A Note On The FFT

Bruno Korst - bkf@ece.utoronto.ca

Abstract
This document is a supplement to the introductory lab, pertaining specifically to the Fast Fourier Transform (FFT). It may be valuable to better understand future experiments, in which students will model frequency domain scopes.

Keywords
Frequency Domain — FFT — bins — resolution

Building your own FFT display

This note provides a brief explanation on how to display and interpret the frequency domain representation of a signal in `Simulink`. It is not meant as a theoretical explanation of the Fast Fourier Transform (FFT).

The FFT is an algorithm which computes the Fourier Transform (i.e., a transformation from time domain to frequency domain) very fast. It is by nature a block computation, meaning that it is performed on blocks, or frames, of data. Therefore, it makes *no sense* to try to compute the FFT of a single sample. The input to an FFT is a block of samples and the output from the FFT is also a block of data representing the frequency domain in complex format.

When using the FFT in `Simulink`, you must gather a number of samples to input to an `FFT` block. This is done by "buffering" the samples, or "storing" them somewhere in memory. You must first determine which frequency resolution you will require and from that you will decide how many samples to store or buffer in. You can think of frequency resolution as how *fine* you want the distance between adjacent frequency points to be at the output. The more points you use to calculate an FFT, the finer your resolution will be.

If for some reason you buffer in a smaller number of samples than the size of the FFT you are about to perform, `Simulink` will pad the data with zeros prior to the FFT operation. This is done to extend the buffered samples up to the number of points you have chosen for the FFT. This padding, however, does not bring any additional information about your input signal, so the output produced is (still) solely based on the information contained in the samples of the input. Your best result is when you buffer as many input samples as the number of points of the FFT you will perform. For the laboratory experiments, we will use 1024 points as a general size.

The output of the FFT will be a block, or frame, of complex values. If you chose 1024 points for the FFT, you will have 1024 complex values as the output of the operation. These will represent magnitude and phase of each frequency "bin" (or point). Remember that now you are in the frequency domain; the first output point represents magnitude and phase at DC (or 0 Hz) and the last point represents magnitude and phase at the sampling frequency. In the lab exercises this last value will likely be 48KHz. This is to say that the frequency resolution obtained is 48,000 divided by 1024, or that there is 46.875 Hz between two adjacent frequency points. Frequencies which do not fall on integer multiples of 46.875Hz will not be represented precisely, resulting in "spectral leakage".

Calculating the magnitude value is not really straightforward. If the input to the FFT is a sinusoid whose frequency falls exactly on a multiple of the frequency interval above, the 1024 point FFT of a single sinusoid will present two peaks [1]. The amplitude of the two components is the amplitude peak of the sinusoid, multiplied by the number of FFT points divided by two. If, however, the frequency of this sinusoid falls inbetween points (or "bins"), the precise calculation of the magnitude is a more involved process, as there will be spectral leakage.

To summarize now, you know you should buffer data prior to performing an FFT with at least as many samples as the number of points of the FFT, you know that 1024 points will be your FFT length and that you should try to display the output of the FFT from 0Hz (DC) to 48KHz (Fs), realizing that this display is mirrorred around Fs/2. For all lab exercises, you can use these numbers and they will yield a good picture. The procedure used in Lab01 will show you how to simulate your own spectrum analyzer.

QUESTION For a $1V_{pp}$, 1KHz and a $1V_{pp}$ 1.5KHz sine waves sampled at 48KHz, why are they displayed differently in the frequency domain if both have the same FFT length of 1024? This is an example of the spectral leakage briefly explained above, when the frequency of the input sinusoid does not fall on an integer multiple of the desired resolution (in this case 48,000/1024).

Specific details on the `Simulink` blocks, their location and setup will be given in the outline for the first experiment.

[1] remember, the output is made of two mirror images. For a single frequency you will see an impulse between DC and Fs/2 and another between Fs/2 and Fs