TEACHER INFORMATION

Projects: Build a Sensor Control Program

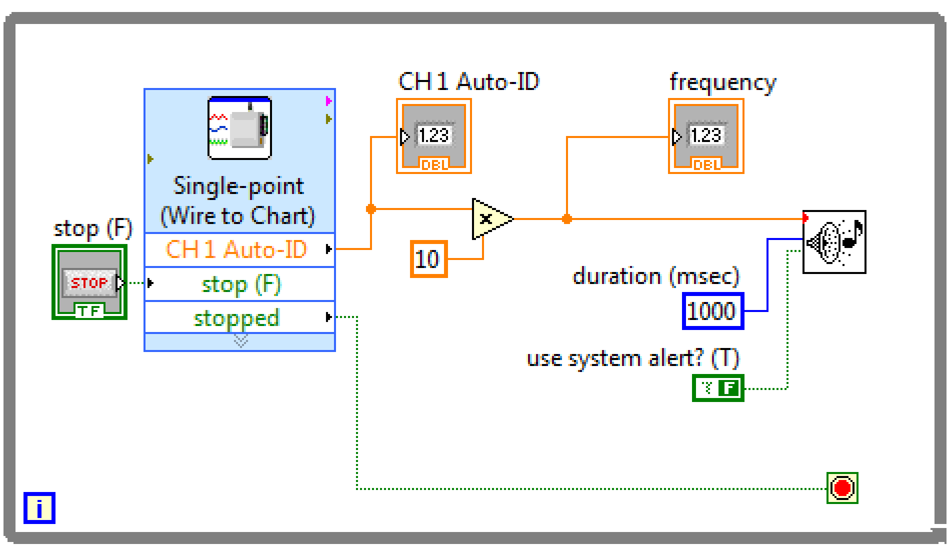
The project section of this book includes two separate challenges. These are fun for students, and they should get a feeling of accomplishment when they get them working.

We consider the first challenge, Audio Feedback, the easier of the two projects. The second project may be the better option for the more advanced students or if you have extra time to allow for an additional project.

Encourage your students to work towards creating a simple and concise program. In addition, the code should be kept visibly clean and organized. Can you tell what the program is supposed to do with a quick look at the code? The front panel should provide the user with the necessary controls and indicators, and should also be clean and organized.

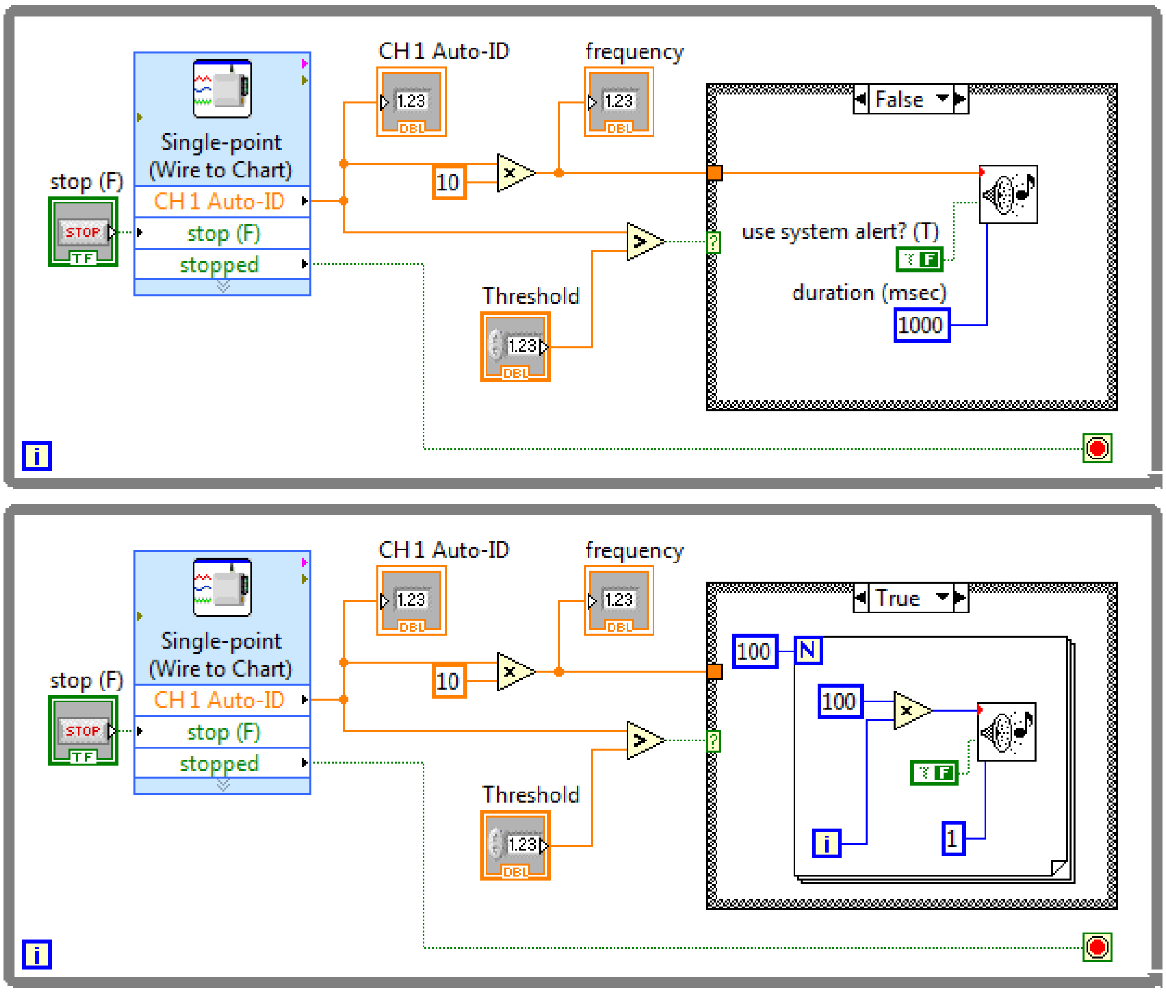
PROJECT 1 – AUDIO FEEDBACK

The Project1 Audio Feedback.vi file can be found on the accompanying CD. This program reads the Vernier Stainless Steel Temperature Probe in a While Loop, with the timing configured in the Analog Express VI set at a rate of 1 sample/second and a length of 120 seconds. The temperature reading is multiplied by 10 in order to create an appropriate frequency level. The result of the multiplication is input into the Beep.vi as the frequency value. The tone is held for a duration of 1000 milliseconds (1 second). After the tone has stayed on for a duration of 1 second, the While Loop iterates and the Express VI outputs another data point.



EXTENSIONS

Extensions to this program could include adding an upper level threshold value. If the threshold is met, a unique sound could signal a warning. An example program that creates a unique pulsing sound when the temperature is above a threshold is shown below. Two screenshots are provided in order to see both cases of the Case Structure.



Top: Audio Feedback program showing the Case Structure's False Case.

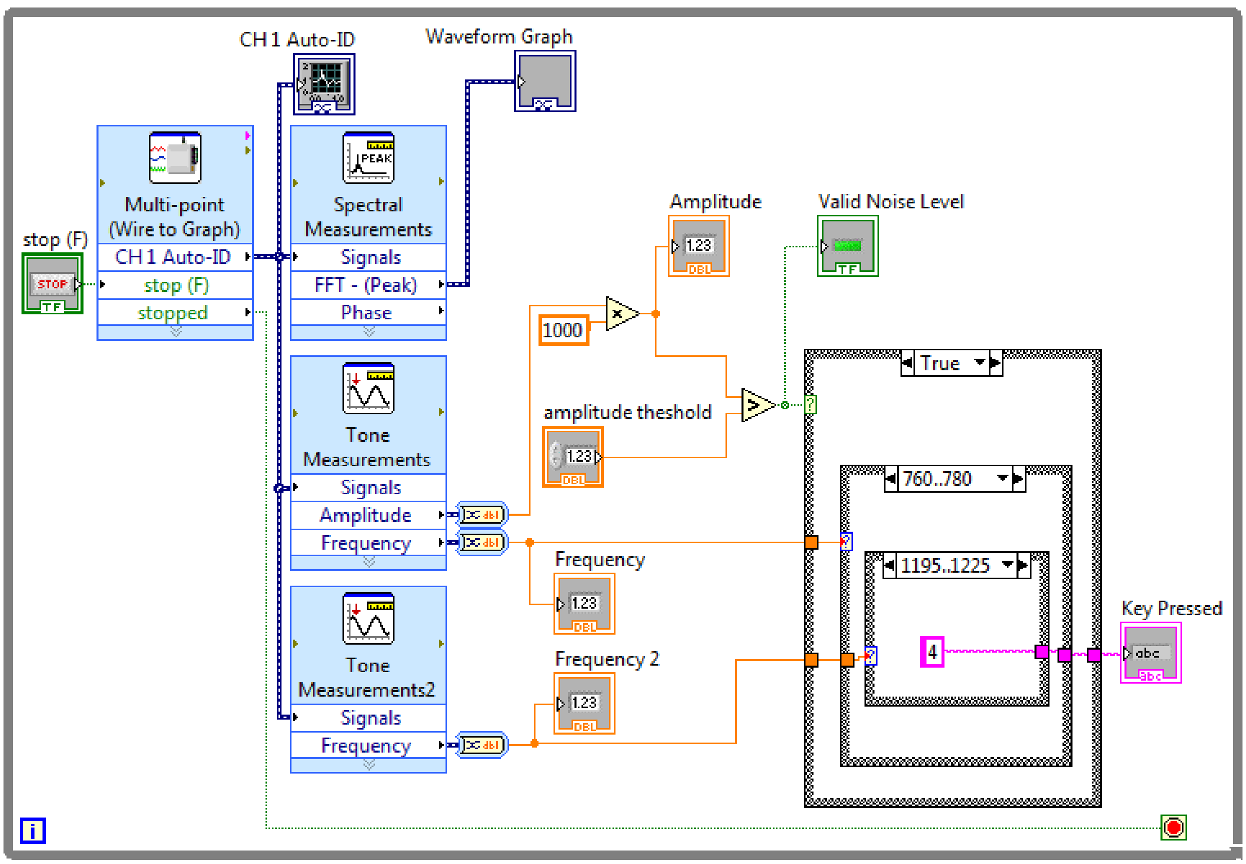
Bottom: Feedback program showing the Case Structure's True Case.

PROJECT 2 – CELL PHONE DECODER

This project is designed to decode the tone of a cell phone or tone dialer. In some classrooms, cell phones are not allowed; therefore, a LabVIEW program called Cell Phone.vi is included on the accompanying CD. This VI uses the computer's speakers to generate DTMF tones for buttons 1–9. And even if you will be using cell phones for this project, the Cell Phone VI might be of interest to the students.

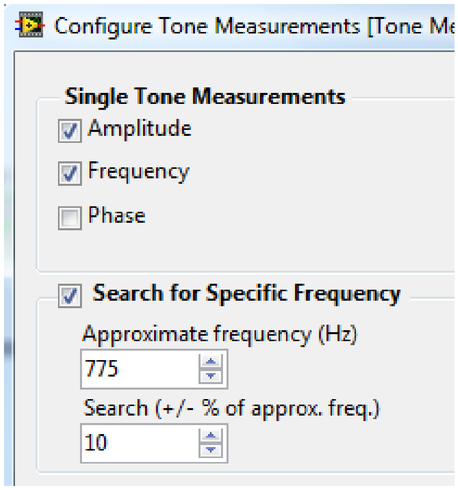
As is the case for many programming challenges, there are different ways to determine the correct number that is being pressed on the phone or tone dialer. One method is to create an FFT waveform, and then perform an analysis on this waveform. This waveform peaks at the location of fundamental frequencies. An analysis to determine where these peaks occur provides the low and high frequency values in the DTMF.

A second method is to use the Tone Measurements Express VI. This VI is described as being able to "find the single tone with the highest amplitude or to search a specified frequency range to find the single tone with the highest amplitude." A program that uses this Express VI to perform the analysis is described below.



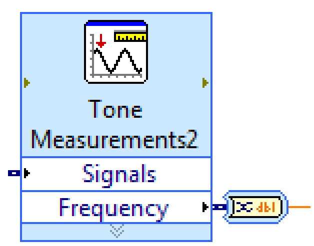
The Project2 DTMF Decoder.vi file can be found on the accompanying CD. This program reads the Vernier Microphone in a While Loop, with the timing configured in the Analog Express VI set to "Repeat" at a rate of 20,000 samples/second and a length of 0.2 seconds. The microphone signal from the Analog Express VI is sent to the Spectral Measurement Express VI. The resulting FFT is displayed on a front panel graph to provide the user with a visual representation of the frequency pairs. The microphone signal is also sent to two Tone Measurements Express VIs. These Express VIs analyze the waveform and determine the low and high frequency. These frequency values are sent to a nested Case Structure that determines the number that corresponds to the frequency pair and displays it to the user.

The analysis in this example program uses the Tone Measurements Express VI to locate the DTMF frequency pair (low frequency and high frequency). When the Tone Measurements Express VI's configuration dialog box is opened, there is an option to "Search for Specific Frequency".

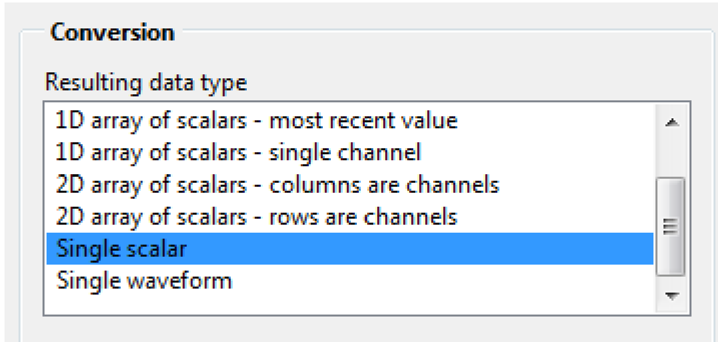


Checking this option allows you to place two instances of the Express VI in your program; one instance is configured to locate the low frequency (in the range of 697 to 852 Hz) and the other instance is configured to locate the high frequency (in the range of 1209 to 1477 Hz).

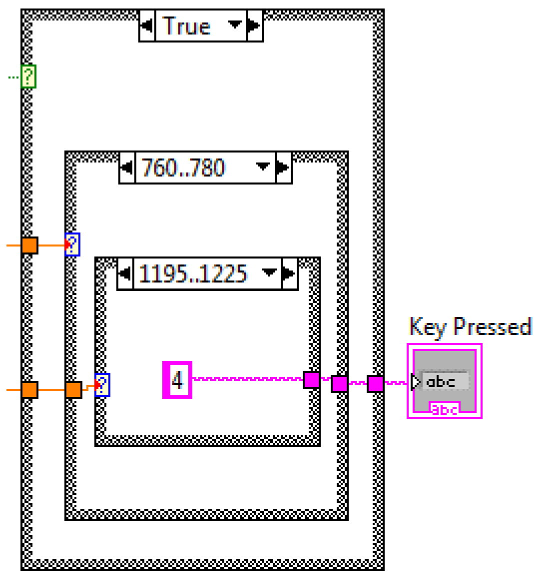
The next step of the program is to use the two frequency values to display the corresponding number 1–9. However, the frequency value sent from the Express VI must first be converted from a dynamic data type to a numeric data type. Express VIs commonly send results as dynamic data. This data type is very flexible; however the Case Structure does not accept this data type. Use the Convert From Dynamic Data Express VI to change the dynamic data to numeric data.



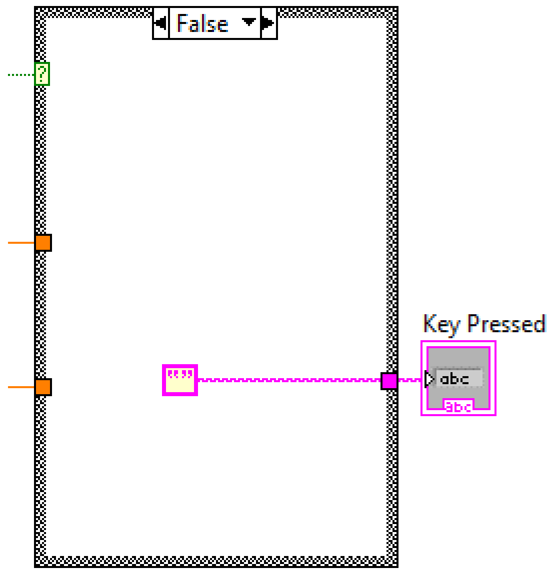
Be sure to double-click the Convert From Dynamic Data Express VI and select “Single scalar” from the “Resulting data type” section of the configuration dialog box.



The frequency values are now numeric data types and can be wired directly to a Case Structure to perform the decision making portion of the program. There are three nested Case Structures in this example, as shown in the figure below. Another option is to sum the high and low frequency values and use that value to determine the button number.



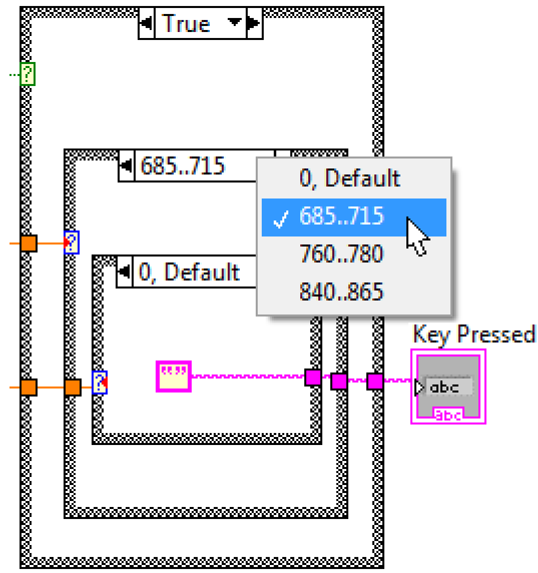
The outside Case Structure is used to determine if the tone is coming from a button press or if it is just white noise picked up by the Microphone (an advanced feature that is not stated as a design requirement). The Tone Measurements Express VI that is configured to find the low frequency is also configured to get the amplitude. This amplitude is compared to a user-defined threshold value using the Greater? Function. If the amplitude is not greater than the threshold value, the Greater? Function sends a Boolean False value to the outside Case Structure. If this is the case, an empty string value should be displayed to the user, and no code should be performed. This case is shown below.



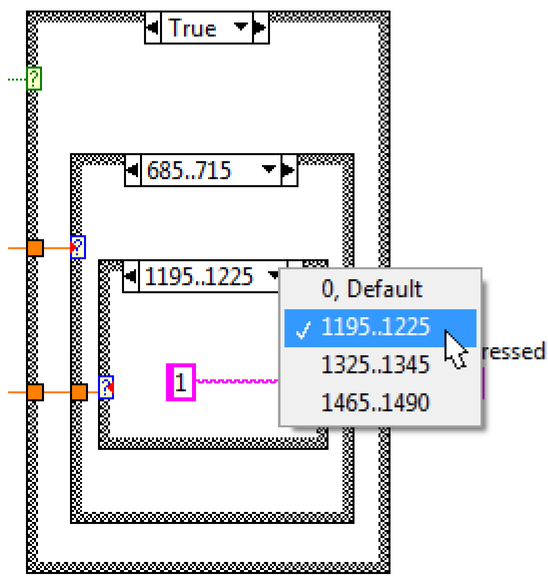
If the amplitude is greater than the threshold value, the Greater? Function sends a Boolean True value to the outside Case Structure. If this is the case, it means the signal is loud enough to be a button press, and the number 1–9 must be determined. The code inside the True Case includes two nested Case Structures to perform the decision making. The outside of these two Case Structures uses the low frequency value as the input value for the case selector; the inside Case Structure uses the high frequency values as the input value.

Remember that the low frequency value for numbers 1, 2, and 3 is 697 Hz. The value for numbers 4, 5, and 6 is 770 Hz. And the value for 7, 8, and 9 is 852 Hz. We cannot expect the result to exactly match these numbers, so the Case Structure is given a range. All three of these

low frequency values have a case, and a fourth default case is included for values outside of the three frequencies. If you click on the down arrow of the Case Structure, these four cases are shown in a shortcut menu, as shown below.



The low frequency is used to narrow the list to three numbers and the high frequency is used as a case selector value to determine the exact number. For example, if the user pressed a “1,” the low frequency is 697 Hz, and the outside Case Structure is sent to the “685..715” case. The high frequency for “1” is 1209 Hz, and the inside Case Structure is sent to the “1195..1225” case.



Inside this case is a string value of “1”. This value is sent to the “Key Pressed” indicator to display the number on the front panel to the user.

EXTENSIONS

1. Can the program be modified to decode what vowel is spoken by a user?

2. Shift registers are used within While Loops to transfer values from one loop iteration to the next. Use a shift register to store the previous button presses. When you press a new button, add that value to the string that was stored in the shift register in order to display all of the buttons that have been pressed.

3. Import an image of a cell phone onto your front panel and display the button press value on that image, as it would appear on a real cell phone.