SPECTRUM SIGNAL PROCESSING CAPABILITIES FOR EPICS

J. Hill, LANL

Abstract

With the Experimental Physics and Industrial Control System (EPICS) [1] an Input Output Controller (IOC) is configured using a network of modular process control building blocks called records. We describe a new fundamental signal processing building block for the system called the spectrum record which is to be implemented internally using the Chirp Z Transform algorithm. The performance and potential applications of the component are discussed.

INTRODUCTION

Currently, there appear to be no signal processing function blocks (records) for EPICS. Signal processing software tends to be CPU intensive and therefore, in the past, was considered to be less appropriate for critical EPICS front end controllers. However, processors are now much faster than when EPICS first appeared, with recent versions of the software it is now possible to run input output controllers (IOCs) on workstation platforms, and it is expected that CPU consumption issues are now easier to manage.

There are certainly many commercial sources for signal processing components, but given that common signal processing algorithms are not hard to implement, and because it appeared that integrating them into the core components of EPICS would be beneficial, then it was decided that a spectrum analyzer function block proof-of-principal should be implemented.

REQUIREMENTS

A general purpose spectrum analyzer should provide the magnitude of the real and complex frequency components of the input waveform within a specified frequency range. When configuring the function block, the user will provide the name of the input function block supplying the waveform to be analyzed, sample rate of this input waveform, starting frequency, stopping frequency, and number of samples within this frequency range.

THEORY OF OPERATION

The Discrete Fourier Transform algorithm produces a spectrum starting at zero Hz and increasing to one half of the highest frequency accurately represented by the input waveform following the Nyquist Theorem; if we require a higher resolution in the frequency domain than what the number of samples in the input waveform suggests, then we must zero pad additional elements at the end of the input waveform.

In contrast, the Chirp Z Transform algorithm [2] is more flexible. It produces evenly spaced samples along any contour in the Z plane. For example, when sampling on the unit circle in the Z plane we can obtain a spectrum within any frequency range meeting the constraints of the Nyquist Theorem, and with any sampling resolution desired within the specified range. This flexibility appears to make the Chirp Z Transform algorithm an excellent choice for implementing a general purpose spectrum analyzer with a high quality zoom, but this comes at a cost of reduced computational efficiency compared to the Discrete Fourier Transform. With the Chirp Z Transform algorithm we can obtain a higher resolution view of the spectrum while using a smaller number of spectrum sample points than are needed to obtain the same level of detail along the entire spectrum with the Discrete Fourier Transform algorithm. So at least in certain situations, reductions in CPU consumption resulting from a reduced frequency domain sample count might make up for losses in computational efficiency.

PRACTICAL APPLICATIONS

To test the software, a calculation function block was configured to sum the output of a random number generator with samples of a 0.5 hertz sinusoid and a 1Hz sinusoid. This calculation function block was updated at 10 Hz and its output was routed into the input of a function block implementing an accumulating ring buffer. The spectrum analyzer function block, also updating at 10 Hz, received its input from this ring buffer waveform. Figure 1 shows the resulting spectrum for a 0.5 Hertz sinusoidal input waveform. Figure 2 shows the resulting spectrum for a 0.5 Hertz sinusoidal input waveform summed with a 1 Hertz sinusoidal input waveform. Figure 3 shows the resulting spectrum for the two sinusoidal input waveforms summed with noise produced by a random number generator. We see the effect of both sinusoid components and also the DC component of the random number generator built into the EPICS calculation function block.

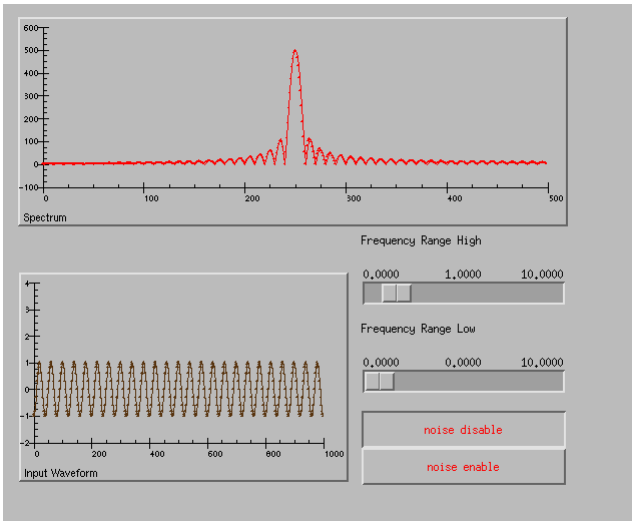

Figure 1: Spectrum of 0.5 Hz Sinusoid

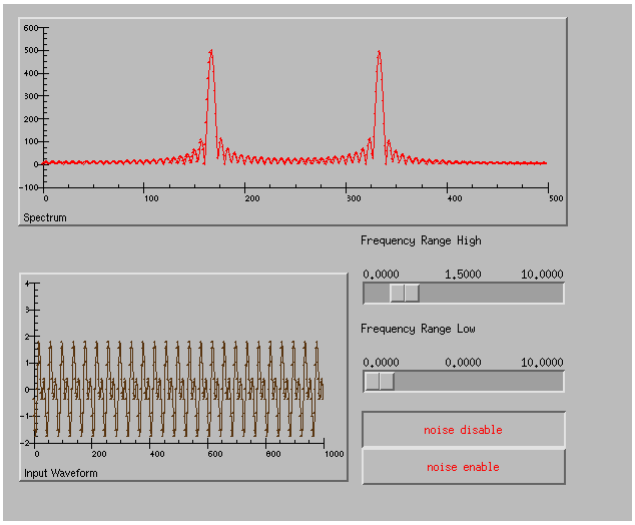

Figure 2: Spectrum of 0.5 Hz Sinusoid Summed with a 1 Hz Sinusoid

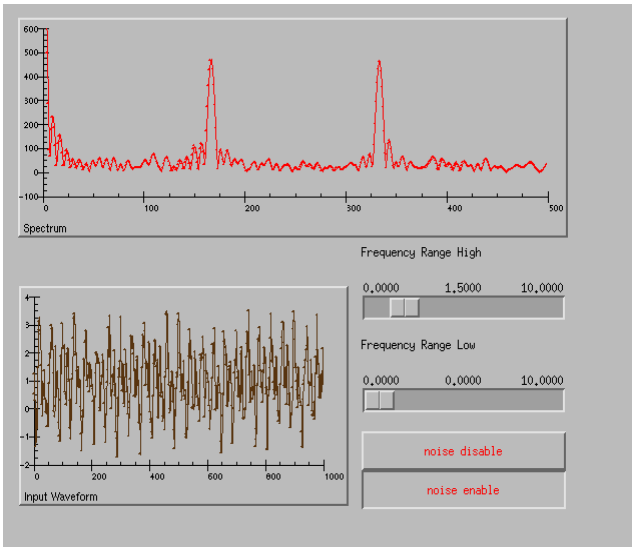

Figure 3: Spectrum of Sinusoid Components Summed with Noise

Figure 4 shows the CPU consumption measured with the test setup and a 10 Hz spectrum update rate. A dual processor, year 2000 vintage, PC was used for the test. When the test was changed to analyze a 500 element spectrum slice of a 1000 element input waveform the CPU consumption was below detection thresholds. Since a dual processor system was in use and all of the signal processing was accomplished by one thread then we can expect that 50% CPU consumption is approaching saturation. The code implements the Chirp Z Transform algorithm with the Fast Fourier Transform (FFT) algorithm and therefore an (N+M) times $\log_2(N+M)$ performance curve is expected where N is the number of input samples and M is the number of frequency domain samples, but this behaviour is not immediately apparent in figure 4. A possible explanation is that since a power of two based FFT was used in the implementation then there were step functions in the performance curve related to what power of two contains the sum of N and M, but this needs further investigation. Nevertheless, with proper choice of sample counts and scan rates, performance appears to be acceptable in a front end controller context.

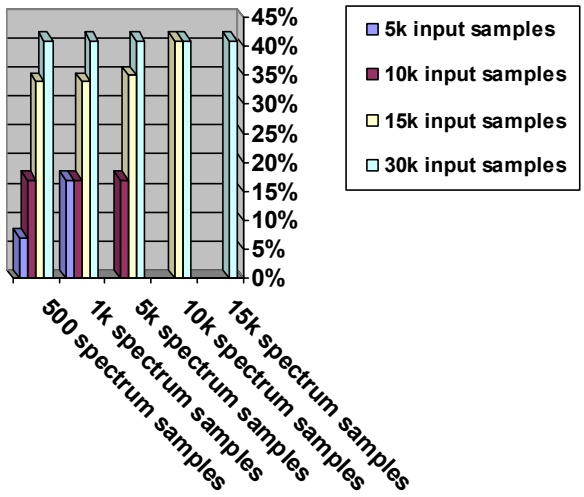

Figure 4: CPU Consumption of 10Hz Spectrum Updates

CONCLUSIONS

A general purpose spectrum analyzer function block employing the Chirp Z Transform algorithm was implemented for EPICS. The software appears to operate correctly in a simple test configuration, and the performance appears to be reasonable for use in carefully selected front end controller contexts. The new function block will soon be available for use by the EPICS community.

REFERENCES

[1] http://www.aps.anl.gov/epics/
[2] "Digital Signal Processing", Chapter 6, Oppenhiem and Schafer, Prentice-Hall