Pop up on the VI "prompt" terminal and select Create Constant. Type in "Enter Filename". When this VI is run, a window will appear called "Enter Filename" in which you can type the name of the file that the data is to be written to.

b. Pop up on the VI "function" terminal and select Create Constant. Use the **Operating** tool to change the terminal value to "create or replace" (it's one of several choices). This specifies the creation of a new file or the replacement an existing file.

Write File function (**File I/O** subpalette). This function writes the concatenated strings to the file.

Close File function (**File I/O** subpalette). This function closes the file.

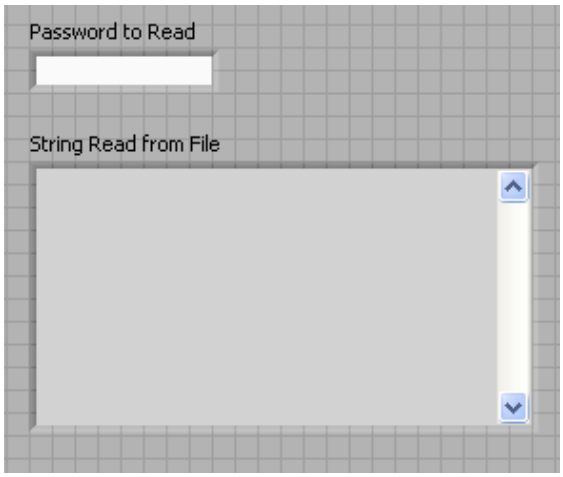
5. Wire the output from the Build String VI to the Data terminal of the Write File icon.
6. Wire "refnum" terminals of the Open File and Write File icons together. Then connect the "dup refnum" of the Write File icon to the "refnum" of the Close File icon. These will execute from left to right and the file will not close until the file is completely written.
7. Save the VI in your library. Name it File Writer.vi.
8. Make entries in all of the front panel controls – include your name in the Statement to Write. Run the VI. When the box appears, make sure that you are in your Z:\ directory, name the file: demofile.txt. Click on Save or OK.
9. Open demofile.txt (double click on it in Explorer) and print it.
10. Once the VI is working correctly, save it and close it.

File Reader VI

Objective: To read data from a file.

You will build a VI that reads the file created in the previous exercise and displays the information read in a string indicator if the user's password matches the specified password from the Build String VI.

1. Open a new VI and build the front panel shown here using the directions that follow.



The front panel contains a string control with Password Display enabled and a string indicator that displays the information read from the file. If the Password string control matches the specified password in Build String VI, the data is read from a file. Otherwise, a message is displayed to indicate that the password did not match.

1. Try to create the block diagram yourself to complete this task. It is similar to the previous exercise except that you will be opening a file instead of creating it and you will read the file instead of writing to it. Reading the file should also be password protected. You will need to use the following function:

Read File function (**File I/O** subpalette). This function reads file size bytes of the data from the file starting at the current file mark (beginning of the file). **NOTE:** You will need to wire the File Size terminal of the **Open/Create/Replace File** icon to the count terminal of the **Read File** icon.

2. Save the VI. Name it File Reader.vi.
3. Run the VI. A dialog box appears. Find the file **demofile.txt** and click on Open or OK. The *String to Read from File* indicator should display the file contents if the *Password to Read* matches the password value in the Build String VI.

Temperature Logger VI

Objective: To save data to a file in a form that a spreadsheet or a word processor can access later.

You will modify the Temperature Control VI to save the time and current temperature to a data file. You will use this VI later, so be sure to save it as the instructions below describe.

1. Open the Temperature Control VI.
2. Select Save As from the File menu and save this VI as Temperature Logger.vi.
3. The front panel already is built. You will modify the block diagram using the following.

Get Date/Time String function (Time & Dialog subpalette). This function returns the time (in string format) when the temperature measurement was taken. The True-False Boolean constant (Boolean subpalette) sets the function to include seconds in the string. Use the Operating tool to change the False Boolean constant to the True Boolean constant.

Tab Constant (String subpalette).

End of Line constant (String subpalette).

Format into String function (String subpalette). This function converts the temperature measurement (a number) to a string and builds the following formatted data string:

Time String (tab) Temperature String (end of line).

4. Save the VI.
5. Run the VI. A dialog box appears, prompting you to enter a filename. Select the Z:\drive, name the file temp.txt and click on OK or Save.

The VI creates a file called temp.txt. The VI then takes readings (one every half-second) and saves the time and temperature data to a file until you press the Enable switch. When the VI finishes, it closes the file.

6. Open the temp.txt file with Excel. It is a Tab delimited ASCII file. Print the file.
7. Close the VI.

Temperature Application VI (Optional)

Objective: Combine Temperature Logger VI and the Temperature Analysis VI create a VI that does the following:

1. Takes a temperature measurement once every second until you stop the VI.
2. Displays both the current temperature and the average of the last three measurements on a waveform chart.
3. If the temperature goes over a preset limit, turns on a front panel LED.
4. After each measurement, logs the date, time (including seconds), temperature, average of the last three measurements, and a one-word message describing whether the temperature is “Normal” or “OVER” the preset limit. The VI should log data so that each item appears in one column of a spreadsheet.
5. After you stop the acquisition, plots both the raw temperature data and a best-fit curve in a graph, and displays the average, maximum, and minimum temperatures.

HINT: Start with the Temperature Logger VI that you built and copy and paste the appropriate parts of the Temperature Analysis VI to complete this assignment.

NOTE: To create an array of data at the border of a While Loop pop-up on the tunnel and select “Enable Indexing”

6. Save your VI as Temperature Application.vi.

Spreadsheet Example VI

Objective: To save a 2D array in a text file so that a spreadsheet can access the file and to display numeric data in a table control.

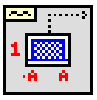
In the Temperature Logger VI exercise, you formatted the string so that tabs separated the columns and end of lines separated the rows. In this exercise, you will examine a VI that saves numeric arrays to a file in a format you can access with a spreadsheet.

1. Open the Spreadsheet Example VI in your library. **The VI is already built.**
2. Run the VI.

- The VI generates a 2D array (128 rows x 3 columns). The first column contains data for a sine waveform; the second column contains data for a noisy waveform; and the third column contains data for a cosine waveform. The VI plots each column in a graph and displays the data in a table indicator.

3. After the VI displays and plots the data, it displays a dialog box for the filename. Type wave1.txt and click on OK or Save. Later, you will examine the file that the VI created.
4. Examine the block diagram. The following VI's are used:

Sine Waveform (**Analyze » Waveform Generation** subpalette). In this exercise, this VI returns a numeric array (128 elements) containing a sine pattern. The constant 90 in the second subVI call specifies the phase of the sine pattern (cosine pattern).



Uniform White Noise Waveform (**Analyze » Waveform Generation** subpalette). In this exercise, this VI returns a numeric array (128 elements) containing a noise pattern.

Transpose 2D Array function (**Array** subpalette). This function rearranges the elements of the 2D array so that element[i,j] becomes element[j,i].

Write To Spreadsheet File (**File I/O** subpalette). This VI formats the 2D array that Build Array creates into a spreadsheet string, and writes the string to a file. The string has the following format:

To Fractional function (**String » Additional String to Number Functions** subpalette). In this exercise, this function converts an array of numeric values to an array of strings that the table indicator displays. The format string specifies the string to be in the 2 precision fractional format.

5. Remove the Transpose 2D Array function and wire the diagram to run without it. Run the VI. Save the output file as wave2.txt.
6. Now popup on the **Transpose** terminal of the **Write to Spreadsheet** function and create a constant. Use the Operate Tool to switch the setting to True. This is another way to transpose the array as it is read into the spreadsheet file.
7. Run the VI and save the file as wave3.txt.
8. Close the VI.

9. Open the wave1.txt using Excel to view its contents. Observe that the sine waveform data appears in the first column, the random waveform data appears in the second column, and the cosine waveform data appears in the third column.
10. Open the file wave2.txt and view its contents. Note that without the **Transpose Array** function, the data is arranged in three rows, not columns.
11. Open the file wave3.txt and view its contents. Note that the data can be transposed using the transpose terminal of the subVI.

NOTE: This example had only three arrays stored in the file. To include more arrays, you can increase the number of inputs to the **Build Array** function.

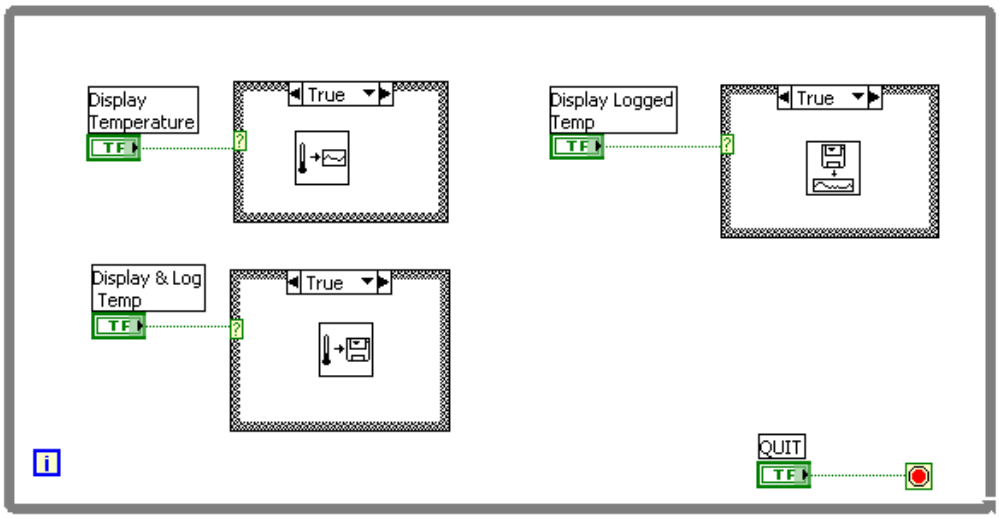
Temperature System VI (Optional)

Objective : To use the setup options for a VI and a subVI and the Key Navigation option for front panel controls.

You will build a temperature monitoring system you can use to view three different subtests on request.

Assume that you require a VI with a user-driven interface. Thus, you want to ensure that the program executes correctly by hiding the Stop button on the toolbar and running the VI when it opens.

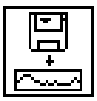
1. Open the Temperature System VI in your library.
2. The front panel contains four labeled buttons. The mechanical action of each button is set to "Latch when pressed." Assign the Key Navigation options for each button to the indicated keyboard key.
3. Build this block diagram using the instructions that follow.



Be sure to leave all the FALSE cases empty. Use the VI's below to build the block diagram:

Display Temp VI (**User Libraries » Basics Course** subpalette). In this exercise, this VI simulates temperature measurement every half-second (500 ms) and plots it on a strip chart. Open the subVI front panel by double-clicking on its icon and examine the block diagram. Close the panel before you proceed.

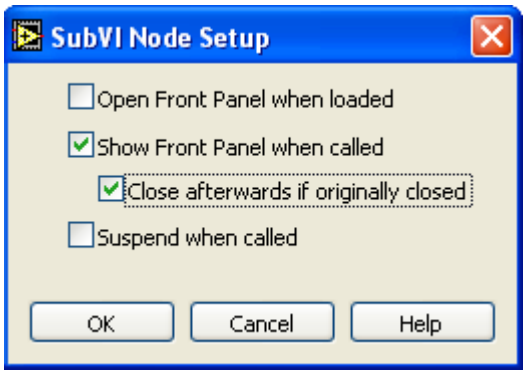
Display and Log Temp VI (**User Libraries » Basics Course** subpalette). In this exercise, this VI simulates temperature measurement every half-second (500 ms), plots it on a strip chart, and logs it to a file. Open the subVI front panel by double-clicking on its icon and examine the block diagram. Close the panel before you proceed.



Display Logged Temp VI (**User Libraries » Basics Course** subpalette). In this exercise, you use this VI to interactively select a file. The VI then opens the file, reads the logged data, and displays it on a graph. Open the subVI front panel by double-clicking on its icon and examine the block diagram. Close the panel before you proceed.

Note: The QUIT button's default state is FALSE.

4. Configure the Display Temp subVI to pop open its front panel when called by popping up on the Display Temp VI icon and choosing SubVI Node Setup from the pop-up menu. Configure the dialog box as shown here.



5. Repeat step 4. for the Display and Log Temp subVI and Display Logged Temp subVI.
6. Save the VI.
7. Return to the front panel and run the VI. Test run all options. Try the key assignments to display the temperature, display and log the temperature, and so on.

NOTE: The three subVIs called from the block diagram all have their "RETURN" buttons assigned to the <return> key. Try pressing <return> to return to the main front panel.

8. Stop the VI.
9. When you are sure everything is in proper working order, configure the Temperature System VI so that it automatically runs when you open the VI. Pop up on the icon pane and choose VI Setup from the pop-up menu. Configure the Execution Options dialog box so that the Run When Opened box is checked.
10. Configure the VI so that none of the buttons is visible in the toolbar during the VI execution. To hide the options, choose Window Options from the VI Setup dialog box and uncheck the Show Toolbar option. Also, disable the menubar.
11. Save all subVIs and save and close the Temperature System VI.

12. Open the Temperature System VI. The VI should automatically execute when you load it.

13. Test run the VI again. When you have finished, close the VI.

Serial Read & Write VI

Objective: To examine a VI that communicates with an RS-232 device.

To talk to any external device through the serial port, you must know exactly how that device connects to your serial port, what serial port settings are supported, and exactly how the string commands and responses are formatted.

1. Open the (Demo) Serial Write & Read VI in your library. It is already built. This VI illustrates the necessary component of a VI for communicating with a serial port device.
2. The Serial Port Initialization VI is used to configure the instrument as a serial device.
   a. When you use a serial device, the software needs to know which "port" it is attached to. The port numbers are assigned as follows:

0:	COM1	5:	COM6	10:	LPT1
1:	COM2	6:	COM7	11:	LPT2
2:	COM3	7:	COM8	12:	LPT3
3:	COM4	8:	COM9	13:	LPT4
4:	COM5				

In this VI, the port number is 0, indicating that the device is connected to the computer at COM1.

   b. The the following settings have also been entered:

baud rate	=	9600
data bits	=	8
parity	=	no parity
stop bits	=	1

   These values are device determined and are generaly provided in the instrument manual. This device (instrument) communicates at a baud rate of 9600, with 8 data bits, no parity and stop bit = 1.

   c. The size of the buffer is set at 2000 bytes. This determines the size of the buffer on the computer in which to store data which is transmitted by the device.

   d. Also specified in the initialization is the method of flow control. The computer and the instrument have to agree to a handshaking protocol. With handshaking, the sender and the receiver notify each other when their buffers fill up. The sender can then stop sending new information until the other end of the serial communication is ready for new data. Either software or hardware handshaking can be done:

      - *Software* handshaking involves embedding control characters in transmitted data. For example, XON/XOFF flow control works by enclosing a transmitted message between the two control characters XON and XOFF.

- *Hardware* handshaking uses voltages on physical wires to control data flow. The RTS and CTS lines of the RS-232 interface are frequently used for this purpose. Most lab equipment uses hardware handshaking.

In this VI, the flow control parameters = hardware handshaking.

5. Run the VI. The string "*IDN?" queries the instrument for its identification. You should receive a response that this is a (Demo) National Instruments GPIB and Serial Device Simulator.
6. Try sending other commands to the Instrument Simulator. Below are some commands to try.

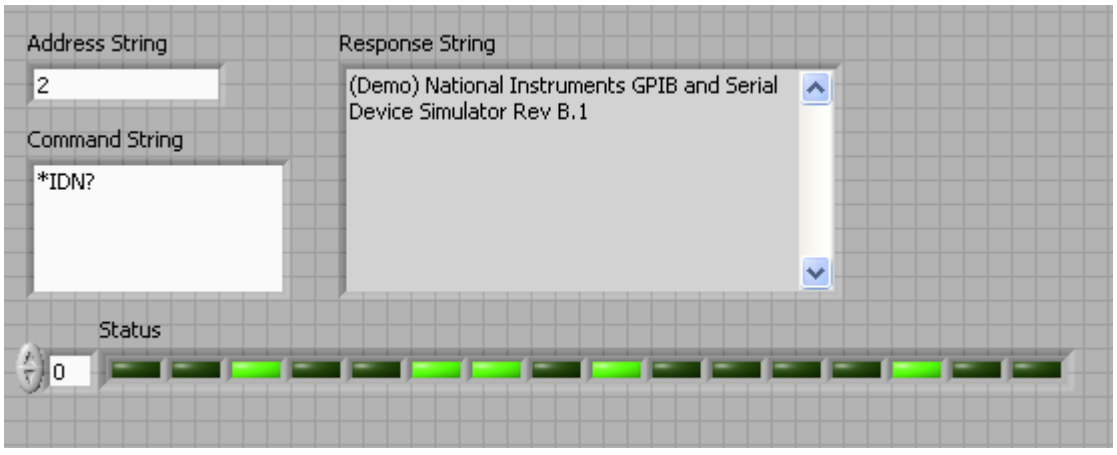
MEAS: DC?	Returns a voltage reading
SOUR:FUNC SIN; SENS:DATA?	Output sine waveform
SOUR:FUNC SQU; SENS:DATA?	Output square waveform
SOUR:FUNC RAND; SENS:DATA?	Output random noise waveform
SOUR:FUNC PCH; SENS:DATA?	Output chirp waveform

Close the VI.

GPIB Write & Read VI

Objective: To make a VI that communicates with a GPIB instrument.

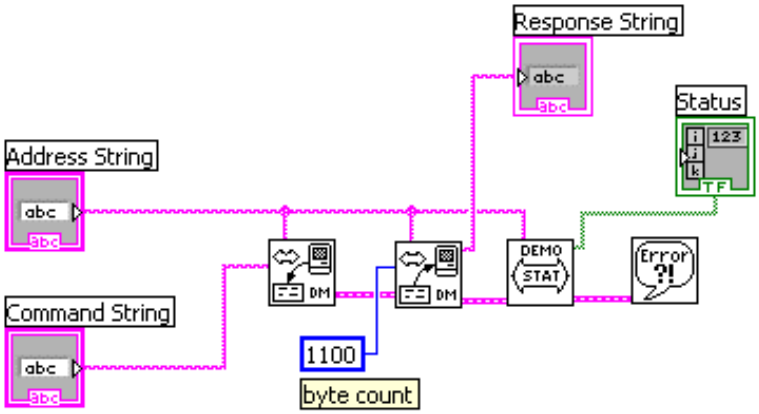
1. Open a new VI and create this front panel.



Create the Status array by first placing an array shell on the panel. Then place a Square LED inside the shell. Resize the array to show all 16 status Booleans.

3. Build the diagram shown here.

NOTE: Use the Demo version of the GPIB functions instead of the read ones. These can be found in the **User Libraries>>Basics** subpalette. The Simple Error Handler.vi can be found in the **Time and Dialog** subpalette.



4. Save this VI as (Demo) GPIB Write & Read.vi.
5. Instead of communicating through a serial port, devices can be interfaced with a GPIB board. Unlike serial ports, GPIB ports are not standard in computers. Boards can be purchased and added to computer. All GPIB functions must know the address of the instrument that they are communicating with. In this example set the address string to 2. This is an address that can be configured on the instrument. Then type *IDN? inside the command string, and run the VI. The string "*IDN?" queries the instrument for its identification. You should receive a response that this is a National Instruments GPIB and Serial Device Simulator.
6. Try sending other commands to the Instrument Simulator. Below are some commands to try.

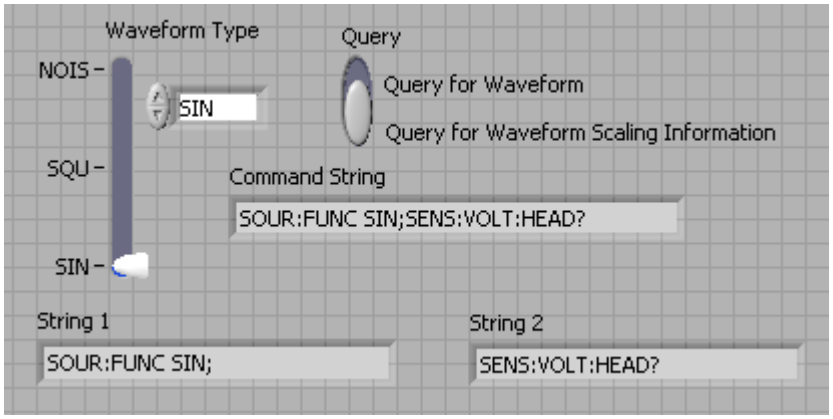
MEAS: DC?	Returns a voltage reading
SOUR:FUNC SIN; SENS:DATA?	Output sine waveform
SOUR:FUNC SQU; SENS:DATA?	Output square waveform
SOUR:FUNC RAND; SENS:DATA?	Output random noise waveform
SOUR:FUNC PCH; SENS:DATA?	Output chirp waveform

7. Close the VI.

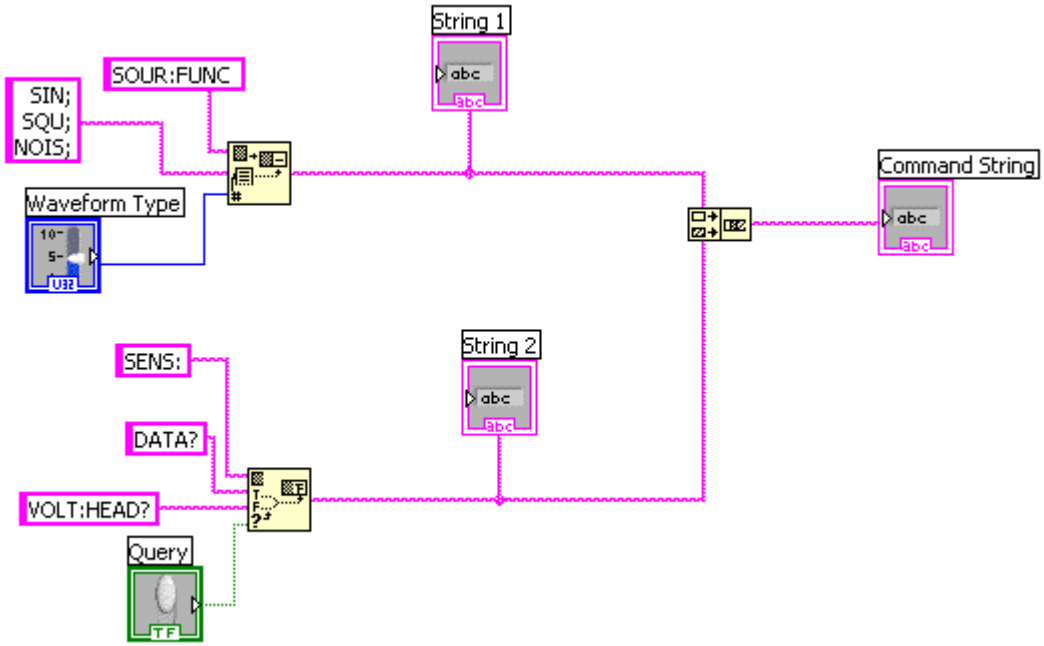
Build Command String VI

Objective: To build a command string based on input selections from the front panel. This will be useful for communicating commands to an instrument.

1. Open a new panel and build the front panel shown below. Be sure to modify the controls and indicators as depicted.



2. The vertical pointer slide can be found on the Numeric subpalette. To make the slider text, pop up on the slide and choose Text Labels. Using the Labeling tool, change “min” to “SIN”. Using the Operating tool, select the “max” option from the text display and use the Labeling tool to change “max” to “SQU”. Pop up on the text display, choose Edit Items, select Insert and add NOIS.
3. Pop up on the slide and choose **Visible Items » Text Display** to hide the text display.
4. Build the block diagram below using the icons and instructions on the next page.



The following functions are used:

Pick Line function (**String » Additional String Functions** subpalette). In this exercise, this function chooses either “SIN;”, “SQU;”, or “NOIS;” depending on the value of the Function slide control, and appends the string to “SOUR:FUNC:”.

Append True/False String function (**String » Additional String Functions** subpalette). In this exercise, this function chooses either “DATA?;” or “VOLT:HEAD;”, based on the value of the Query switch, and appends the string to “SENS:”.

Concatenate Strings function (**String** subpalette). In this exercise, this function concatenates the output string of the Pick Line & Append, a semicolon, and the output string of Select & Append.

3. Return to the front panel and run the VI. Try different settings for the controls and observe the string indicators.
4. Close and save the VI. Name it Build Command String.vi.