Lesson 17

Applying Graphical Operator Interfaces and Filtering

This lesson will apply operator interfaces to a variety of tasks such as using menus, creating status panels, importing bitmaps for panel backgrounds, creating a high-impact warning, and plotting via MATLAB®. This lesson will restrict itself to applying a few operator-interface fundamentals to customizing user interfaces. It consists of a pre-lab and five labs.

Lesson 17 Pre-lab Summary

The following are described in the Lesson 17 Pre-lab. See Appendix E, page E-85.

- Status Panel
- Bitmap
- Nested UserFunctions
- an overview of filter theory

Details of accessing and applying the operator interfaces are provided in Appendix C.

The filter-theory material in Appendix E is included for those of you who design analog and digital filters. Included with VEE Pro is a special, pre-devised program for Digfilt – a Third Order Elliptical filter. As noted in Lesson 1, Appendix B includes a cross-reference to each of these items and to all objects and subprograms in the labs of this and later lessons.

Overview

Lab 17.1 – Creating a Status Panel
This lab will show you how to create a status panel that provides test-in-progress information.

Lab 17.2 – Importing Bitmaps for Panel Backgrounds
This lab will show you how bitmaps can add impact to your test displays.

Lab 17.3 – Creating a High Impact Warning
This lab will show you how to nest UserFunctions.

Lab 17.4 – Exploring a Pre-Designed Digital Filter Program
This lab will show you how to explore the capability of a VEE Pro supplied digital filter program.

Lab 17.5 – Using MATLAB® to Display the Pre-Designed Digital Filter
This lab will show you how to use MATLAB® to graph the Agilent digital filter output waveform versus time for a variety of excitations.

This is the last lesson that will directly assist you in the development of programs. The next, and final, lesson will assist you in learning how to improve your programs and devised tests, better analyze those tests, and present the results of those tests to others so they can decide with you what to do next.

Lab 17.1 – Creating a Status Panel

This lab will show you how to create a status panel and use this panel to provide you with status information regarding your in-progress test. This lab could be devised as a modular program (UserFunction) that could be called upon later.

1. Clear your Work Area, deselect the Program Explorer, and maximize the Work Area.

Creating a status panel for in-progress test

2. Select Menu Bar => Device => Sequencer; place it on the left-center of your Work Area; double-click on its transaction bar.
3. Leave the default name test1.
4. Replace the FUNCTION: field with random( ); click OK. See Figure 17.1.

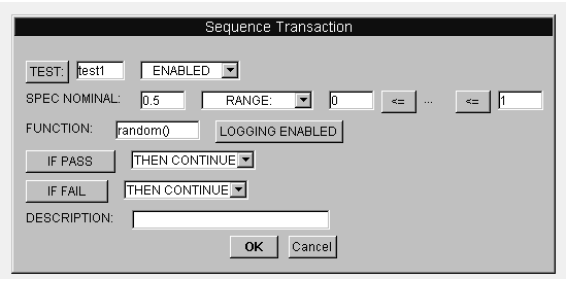

Figure 17.1. Configuring test1

5. Double-click in the Sequencer object on the white space below test1; configure a second and identical test named test2. Replace FUNCTION field with random(); click OK.
   Note: See Figure 17.1 again.
6. Open the Sequencer Properties box; choose the Logging tab.
7. Go to Logging Mode; set Log Each Transaction To:logTest(thisTest).
8. Click OK.
   Note: As the Sequencer executes each transaction, it will create a record for each test – "thisTest" – whose fields can be configured under the same tab. Then you can create a UserFunction such as

"logTest". The Sequencer will call your LogTest( ) UserFunction with the Record "this Test" at the end of each executed transaction. Your status panel will update itself automatically.

9. Select Menu Bar <= Device <= Function & Object Browser; select Type: Built-in Functions; Category: Panel; Member: showPanel.
10. Click on Create Formula; place it above the Sequencer.
11. Delete the five input pins; then edit the showPanel parameters within the parentheses to read: showPanel("logTest",420,180); reduce the Sequencer's vertical size.
12. Connect the Result (output) terminal from showPanel to the Sequencer input (top) terminal.
13. Select Menu Bar => Device => UserFunction; name it logTest.
14. Add a logTest input pin. (The logging Record "thisTest" will be the input.)
15. Select Menu Bar => Display => Logging AlphaNumeric; place it in the logTest UserFunction; deselect Clear at Activate; size it; connect it to the input pin as shown in Figure 17.2.

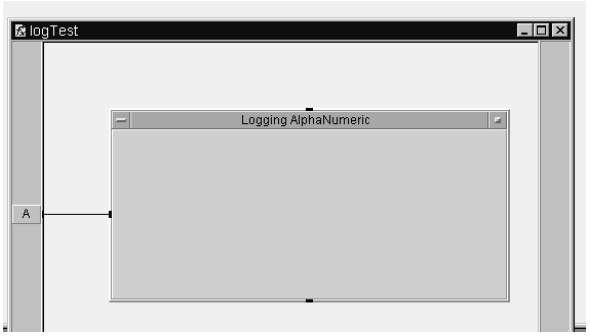

Figure 17.2. Detail view of the UserFunction logTest before running

16. Select the Logging AlphaNumeric Object and make it a panel: click Edit => Add to Panel.
17. In the Panel View, adjust the sizing and placement to satisfy yourself.
18. Deselect the display and Panel View Title Bars in the Logging Alphanumeric Properties box to simplify the later presentation. Run this program. See Figure 17.3.

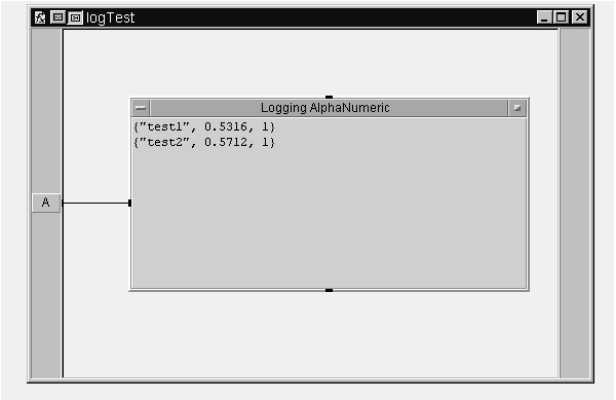

Figure 17.3. Detail view of the UserFunction logTest after running

19. Return to the Main window.
20. Select Menu Bar => Flow => Confirm(OK); place it below Sequencer.
21. Select Menu Bar => Device => Function & Object Browser; choose Type: Built-in Functions; Category: Panel; Member: hidePanel; click Create Formula; click OK.

    or

    Go to the Explorer window; right-click on the logTest function; Generate; HidePanel; an automatically labeled object will appear.
22. Place the Function & Object Browser below the "OK" button.
23 Change the hidePanel parameter (within the parentheses) to "logTest"; delete its data-input terminal.
24. Connect the objects as shown in Figure 17.4.

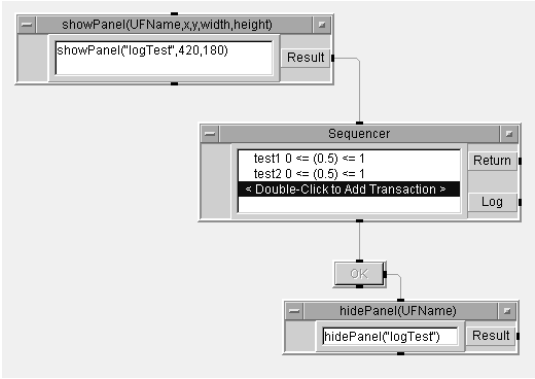

Figure 17.4. Main program before running

25. Save As… LAB17-1.
26. Run your program. It should look like Figure 17.5. Step through this program to see the status panel appear, update with the results from Test1, update with the results from Test2, then disappear.

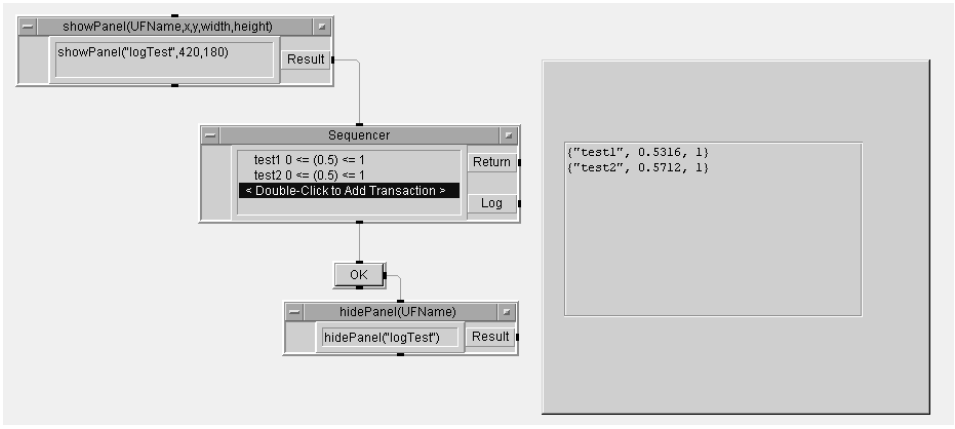

Figure 17.5. Main program after running with status panel

Note 1: What does the "1" in each logTest line represent? Change the limit to "2" and observe the result of a new run to see if it has any effect on the "1". Experiment with other variations to determine the meaning of the "1".

Note 2: You have used a Confirm OK object to trigger the hidePanel object; you could place it elsewhere in your program to time its execution.

◊ 27. Remove checkmarks from "Clear at Activate" and "Clear at PreRun"; run your program several times; observe the output display.

◊ 28. Add a checkmark to "Clear at PreRun"; run your program several times; observe the output display.

◊ 29. Remove checkmark from "Clear at PreRun"; add checkmark to "Clear at Activate"; run your program several times; observe the output display.

30. Save your final program again if you are satisfied with the ending checkmark status. Or else, close your program without saving to return to the original settings.

Lab 17.2 – Importing Bitmaps for Panel Backgrounds

This lab will show you how bitmaps can add impact to your test displays. Bitmaps can be imported for icons, for the Picture Object, or for the panel view backgrounds in a UserObject or a UserFunction.

1. Clear your Work Area, deselect the Program Explorer, and maximize Main.

Importing a bitmap for a panel background

2. Select Menu Bar => Device => UserFunction.
3. Select Menu Bar => Flow => Confirm (OK); select Menu Bar => Display => Label; place them in the UserFunction window.
4. Change the name of the UserFunction to Bitmap.
5. Select the OK and the Label objects; highlight them with a shadow.
6. Open the pop-up Edit menu by placing the pointer on the UserFunction Work Area and clicking on the right-hand mouse button.
7. Select Add to Panel.
8. Open the Bitmap object menu; select Properties.
9. Select Show Panel on Execute.
10. Deselect Show Title Bar under Pop-up Panel.
11. Open the Panel folder tab; change the Grid Size to 2.
12. Select default.gif and Scaled under Background Picture.
13. Click OK.
14. Open the Properties box for the Label object by clicking it with the mouse right-hand button; change the title to Bitmap Function.
15. Click Center Justify.
16. Open the Colors folder; select Light Gray for the Background of the label.
17. Click OK.
18. Re-open the Label Properties dialog box; open the Fonts folder; choose a larger font with bold type.
19. Select Automatically Resize Object on Font Change.
20. Open the Appearance folder tab; go to Border; choose Raised; click OK.
21. Convert (minimize) Bitmap to an object.
22. Go to the Main window.
23 Select Menu Bar =>Device => Call; go to its object menu and click on Select Function.
24. Choose Type: Local User Functions; Category: <All>; Member: Bitmap.

25. Click OK.
26. Save As… LAB17-2.
27. Run this program. The pop-up box should look like Figure 17.6.

Figure 17.6. The Bitmap Function

Note 1: To secure this program from alteration:

a. Create a panel view for your run-time version of the program.
b. Save the program so you have a copy you can alter.
c. Select Menu Bar => File => Create Run Time Version… .

Note 2: VEE will automatically use a *.vxe extension to indicate a secured version.

Lab 17.3 – Creating a High Impact Warning

This lab will show you how to nest UserFunctions. As an example:

- The first function will be the alarm, which will display a big red square and then beep.
- The second function will call the alarm to beep repeatedly. It will cause a blinking-light effect and a pulsing sound until the user turns off the alarm.

For a specific application of a warning, see previously presented Lab 14.4.

1. Clear your Work Area, do not deselect the Program Explorer, maximize Main.

Creating a high impact warning

2. Select Menu Bar => Device => UserFunction; change the name to alarm; click OK.
3. Select Menu Bar => Display => Beep; place it in the upper-left corner of the "Alarm" UserFunction.

4. Adjust the settings so you have a loud beep:
   Frequency (Hz): 1000
   Duration (sec): 1
   Volume (0-100): 100
5. Select Menu Bar => Display => Indicator => Color Alarm; place it in the upper-right corner of "alarm".
6. Open the Color Alarm object menu; click Properties.
7. Deselect Show Title Bar; click Layout: Rectangular.
8. Go to Limits; delete Mid Text and Low Text; click OK.
9. Select Menu Bar => Data => Constant => Real32; place it to the left of the green Alarm button; change its value to 1.
10. Connect the Real32 output (right) pin to the Color Alarm input (left) pin.
    Note: The Alarm will receive the value of **1** on its output which, at runtime, will change its color to red. You may customize the display to have different high and mid-limit values. You may even change those values at runtime via a control input.
11. Select Menu Bar => Flow => Delay; place it below Color Alarm; set it to 1; connect its sequence-input (top) pin to the Color Alarm sequence-out (bottom) pin.
    Note 1: This will synchronize the Beep object with the display on the screen.
    Note 2: On some pc-Windows machines, the Beep object is no longer supported. Thus, the duration and frequency in the Beep Objects will not work.
12. Select Menu Bar => Display => Note Pad; add the message:

    TURN OFF INSTRUMENTS!

    size the Note Pad to emphasize the instructions. See Figure 17.7.

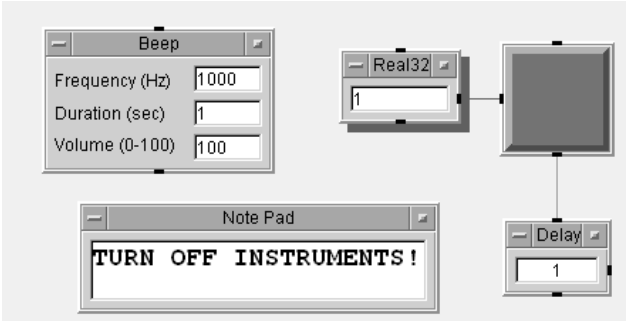

Figure 17.7. Detail view of the UserFunction: alarm

13. Go to the Main window.
14. Select Menu Bar => Device => Call.
15. Go to the Call Function object menu; choose Select Function.
16. Select Type: Local User Functions; Category: <All>; Member: alarm.
17. Click OK.
18. Save As… LAB17-3.

19. Test your program by running it. In the "alarm" UserFunction window, the Color Alarm button should now be red. Stay in the alarm window.
20. Select the Color Alarm display and the Note Pad.
21. Use the mouse right-hand mouse button; open the pop-up Edit menu; select Add To Panel; size and arrange the objects so the Note Pad is under the High (Color Alarm) object.
22. Open the Note Pad object menu; click Properties.
23. Go to Open View and deselect Show Title Bar; go to Editing and deselect Enabled; open the Fonts folder tab; go to Object, Text: deselect Default; select Arial Black, Size: 14; click OK.
    Note: Arial Black avoids the need to select the Font Style: Bold.
24. Select Automatically Resize Object on Font Change; click OK.
    Note: The Properties dialog box will automatically close.
25. Return to the Panel View; go to the Color Alarm Properties dialog box; select the Appearance folder; change the Border to: Raised
26. Repeat Step 25 for the Note Pad.
27. Double-click on the alarm Title Bar to get its Properties dialog box.
28. Select Show Panel on Execute; deselect Show Title Bar.
29. Click OK.
30. Convert (minimize) alarm to an icon.
31. Go to the Main window; delete the Call object.
    Note 1: VEE will still hold the function alarm in memory.
    Note 2: The following steps will cause the alarm function to be repeatedly called.
32. Select Menu Bar => Device => UserFunction; change the name of UserFunction to warning; click OK.
33. Select Menu Bar => Flow => Repeat => Until Break; place it in the upper-left corner of warning.
34. Select Menu Bar => Device => Call; place it below Until Break; change the Function Name to alarm.
35. Connect the Until Break data-output (right) pin to the sequence-input (top) pin of Call alarm.
    Note: The following steps will ask users if they want to turn off the alarm.
36. Select Menu Bar => Data => Toggle Control => Check Box; place it below Call alarm.
37. Right-click on Check Box; go to the Properties box in the resulting object; change its name to "Turn off alarm?"; go to Layout and select Scaled.
38. Go to Initialization; check Initialize at PreRun; set initial value to 0.
39. Go to Fonts; select Arial Black, size 14 for the Object Text; click OK twice.
40 Connect the Call alarm sequence-out ((bottom) pin to the Turn off alarm? sequence-in (top) pin.
    Note: When the user clicks the box, a checkmark will appear and the object will be set up to display: a 1. Otherwise, the object will be set up to display a 0. (The output can be tested with an If/Then/Else object to tell VEE what to do next.)
41. Select Menu Bar => Flow => If/Then/Else; place it the right of the Turn off alarm?.
42. Connect the Turn off alarm? data-output (right: Toggle) pin to the If/Then/Else data-input (A) pin.
43. Edit the expression in the If/Then/Else object (white space) to a==1.
    Note: If terminal A holds a 1, the Then output will fire. Otherwise, the Else output will fire.
44. Select Menu Bar => Flow => Repeat => Break; place it below and to the right of If/Then/Else; connect its top pin to the Then output pin of If/Then/Else. See Figure 17.8.

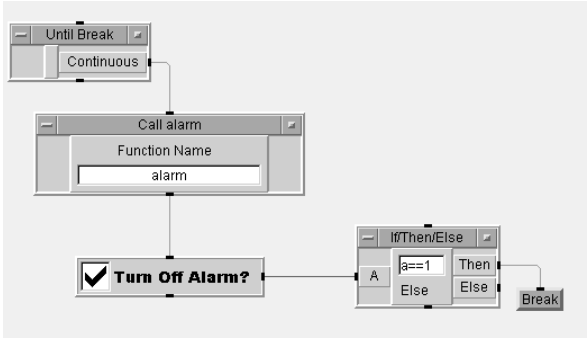

Figure 17.8. Detail view of the UserFunction: warning

45. Select the Turn off alarm? object by clicking on its right side.
46. Open the pop-up edit menu (by clicking the right-hand mouse button on the white area of warning); select Add To Panel.
47 Size the panel view to surround Turn off alarm?
48. Open the warning Properties box, select Show Panel on Execute, and deselect Show Title Bar; click OK. (The Title Bar does not need to be seen by the operator.)
    Note: You should now have three icons at the bottom of your screen: alarm, Main, and warning. Align them along the bottom of your screen if necessary.
49. Go to Main; Select Menu Bar => Device => Call; place it in the top-center of Main; open its object menu, click Select Function, select Type: Local User Functions; Category: <All>; Member: warning; click OK.
50. Minimize the Main window.
51. Run this program; reposition the alarm and warning panels if necessary. See Figure 17.9.

Figure 17.9. The Warning program

52. Save As… LAB17-3.

Lab 17.4 – Exploring a Pre-Designed Digital Filter Program

This lab will show you how to explore the capability of a VEE Pro supplied digital filter program.

Close your VEE program if necessary.

1. Go to the hard-drive location where the program files are stored; open the Program Files folder. (You may use Ctrl F as a means for locating these files.)

   or:

2. Go to the Help menu in VEE; select Open Example… => Applications to reach the Agilent digital filter program.

Exploring the Agilent digital filter program

3. Double-click on the digfilt.vee icon. The program is shown in Figure 17.10.

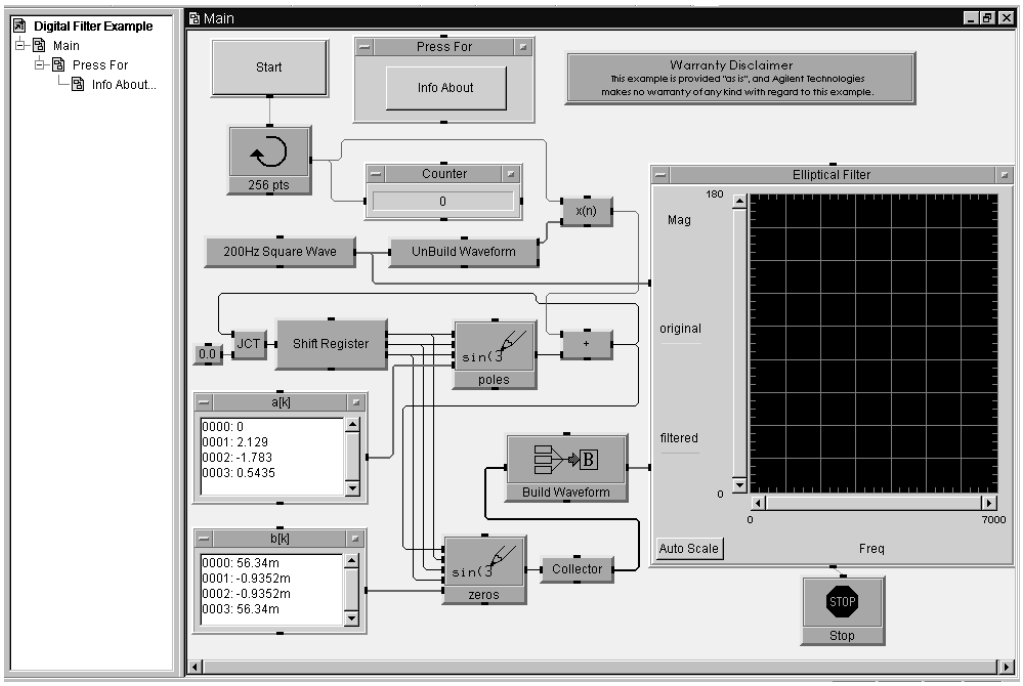

Figure 17.10. The digfilt.vee program

4. Run this program by clicking on its Start button. The display will show the frequency-magnitude distribution of a 200 Hz square wave in green and its filtered output superimposed in yellow.
5. Open the "200Hz Square Wave" object.

6. Run the program with each of the following waveform function settings:
   Sine or cosine
   Tri
   and either
   +Ramp or –Ramp.
7. Observe, for each of the above selected waveforms, the change in frequency and magnitude.
8. Return to the square-wave waveform selection.
9. Apply your knowledge of elliptical filter poles and zeros (see note 2 below), change the a[k] (pole) and b[k] (zero) values; start the program to observe the results on the frequency-domain display.
   Note 1: It may be necessary to select the display Auto Scale.
   Note 2: This step is designed for college students exploring filter design. Therefore, it may be skipped if you have no interest in this material.
10. Close this program without saving it unless you plan to use the results of your computations.
11. Rename the program to describe your chosen filter values if you so desire. Place it in whatever folder you prefer.
    Note 1: If you intend to design several filters, then create a new folder for these new programs.
    Note 2: You may prefer to adapt the fundamental elliptical-filter program to provide other types of filters. See Appendix E, page E-76.

Lab 17.5 – Using MATLAB® to Display the Pre-Designed Digital Filter

This lab will show you how to use MATLAB® to graph the Agilent digital filter output to show directly the harmonic numbers and their companion heights.

Open your VEE program.

1. Clear your Work Area, deselect the Program Explorer, and maximize Main.

Displaying the Agilent digital filter output using MATLAB®

2. Select Menu Bar => Device => Virtual Source => Function Generator; place it in the upper-left corner of Main.
3. Change the following Function Generator settings:
   Function: Square
   Frequency: 250
   Amplitude: 1.5
   Dc Offset: 1
   Num Points: 250
4. Select Menu Bar => Data => Unbuild Data => Waveform; place it to the right of the Function Generator.
5. Select Menu Bar => Display => Waveform (Time); place it below the Function Generator; change the following:
   Title Bar to Unfiltered Waveform
   Mag values to Maximum: 3, Minimum: -1, turn Automatic Scaling Off.
6. Connect the Function Generator output (Func) to the Unbuild Waveform input and to the Unfiltered Waveform (Trace1).

7. Select Menu Bar => Device => MATLAB Script; place it to the right of Unbuild Waveform.
8. Change the title of the MATLAB Script input terminals to “data” (top) and “time” (bottom). Connect these two terminals to the output terminals of Unbuild Waveform.
9. Expand the MATLAB Script object to accept the following code in its expression space:

```
%Filter Waveform
a=[2.129,-1.783,0.5435];
b=[56.34,-0.9352,-0.9352,56.34]*1e-3;
X=filter(b,a,data);
t=[0:length(data)-1]*time/length(data);

%Plot
plot(t,X);
xlabel('time');
ylabel('amplitude');
title('Filtered Waveform');
```

   Adjust the white space to display all of this code.
10. Delete the MATLAB Script output terminal.
11. Select Menu Bar => Flow => Confirm (OK); place it below MATLAB Script; change its title bar to: Done.
12. Connect the Done button to the bottom terminal of MATLAB Script. See Figure 17.11.

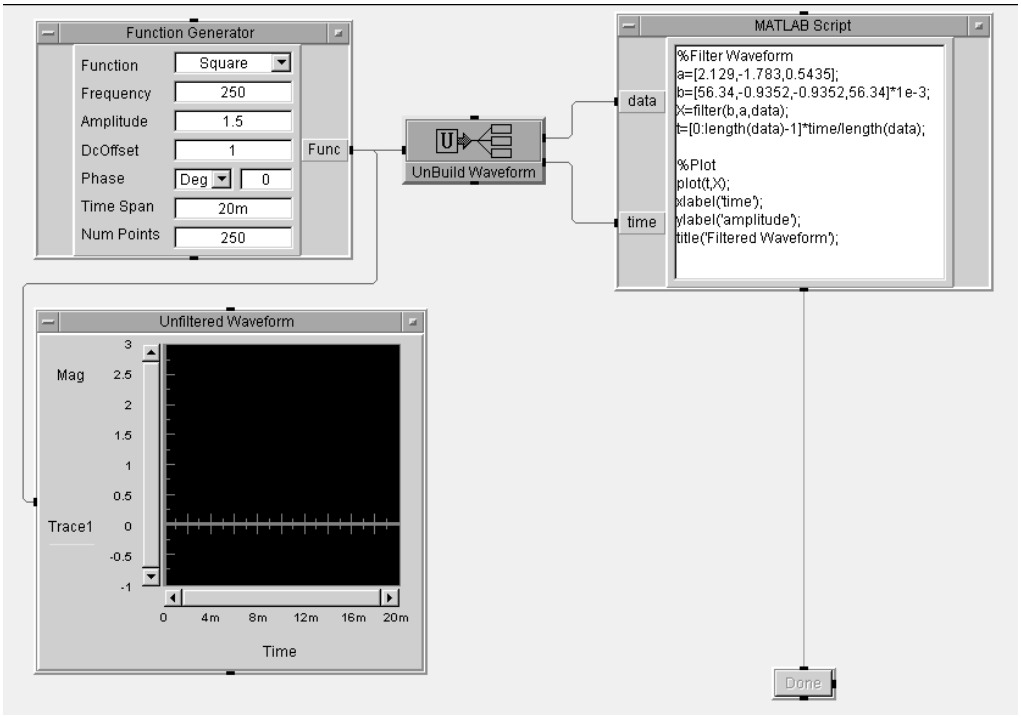

Figure 17.11. LAB 17.5 before running

13. Save this program as LAB 17-5.
14. Run this program. See Figure 17.12 where the MATLAB Figure No. 1 has been sized and placed next to the unfiltered waveform so these two waveforms may be easily compared.
    Note 1: Compare the two waveforms. The filter causes a change in slope of the waveform, a shift in the waveform's vertical location (offset), and a change in its magnitude. In a real-life situation, an operational amplifier would be added by the designer to correct these vertical-value changes.
    Note 2: If the MATLAB® program is installed on your computer, then it is possible to open the MATLAB Command icon at the bottom of your screen. In the space provided, additional programming may be entered. For further information, visit: www.mathworks.com.

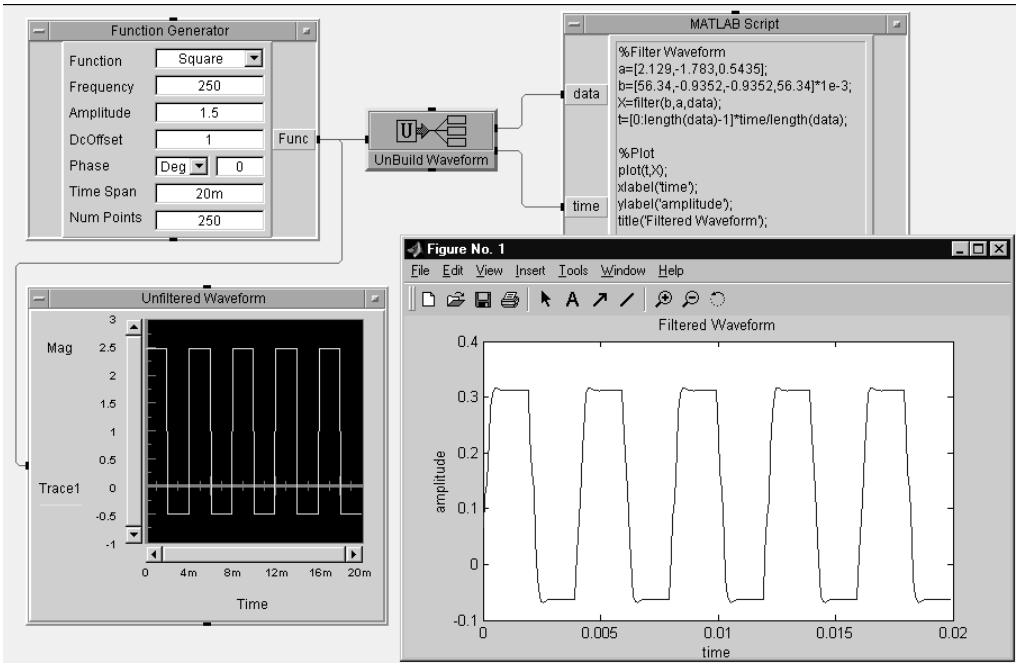

Figure 17.12. LAB 17.5 after running

15. Change the Function Generator setting to DcOnly; run this program again.
16. Compare the filtered and unfiltered waveforms; see Figure 17.13.
    Note: Time is required for the Dc value to reach its final value. In the process, the number of poles and zeros within the filter design will cause an overshoot from the filter's output final value. Again, an operational amplifier could be added to correct only the location of the final value. See Appendix E for additional detail on filter design.

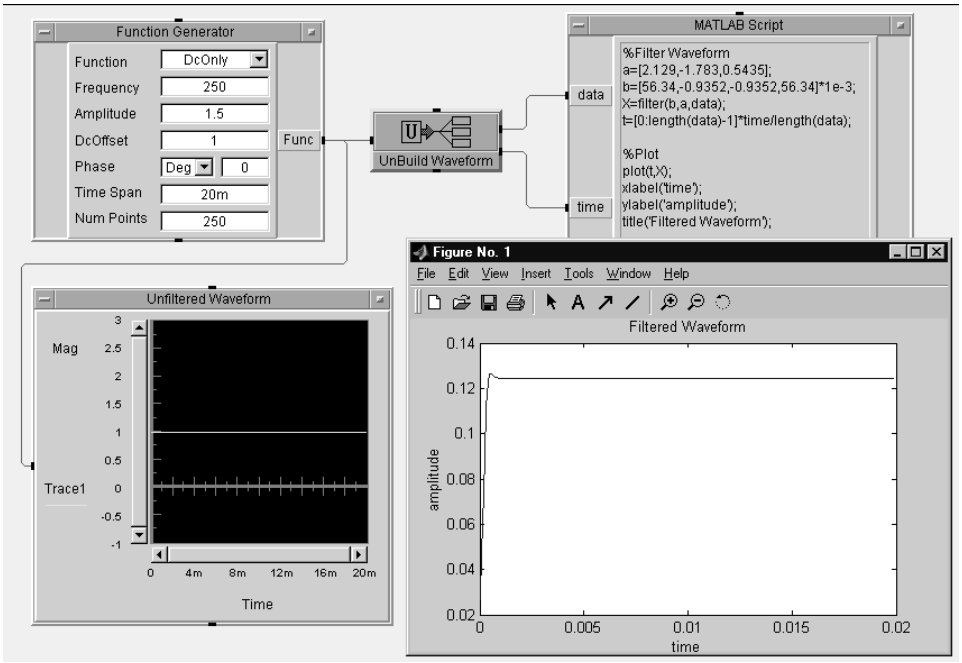

Figure 17.13. A filtered Dc offset

◊ 17. Examine the effects on the filter design for the following Function Generator functions: sine, cosine, triangular (tri), +Ramp, and -Ramp.

18. Compare these waveforms to the square-wave; write your notes in the following table:

Waveform	Change in Amplitude	Change in Offset	Other
Square			
Sine			
Cosine			
Tri(angular)			
-Ramp			
DcOnly			

19. Close this program without saving it.

Lesson 17 Summary

VEE Pro is a very versatile programming approach. New vendor-developed ActiveX controls are becoming available on an almost monthly basis.

In this lesson you learned to apply operator interfaces to a variety of tasks such as using menus, creating status panels, importing bitmaps for panel backgrounds, creating a high-impact warning, and using an ActiveX control. This is the last lesson that will directly assist you in the development of programs.

Lab 17.1 showed you how to create a status panel and use this panel to provide you with status information regarding your in-progress test. This lab can also be devised as a modular program (UserFunction) that could be called upon later.

Lab 17.2 showed you how bitmaps can add impact to your test displays. You learned that bitmaps can be imported for icons, for the Picture Object, or for the panel view backgrounds in a UserObject or a UserFunction.

Lab 17.3 showed you how to create a high impact warning.

Lab 17.4 showed you how to use the Agilent pre-designed filter program.

Lab 17.5 showed you how to use MATLAB® to graph the Agilent digital filter output waveform versus time for a variety of excitations.

If you are using a DAQ card, then there would be five steps to follow when developing a program:

1. Identify and define your objectives.
   Examine the characteristics of the sensors that you will use to monitor your equipment.
2. Expand your objectives to include specific parameters that you plan to measure.
   Select your sensors; identify the VEE Pro icons that will provide their simulation.
3. Identify the VEE Pro Objects that will process your specific parameters.
   Write the program that applies these objects and the hardware simulators.
4. Examine the hardware that you plan to monitor and the related plug-in DAQ cards.
   Select those DAQ cards; assure yourself that there is a slot for them in your computer.
5. Validate your program after you have attached the hardware-access cards.
   Include in your program the ability to prepare Excel™ and Word™ hard copy.

There will be variations to the above steps; these variations will depend upon the program that you want to develop, test, apply, and evaluate.

You are now ready to learn how to improve various aspects of VEE Pro programs.