Basic Data Acquisition with LabVIEW

INTRODUCTION

This tutorial introduces the creation of LabView Virtual Instruments (VI's), in several individual lessons. These lessons create a simple sine wave signal, with controllable amplitude and frequency, and it is measured by a separate signal sampling device. Both signals are displayed, and signal acquisition effects such as aliasing and saturation can be observed.

LESSON 1: INTRODUCTION TO VI COMPONENTS & TOOLS

A LabVIEW VI consists of two separate windows: the front panel and the block diagram. Think of a typical electrical box, where the front of the box has all of the meters and switches, and the components that connect the switches and meters are located inside the box. The LabVIEW front panel is similar to the front panel of the electrical box, and the LabVIEW block diagram is similar to the innards of the electrical box. That is, the outputs and inputs of the VI go on the front panel, and the mathematics, conditions, and algorithms behind the inputs and outputs go in the Block Diagram.

Open LabVIEW from the **Start Menu→All Apps→N→NI LabVIEW SP1 (32-bit)**. In the getting started window that comes up, selelct **Create Project→ Blank VI→Finish**. The block diagram and front panels of a new, untitled VI will appear. In the front panel window, use **File → Save As** to save your VI in a user-modifiable location. LabVIEW is notorious for crashing frequently without warning, so it is best to save your work often.

Next, open up the Tools Palette, **View → Tools Palette**, if it is not already open. You have several different tools at your disposal while editing the VI, and they change when you switch between the block diagram and front panel. You can use the palette, or you can sequence through the tools by pushing the **Tab** key. Available tools:

Extended Index Finger: Acts as if the object you are manipulating is fixed and that its behavior cannot be modified. Use this tool to toggle buttons and enter values into text boxes.
Arrow: Use this to modify objects instead of interact with them. With the arrow you can resize objects, move them around, and select them.
Text Cursor: Use this cursor to edit any text fields in the Block Diagram and Front Panel. You can also add text to the block diagram and front panel with this tool.
Spool of Wire: This cursor only works in the Block Diagram. Use it to create wires between different objects, so that they interact with each other.

These are the fundamental tools. The rest of the tools are explained within the LabVIEW help files.

Now go to the Block Diagram. **Right Click** in the workspace so that the **Functions** menu pops up. These are all of the LabVIEW objects available for the Block Diagram. Select **Express → Exec Contronl → While Loop**, and create a rectangle by holding down the mouse at the top-left of the Block Diagram and dragging down to the right before releasing. Once LabVIEW enters the while loop, things within it continue to execute until the Boolean condition at the stop sign is met.

Return to the Front Panel. **Right Click** in the workspace to bring up the **Controls** menu, and select **Express → Buttons → Text Button**. Place this button in the bottom right of the workspace.

Use the Text Cursor to change the text within the button to **"Exit"**, and the label above it to **"Exit"**. Next, **Right Click** on the button to bring up its menu. De-select **Visible Items → Label** to hide the label above the button. In the same menu, select **Mechanical Action → Switch Until Released** to change the behavior of the button accordingly. Your front panel should now look like the panel in Figure 1.

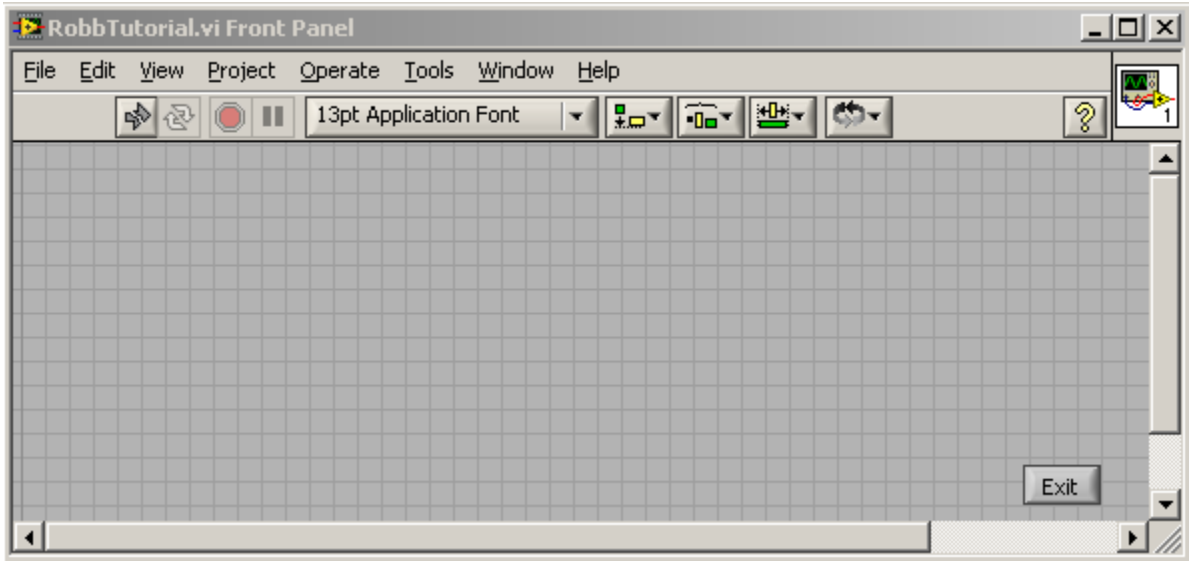

Figure 1: Front Panel with Exit Button

Return to the Block Diagram. Notice an object has been created, named **"Exit"**, for the button you created in the front panel. Use the arrow cursor to move it inside the while loop you created. Then use the Wire Cursor to create a wire between it and the stop sign. The Block Diagram should resemble the one below, in Figure 2.

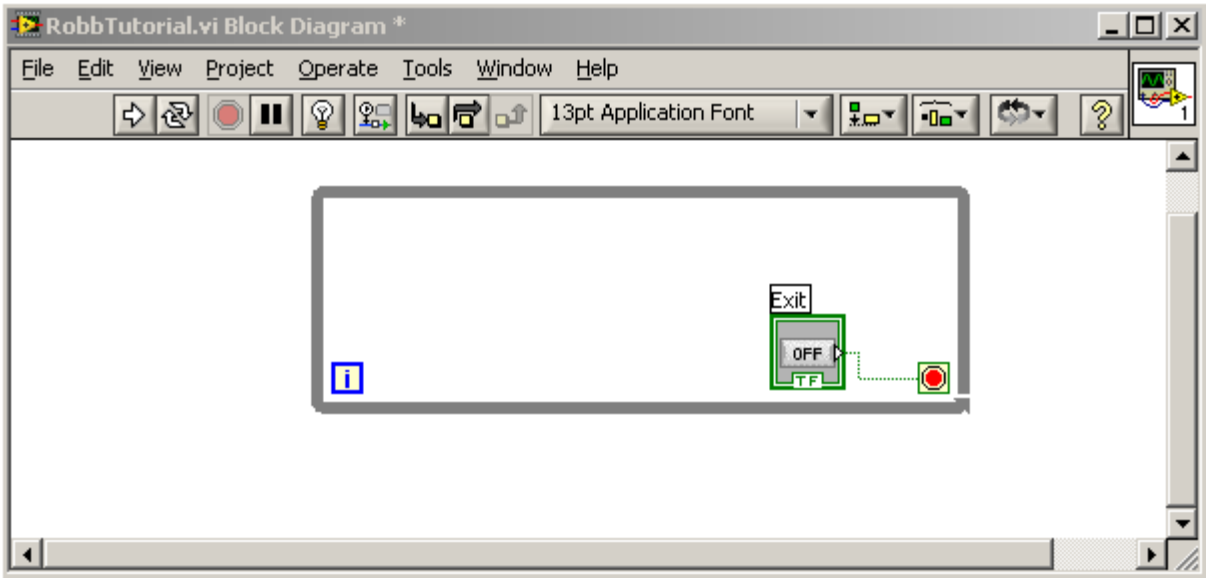

Figure 2: Block Diagram of Exit

The While Loop will now run until the button is clicked on the Front Panel. Try it out by saving your work and clicking on the arrow up near the menu bar to see your VI work. Click on the button to end it. You have now created your first functional VI!

LESSON 2: VIEWING A SIGNAL

LabVIEW is very good at interfacing with laboratory equipment and bringing signals into the computer. You probably don't have access to this equipment, so we will create a signal within our VI using LabVIEW objects. We will also add editable fields to adjust the signal's magnitude and frequency.

Right-click in the front panel of the same file as before. Select **Graph Indicators → Waveform Graph**, and place the graph in the top left corner of the screen. Rename the label at the top-left of the graph to "Original Signal".

Right-click in the front panel again, and select **Num Ctrls → Num Ctrl**, and place this text box to the left of the graph. Place another Num Ctrl right below. Label the first Num Ctrl "Amplitude", and the second Num Ctrl "Frequency".

Change the Amplitude value to 1, and the Frequency value to 5. Right-click inside the text-box of each Num Ctrl and select **Data Operations → Make Current Value Default**. This forces the values to start at 1 and 5.

One you have made these changes, the front panel should now look like Figure 3 below.

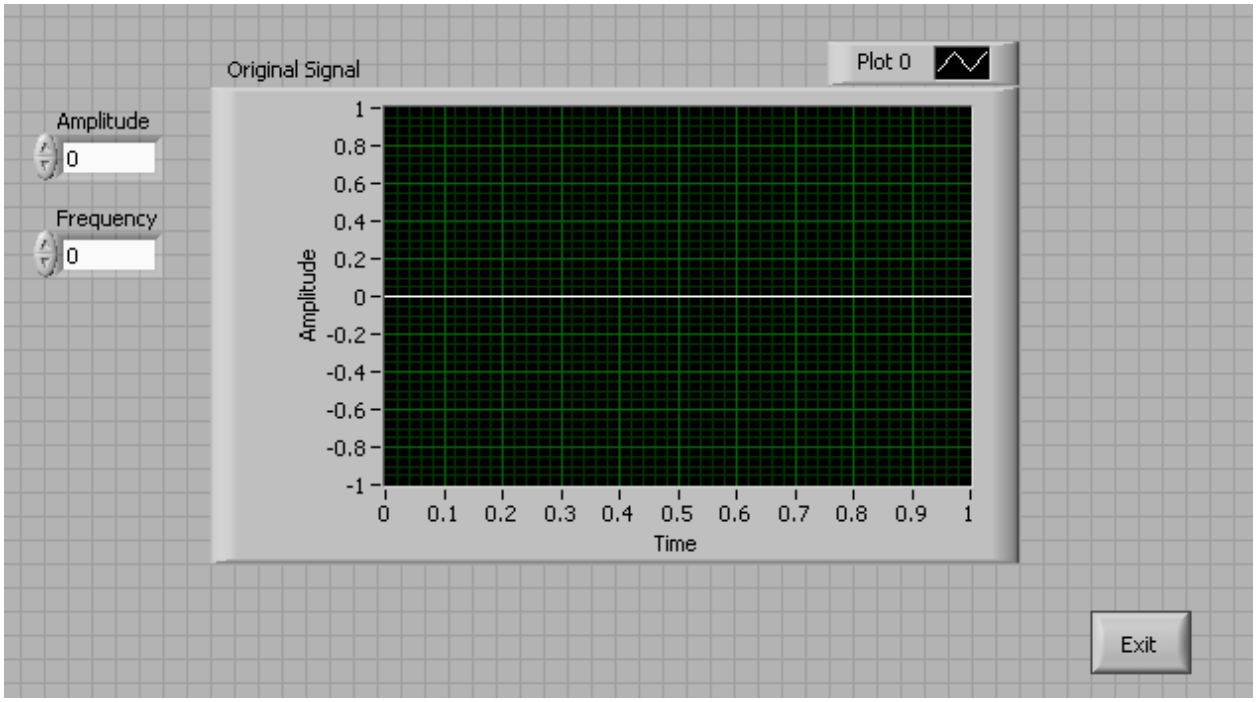

Figure 3: Front Panel with Waveform Graph

Now go to the block diagram. Drag the three new blocks, "Original Signal", "Frequency", and "Amplitude", inside the while loop. These blocks represent the numbers and signal that appear on the front panel.

Add a fourth block by right-clicking in the screen and selecting **Signal Processing → Wfm Generation → Sine Wfm**. This block creates a sine wave signal based on numbers that are provided to it using block diagram wires. Use the wire spool tool to see all the connections that can go into the block. You should see, clockwise from top, the following connectors: "reset signal", "offset", "signal out", "error out", "sampling info", "error in", "phase", "amplitude", and "frequency". You can get more information for each of these connectors by selecting **Help → Show Context Help**, and selecting the block in question.

With the wire-spool tool, right click on the "phase" connector of the Sine Wfm block. Select **Create → Constant**. This automatically attaches a constant number to a connector. Leave this value at 0.

Move the blocks around inside the while loop so that the Frequency and Amplitude blocks appear to the left, so the Sine Wfm block appears in the middle, and so the Original Signal block appears on the right. Use the wire spool cursor to connect the blocks together as shown in Figure 4. Save and run the script. The front panel should be similar to Figure 5.

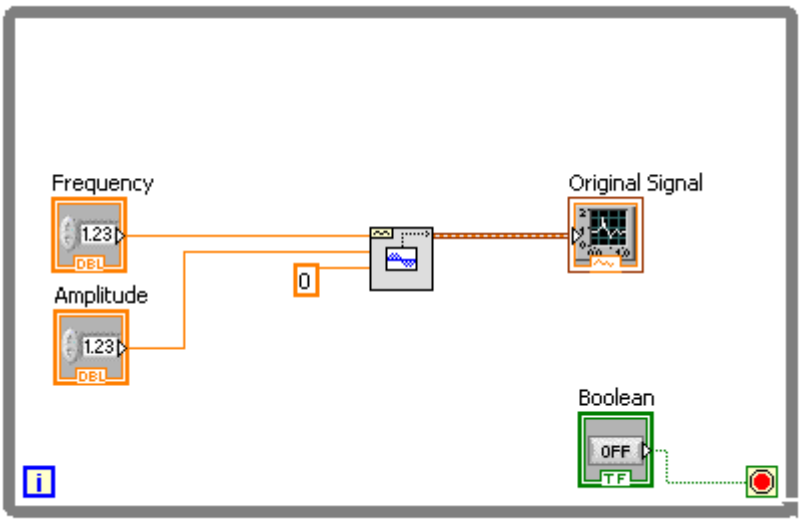

Figure 4: Block Diagram Connections

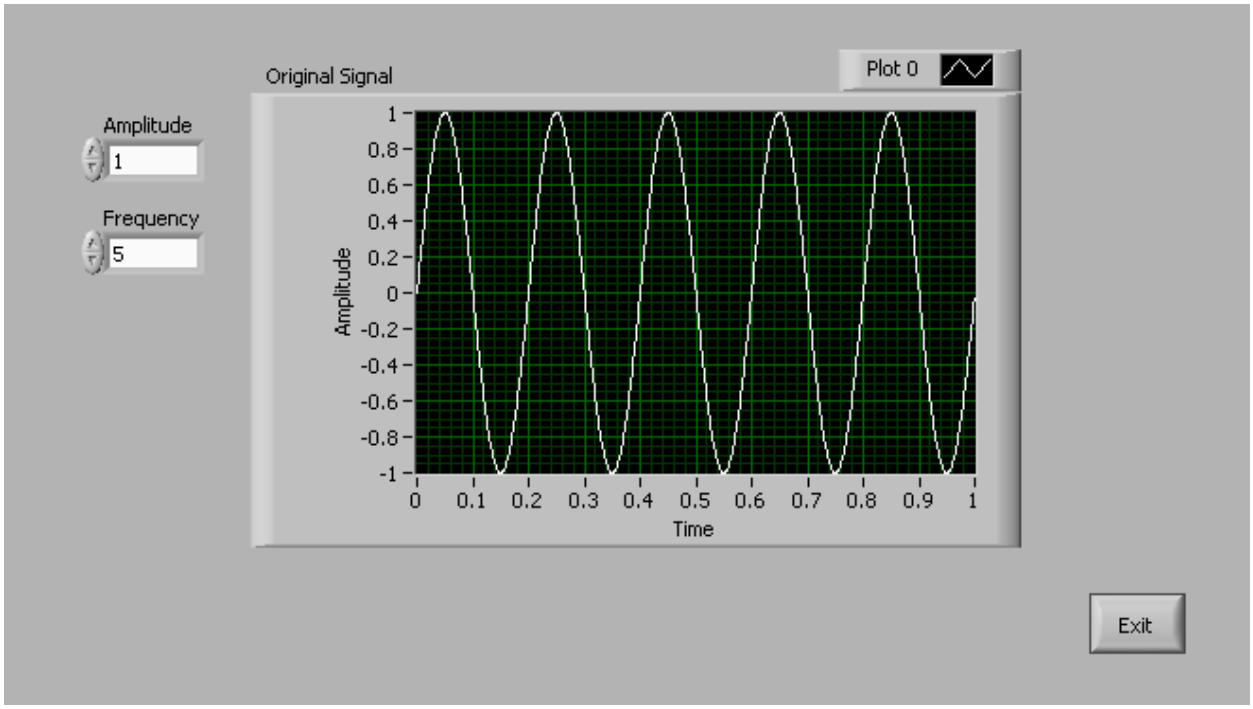

Figure 5: Front Panel During Run

LESSON 3: BUILDING THE SAMPLING WAVEFORM

We will now add a second graph, which will acquire the original signal at a user-specified sampling frequencing and display the samples.

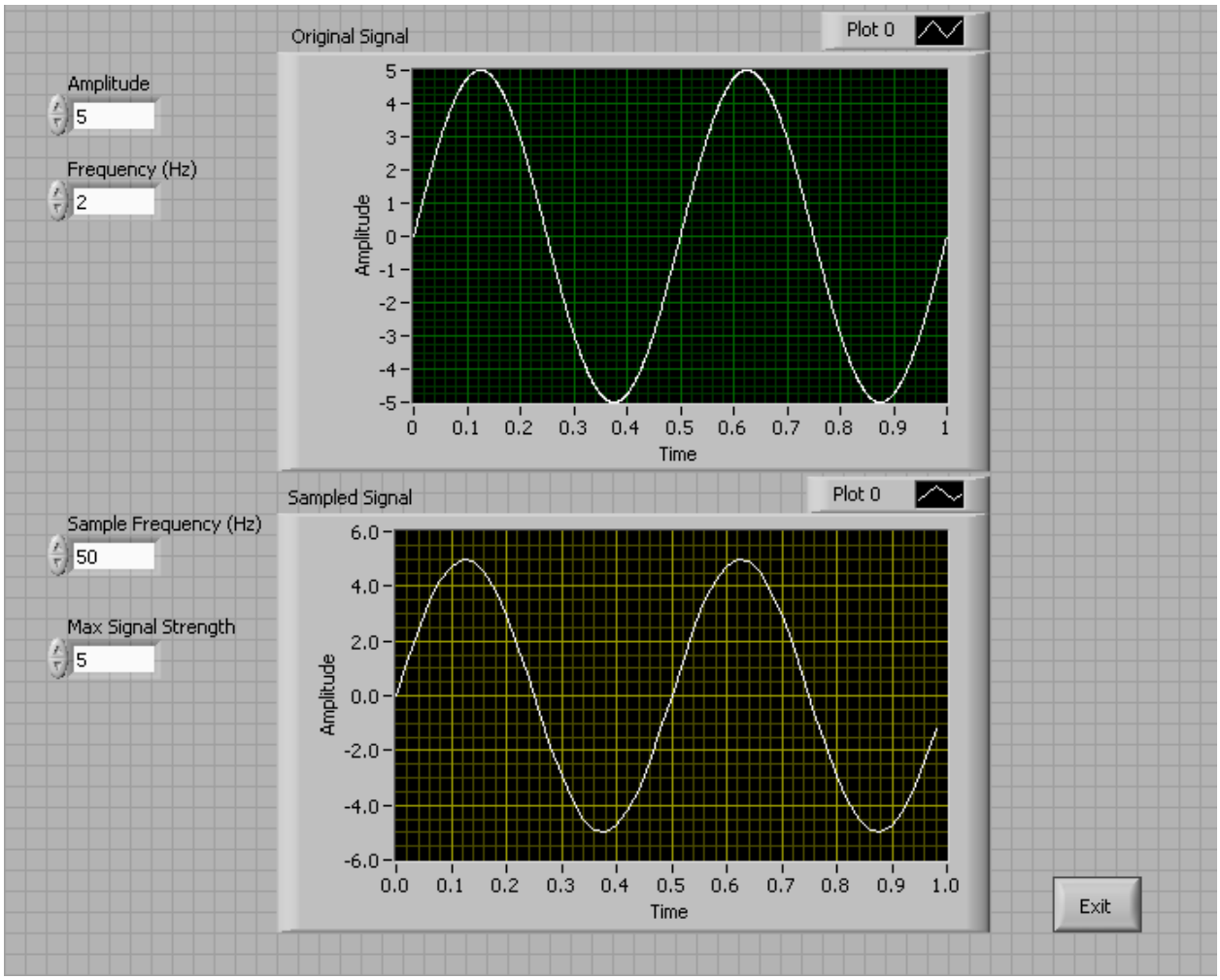

Figure 6: Final Front Panel

Add the numerical controls and second graph as shown in Figure 6. The second graph, at bottom, is an XY Graph, and it can be found by right-clicking in the screen and selecting **Graph Indicators → XY Graph**.

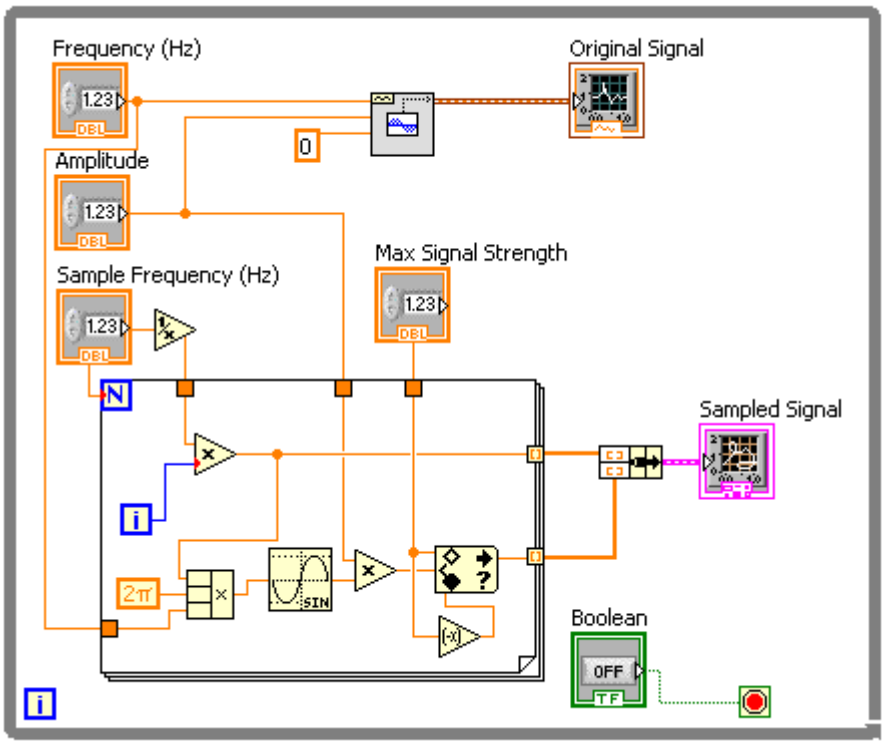

Figure 7: Final Block Diagram

Modify your block diagram as shown in Figure 7. Most of the new blocks can be found by right-clicking and selecting **Programming → Mathematics → Numeric** and **Programming → Comparison**.

The smaller box with an **N** in the upper-left-hand corner is a for loop (**Programming → Structures**). It executes all of the components inside of it **N** times. In our case, it executes the same number of times as the sample frequency. The other for-loop variable, **i**, counts from 0 to **N**-1, incrementing once for each iteration of the for-loop. We multiply **i** with the period of the sampling frequency, which we find using the **1/x** block. This gives us a value for the time of one sample.

We use a compound arithmetic block (**Programming → Numeric → Compound Arithmetic,** place the block and right click on it click **Change Mode → Multiply** right click again and click A**dd input**) to multiply (**Programming → Numeric**) the time with constant **2*pi** (**Programming → Numeric → Math Constants**), and with the frequency of the actual signal. We then take the **sin** of this value (**Mathematics → Elementary → Triginometric**), and multiply it by the amplitude of the actual signal. This completes the equation **y = A*sin(2*pi*f*t)**.

The **y** value is then passed through an **In Range & Coerce** block (**Programming → Comparison**) to demonstrate saturation. If **y** is greater than **Max Signal Strength** or less than **Max Signal Strength*(-1)** (**Programming → Numeric → Negate**), then it is coerced to one of these extremes.

The **t** and **y** values are automatically compiled into arrays by the **for** loop. Array signals appear thicker than scalar signals, as shown in Figure 7. The two arrays are then bundled into a

cluster (**Programming → Cluster, Class, & Variant → Bundle**), which makes a compatible signal for the XY Graph.

Operation of the virtual instrument should now be seamless and error free…

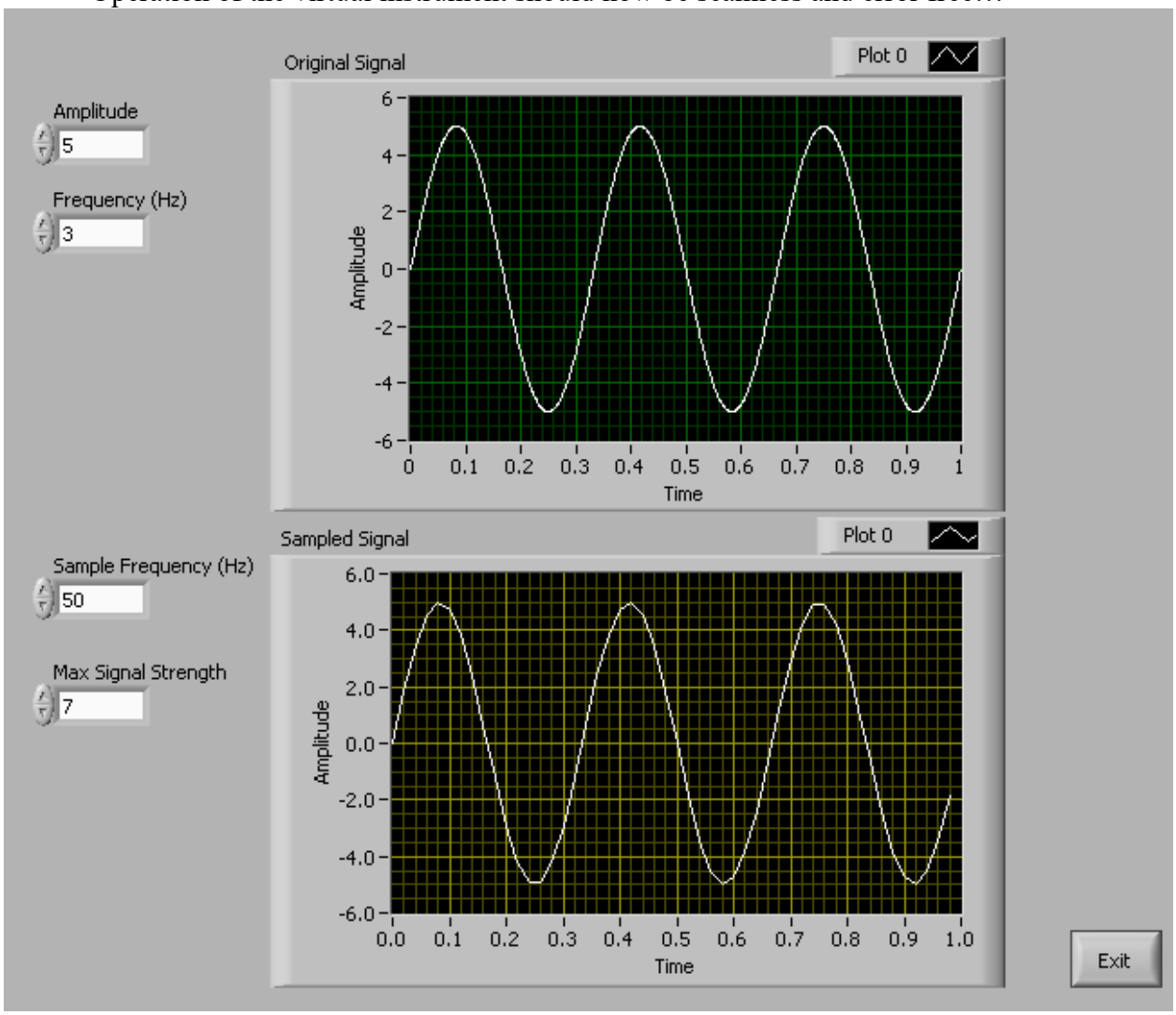

Figure 8: Good Sampling.

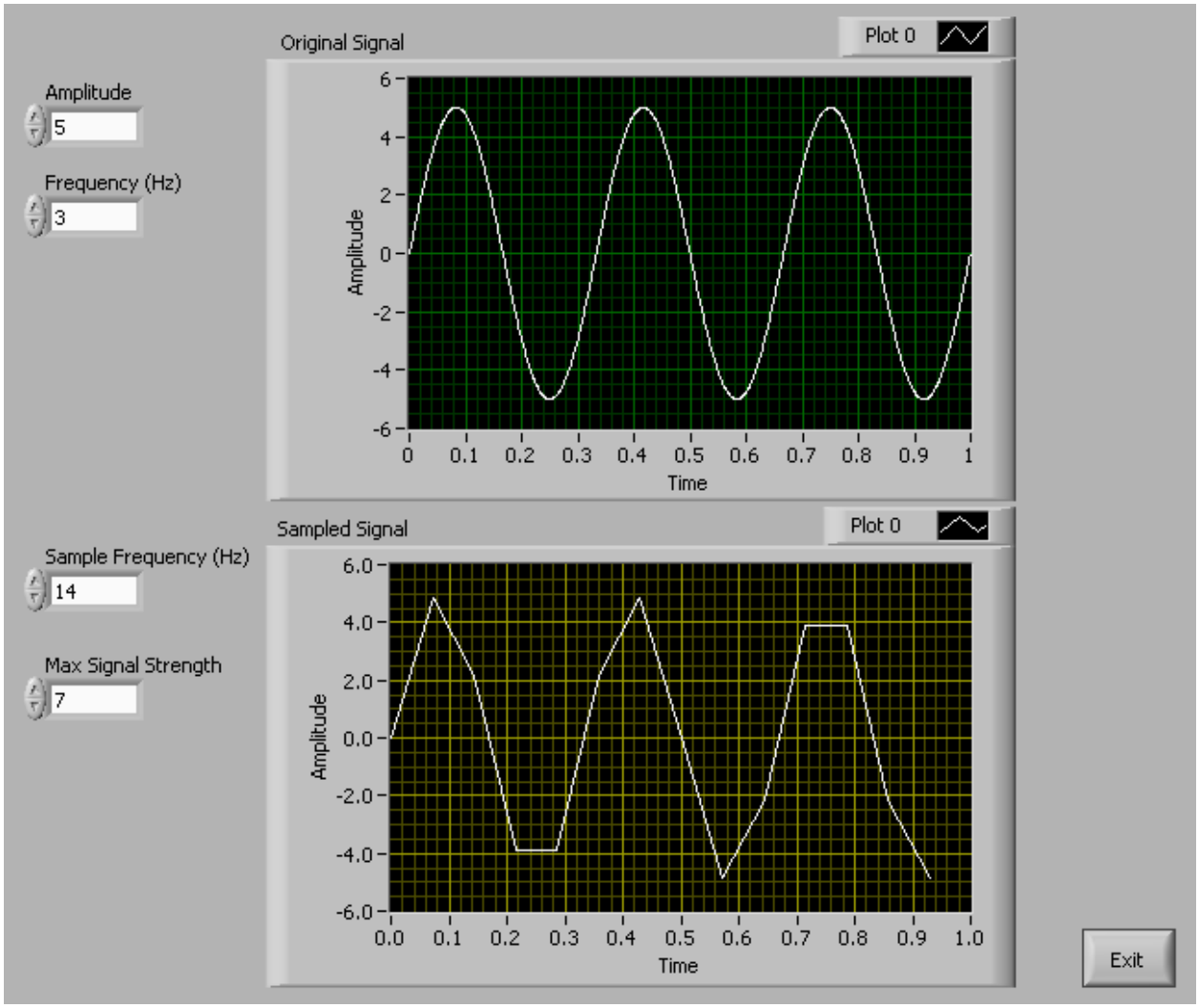

Figure 9: Degraded Sampling.

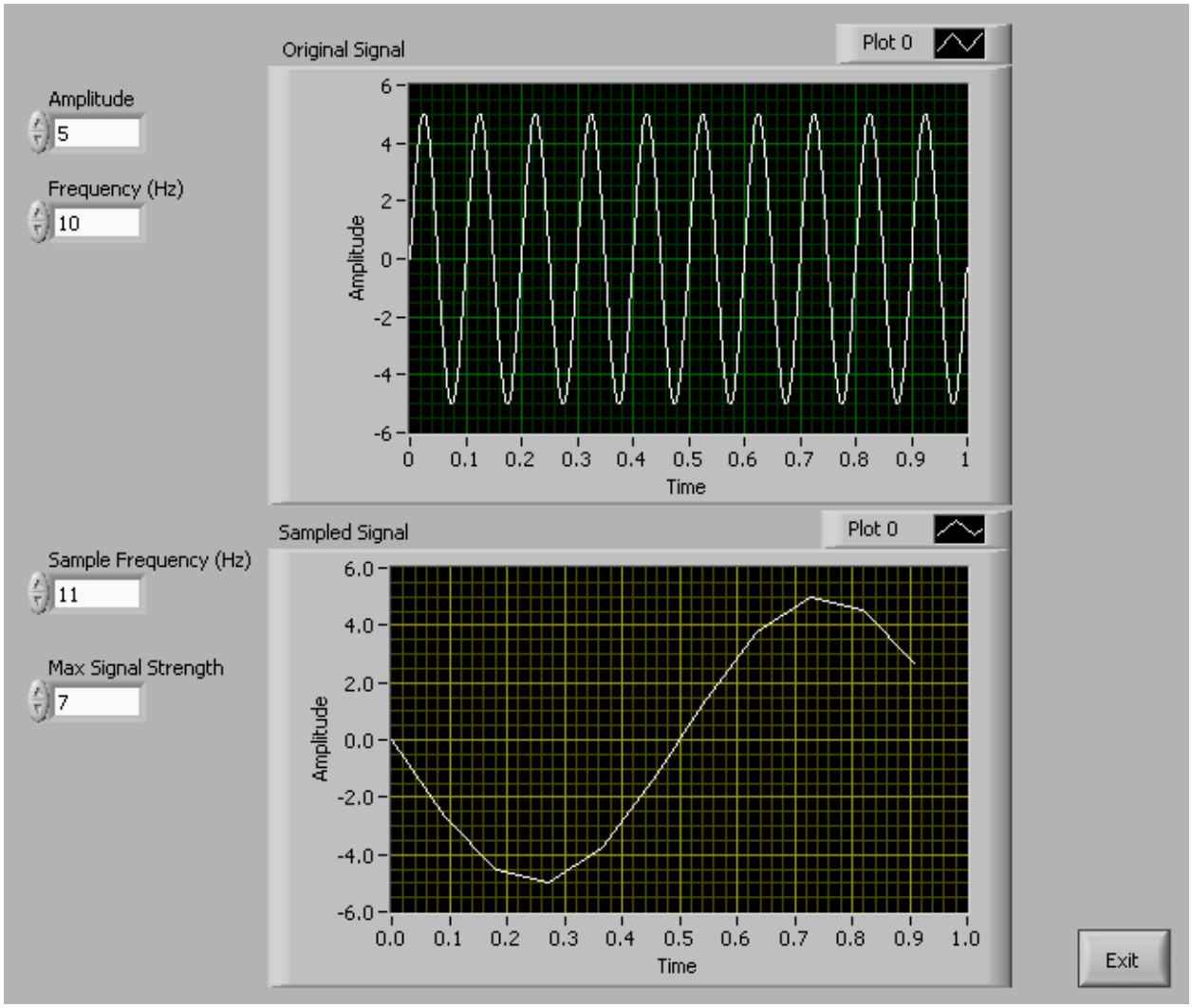

Figure 10: Aliasing. The 10 Hz signal "folds" and appears as a 1 Hz signal.

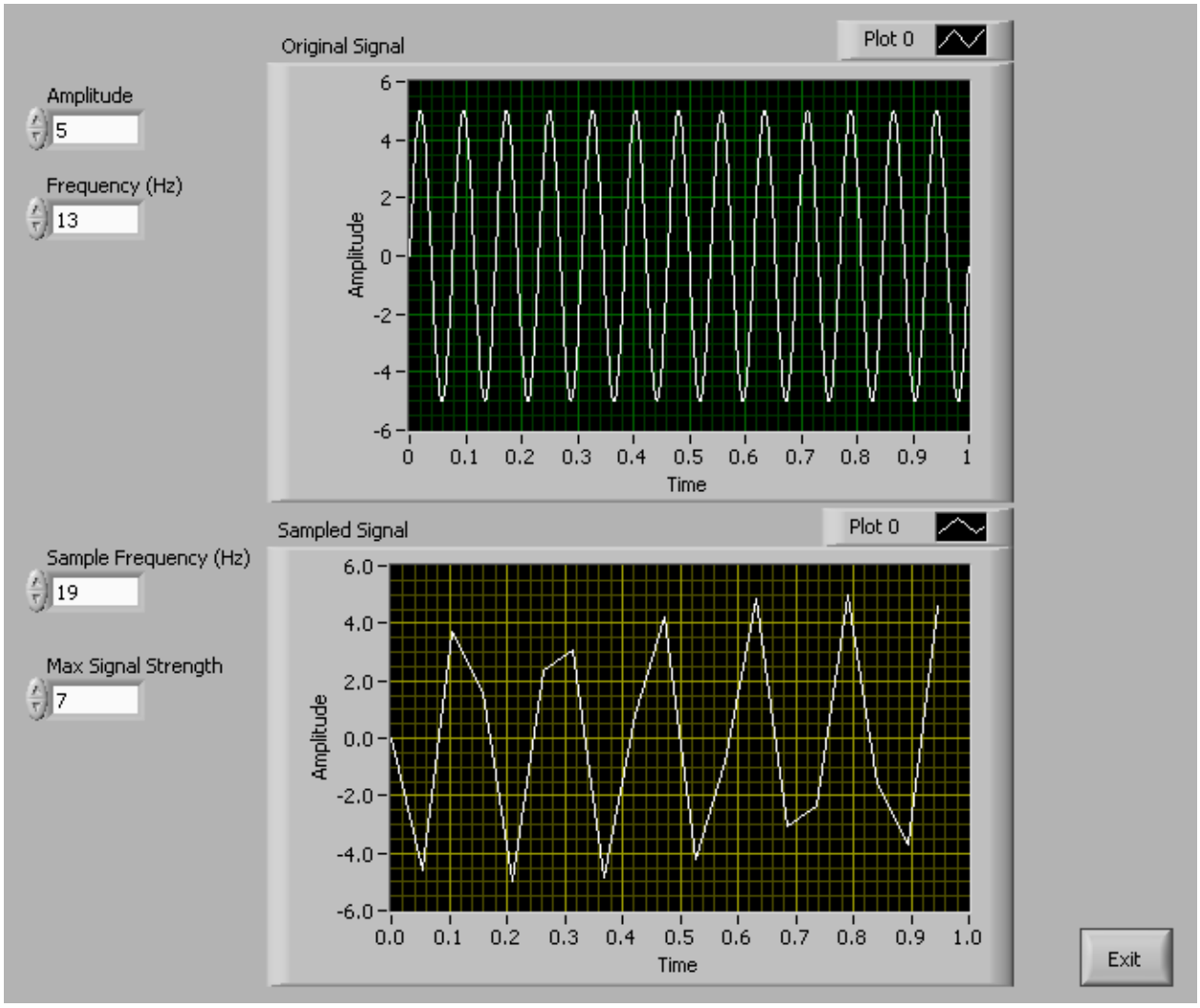

Figure 11: More Aliasing. The sampled signal frequency does not match the original signal.

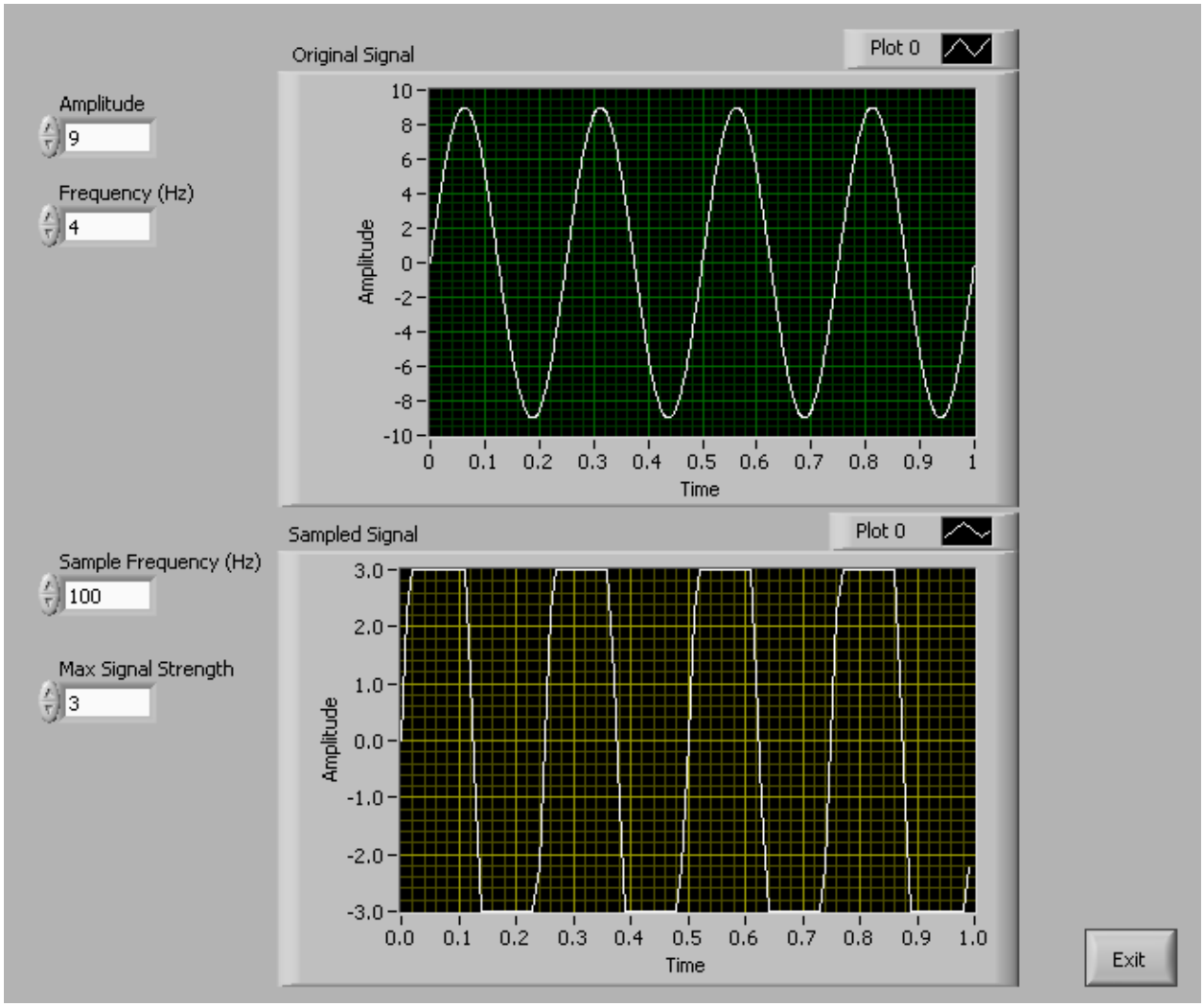

Figure 12: Saturation.