Generate a Swept Sine Test Signal

Martin Rowe - October 01, 2000

Frequency response is a crucial specification in microphones, speakers, amplifiers, and other audio products. An easy way to determine frequency response is to feed a swept sine wave into your UUT and measure the unit’s output. If you have a data-acquisition system—a plug-in card or an external bus-based system (parallel, USB, IEEE 488, etc.)—you can easily program your system’s analog output to generate a swept sine wave.

Some programming languages designed for test and measurement have a swept sine function where you enter start frequency, stop frequency, and sweep time. If you’re using a general-purpose language, though, you’ll have to generate the swept sine wave point by point, then send the data to your data-acquisition system’s driver to load the data into your hardware.

To produce the swept sine wave signal, your code must generate an array of numbers that represent the swept sine’s amplitude at each sample point in the waveform. Unlike when programming a sine wave of a single frequency, you have to account for frequency change from one point to the next. Consequently, with the addition of each successive point, the array must cover a greater portion of a cycle.

The basic form of the equation that generates the array is

\[ y(i) = A \sin \left( \frac{\alpha i^2}{2} + \beta i \right) \]

where:

- \( y \) = amplitude
- \( i \) is an integer
- \( A \) = peak voltage amplitude
- \( \alpha \) and \( \beta \) are variables:

\[ \alpha = \frac{2\pi(f_2 - f_1)}{n} \]

\[ \beta = 2\alpha f_1 \]
where:

\[ n = \text{number of samples} \]
\[ f_1 = \text{normalized start frequency} \]
\[ f_2 = \text{normalized stop frequency} \]

The variables \( f_1 \) and \( f_2 \) are expressed in units of cycles per sample. To obtain the value of \( f_1 \), divide the sine wave’s start (lowest) frequency in hertz by the sample rate. To obtain \( f_2 \), divide the sine wave’s stop (highest) frequency by the sample rate.

You need a sample rate that will produce a reasonable representation of a sine wave at the stop frequency. You should use at least 10 samples/cycle at the stop frequency. Assume that you want to sweep the range from 100 Hz to 5 kHz in 3 s. At 5 kHz, you will need 10 samples, so the sample rate is 50 ksamples/s. (Be sure to check your data-acquisition system’s specifications to make sure it can produce the sample rate you need.) Having calculated the sample rate, you can then calculate \( f_1 \) and \( f_2 \) and enter them into the equations.

The value of \( n \), the total number of samples needed, depends on the sample rate and the sweep time. For a 3-s sweep at 50 ksamples/s, \( n = 150,000 \) samples.

Next, you need to calculate the signal’s amplitude at each point. Listing 1 contains code that generates a swept sine wave. The key to getting the frequency to increase with each sample is to place the index value in a for-next loop. Doing so will integrate the frequency so that the portion of a cycle covered by each point increases with each iteration of the loop.

![Figure 1](image)

**Figure 1.** An Excel spreadsheet proves that the sine wave’s frequency increases with each sample.

**Figure 1** shows the results of generating the array. In this example, I set the sweep time to 1 s, which generated an array of 50,000 points. The plot shows just the first 5000 points, which with a 1-s sweep still lets you see the increase in frequency. Plotting 150,000 points would cause the swept sine to appear aliased because of screen resolution.
I could have expanded the plot, but then it wouldn’t have fit on the screen. Because the cycle-to-cycle change in frequency is relatively small, you wouldn’t be able to see the frequency increase as easily. Setting the sweep time to 1 s lets you better see the increase in frequency in a plot that fits on the screen.

Listing 1 shows “hard coded” values for start frequency, stop frequency, sweep time, and amplitude. You may, however, choose to let the user of your application enter these values through text box sliders, dials, or radio buttons. Use radio buttons when you want to limit the user to just a few choices. Depending on your application, you may need to add a pause button to activate code that pauses the loop counter and holds the frequency until the pause button is released. T&MW

You can contact Martin Rowe at mrowe@tmworld.com.

Listing 1

Sub sweep_sine()
    Dim ampl As Single 'Vpeak
    Dim sample_rate As Integer 'samples/sec
    Dim samples As Integer 'samples for entire waveform
    Dim f_start As Integer 'start frequency in Hertz
    Dim f_stop As Integer 'stop frequency in Hertz
    Dim f1 As Single 'normalized f_start
    Dim f2 As Single 'normalized f_stop
    Dim pi As Double
    Dim sweeptime As Single 'sweep time in seconds
    Dim a As Double 'intermediate calculation
    Dim b As Double 'intermediate calculation
    Dim index As Integer 'sample counter
    Dim signal(index) As Double 'voltage for output to DAC driver ' set variables

    pi = 3.1415927
    f_start = 100
    f_stop = 5000

    ' need enough samples at high end to get a reasonable sine wave
    ' assume 10 samples/cycle at f_stop
    sample_rate = f_stop * 10 ' multiply sample rate by sweep time to get total number of samples
    samples = sample_rate * sweeptime ' normalize start and stop frequencies
    f1 = f_start / sample_rate
    f2 = f_stop / sample_rate 'calculate intermediate variables
    a = 2 * pi * (f2 - f1) / samples
    b = 2 * pi * f1 'generate array of voltages that form a swept sine wave
    For index = 0 To samples - 1
        signal(index) = ampl * Sin(a * index ^ 2 / 2 + b * index)
    Next index
    ' add your code to send array signal(index) to your driver
End Sub