Software library contains extensive DSP algorithms

Data Translation announced the release of the Signal Processing Library for .NET. This hardware-independent software library contains extensive DSP algorithms that can be seamlessly integrated into many measurement applications, such as sound and vibration.

```
//Sample code
using DataTranslation.SignalProcessing;

// Get a single-channel FFT interface
private SingleChannelFFT mSingleFFT = new SingleChannelFFT();

// Set the FFT Size. The number of spectral lines will be 1/2 this number.
mSingleFFT.FFTParameters.FFTSize = 1024;
// Set the Sampling Frequency
mSingleFFT.FFTParameters.SamplingFrequency = 5000;
// Set the Spectrum function to perform
mSingleFFT.FFTParameters.SpectrumFunction = SingleChannelSpectrumFunction.AutoSpectrum;
// Set the Window parameters
mSingleFFT.FFTParameters.Window.WindowType = WindowType.Hanning;

// Some Window types have additional parameters...
// If you are using a Cosine Taper window type, set the CosineTaperPercent property
if (mSingleFFT.FFTParameters.Window.WindowType == WindowType.CosineTaper)
     mSingleFFT.FFTParameters.Window.CosineTaperPercent = 50.0;
// If you are using a Force window type, set the ForceWindowStartPercent and ForceWindowEndPercent properties
else if (mSingleFFT.FFTParameters.Window.WindowType == WindowType.Force)
{
     mSingleFFT.FFTParameters.Window.ForceWindowStartPercent = 10.0;
     mSingleFFT.FFTParameters.Window.ForceWindowEndPercent = 90.0;
}
// If you are using an Exponential window type, set the ExpWindowFinalValue property
else if (mSingleFFT.FFTParameters.Window.WindowType == WindowType.Exponential)
     mSingleFFT.FFTParameters.Window.ExpWindowFinalValue = 0.1;

// Set the Scaling Type
mSingleFFT.FFTParameters.ScalingType = ScalingType.Linear;
// If you are using dB Scaling, set the DbReferenceValue property
if (mSingleFFT.FFTParameters.ScalingType == ScalingType.db)
  mSingleFFT.FFTParameters.DbReferenceValue = 10.0;
// Set the Scaling Value
mSingleFFT.FFTParameters.ScalingValue = ScalingValue.Peak;
// Set the Integration Type
mSingleFFT.FFTParameters.IntegrationType = IntegrationType.None;
// Set the Phase Noise Threshold Value
mSingleFFT.FFTParameters.PhaseNoiseThreshold = 0.000001;

FFTResultSingle fftResult = null;
try
{
   // Compute the FFT and return the result and intermediate data in a variable of type FFTResultSingle
   fftResult = mSingleFFT.ComputeFFTDetailed(buf);
}
catch (Exception ex)
{
    MessageBox.Show("Error computing FFT: " + ex.Message);
    throw;
}
return fftResult;
// Access the FFT results
result = fftResult.RealResult;
```

The Signal Processing Library for .NET contains native .NET object-oriented classes for performing single-channel and two-channel FFT operations and for calculating signal metrics, making this comprehensive library ideal for sound and vibration

application development. The open design architecture assists in rapid development of signal processing applications using measurement data from any device.**Pricing and availability**

The Signal Processing Library for .NET (SP8075-KEY) is available royalty-free under a download license for $995. It may be distributed widely as part of any user application.

http://www.datatranslation.com/products/test-and-measurement-software/Signal-Processing-Library/default.asp [1]

Links:
[1] http://www.datatranslation.com/products/test-and-measurement-software/Signal-Processing-Library/default.asp