Hideo Okawara's
Mixed Signal Lecture Series

DSP-Based Testing – Fundamentals 52
2 FFT in 1

ADVANTEST Corporation
August 2013

Preface to the Series

ADC and DAC are the most typical mixed signal devices. In mixed signal testing, an analog stimulus signal is generated by an arbitrary waveform generator (AWG) which employs a D/A converter inside, and an analog signal is measured by a digitizer or a sampler, which employs an A/D converter inside. The stimulus signal is created by using a mathematical method, and the measured signal is processed with mathematical method, extracting various parameters. It is based on digital signal processing (DSP) so that our test methodologies are often called DSP-based testing.

Test/application engineers in the mixed signal field should have thorough knowledge about DSP-based testing. FFT (Fast Fourier Transform) is the most powerful tool here. This corner will deliver a series of fundamental knowledge of DSP-based testing, especially FFT and its related topics. It will help test/application engineers comprehend what the DSP-based testing is and assorted techniques.

Editor's Note

For other articles on this series, please visit the Advantest web site at
http://www1.verigy.com/ate/news/newsletter/index.htm

Preface

In our mixed signal tests, the FFT (Fast Fourier Transform) is the most frequently used tool to analyze measured signals. The input data of the FFT is usually a measured signal waveform, while the output data is the frequency spectrum of the signal. The frequency spectrum is a complex number array, but the measured signal is always a real number array. ("Real number" means non-complex number or it does not have imaginary component.) From the mathematical point of view, the FFT is compatible to complex number input. Since measured signal data is always real number, the imaginary part of the input container is empty or rather zero. So the imaginary field can be utilized for another measured signal data area actually. Consequently, a single FFT routine can process two sets of independent signal data simultaneously, generating two sets of frequency spectrum. Of course, the output spectra is mixed together, so it should be definitely separated from each other. How much processing time can we save in this scheme? This is the motivation for this experimental report about the 2 FFT in 1. However, the reality is.

Waveform and Frequency Spectrum

Let's describe a sinusoidal waveform as $A_k\cos(k\omega t+\theta_k)$, which can be expanded by using the Euler's formula as follows;

$$\begin{aligned} A_k\cos(k\omega t+\theta_k) &= A_k\cos\theta_k\cdot\cos k\omega t - A_k\sin\theta_k\cdot\sin k\omega t \\ &= A_k\cos\theta_k\cdot\frac{e^{jk\omega t}+e^{-jk\omega t}}{2} - A_k\sin\theta_k\cdot\frac{e^{jk\omega t}-e^{-jk\omega t}}{2j} \\ &= \frac{A_k}{2}(\cos\theta_k + j\cdot\sin\theta_k)e^{jk\omega t} + \frac{A_k}{2}(\cos\theta_k - j\cdot\sin\theta_k)e^{-jk\omega t} \end{aligned} \quad (1)$$

Amplitude, Angular Frequency, Phase

Equation (1) means that a waveform can be expressed with a combination of the positive frequency component $e^{jk\omega t}$ and the negative frequency component $e^{-jk\omega t}$. The coefficients to $e^{jk\omega t}$ and $e^{-jk\omega t}$ correspond to the frequency spectrum of the waveform, which are complex conjugate each other.

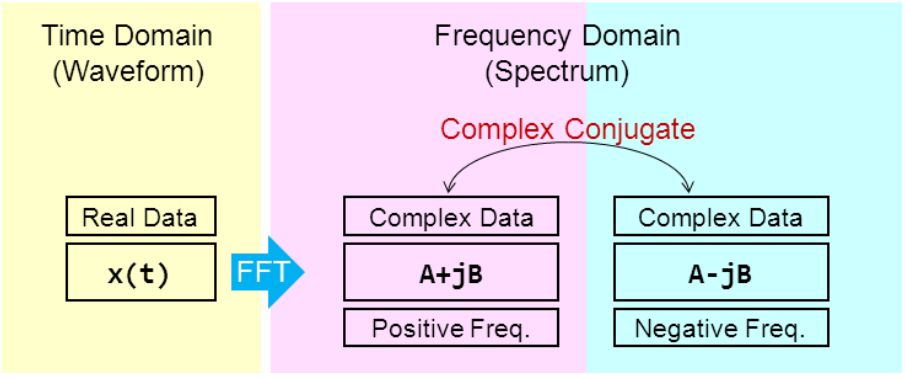

Figure 1: Waveform and Spectrum

The FFT routine processes a time domain waveform $x(t)$ which is a real number data, generating a complex conjugate pair of spectrum $A+jB$ and $A-jB$ as Figure 1 illustrates. Let's get 2 real waveforms $x(t)$ and $y(t)$ processed by the FFT respectively, generating 2 sets of complex conjugate spectrum pairs of $A\pm jB$ and $C\pm jD$ as Figure 2.

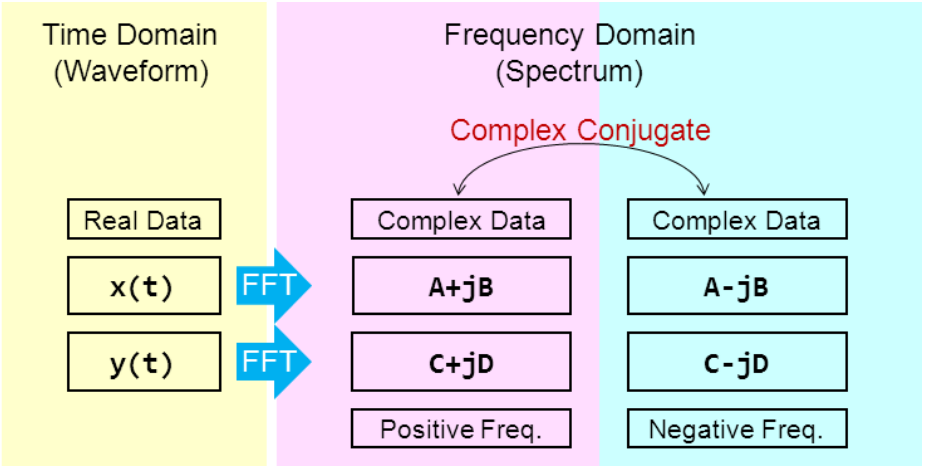

Figure 2: 2 Waveforms and 2 sets of Spectrum

When multiplying j to the waveform $y(t)$, the spectrum $C \pm jD$ is modified as Figure 3 illustrates. So the waveform $x(t)$ is real number data while the waveform $j \bullet y(t)$ becomes imaginary number data.

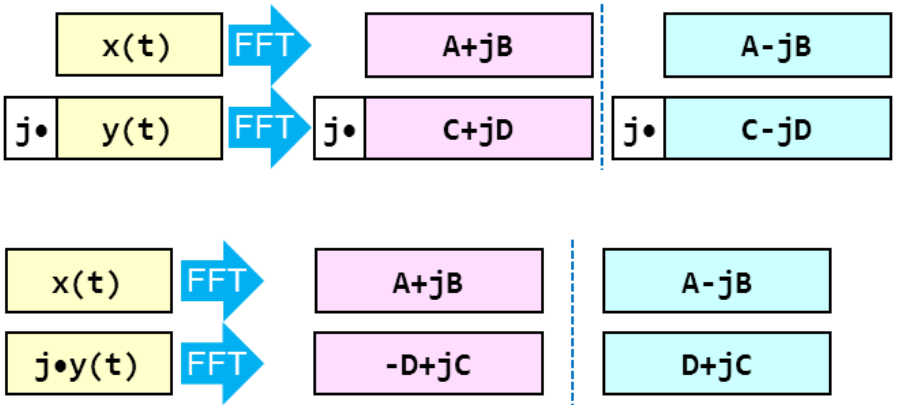

Figure 3: Multiplying « *j* » to Waveform *y*(*t*)

Consequently the waveform data $x(t)$ and $y(t)$ can be stuffed into a single complex number container, which can be processed by a single FFT routine.

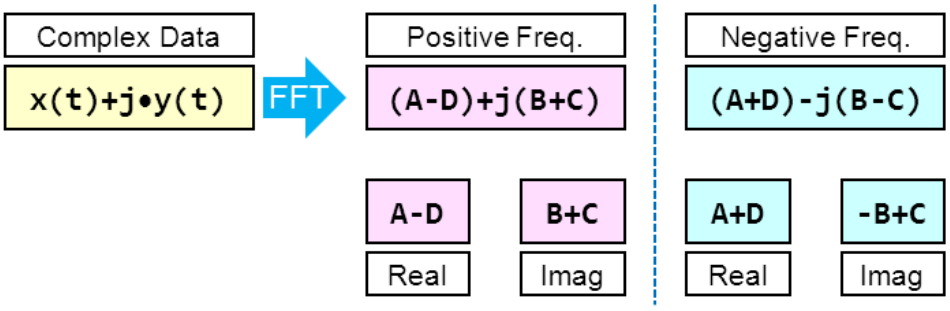

Figure 4: Waveforms *x*(*t*) and *y*(*t*) Processed by FFT

Then as Figure 4 illustrates, the FFT generates combined spectrum of $(A\text{-}D)+j(B+C)$ and $(A+D)\text{-}j(B\text{-}C)$. The positive frequency and the negative frequency components are separated as highlighted with colors. The real and the imaginary parts are already separated. Therefore the FFT result can be separated in four components $(A\text{-}D)$, $(B+C)$, $(A+D)$ and $(\text{-}B+C)$ as described in Figure

4. So manipulating the four terms each other as described in Figure 5, *A*, *B*, *C* and *D* can successfully be separated. Eventually you can reconstruct the frequency spectrum of *A*+*jB* for the waveform *x*(*t*) and the frequency spectrum of *C*+*jD* for the waveform *y*(*t*).

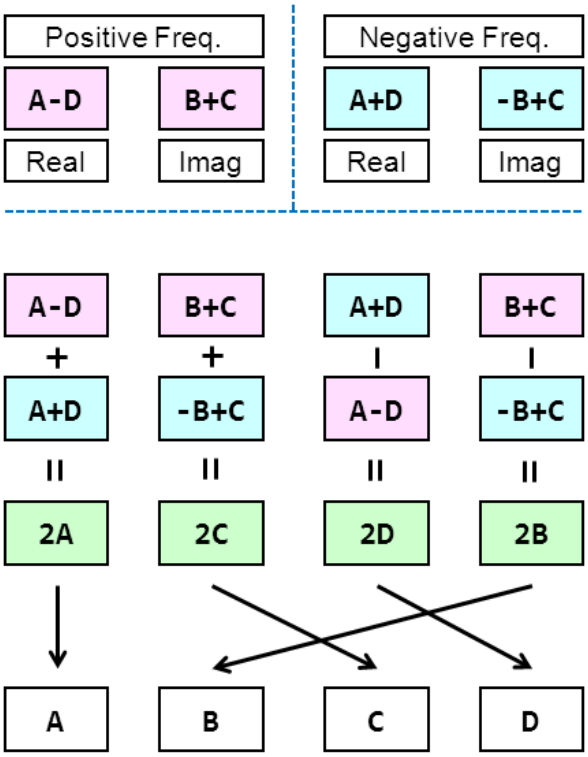

Figure 5: Spectrum Manipulation

Test Program

Let us look at the sample program code in List 1. When you have two sets of real number waveform data in the arrays of *dWave1*[] and *dWave2*[], you can assign one array to the real part and the other to the imaginary part of the input complex number container. Of course, two waveform arrays must have the same length. This will be performed from Line 31 through to Line 34. Then you perform the FFT processing once at Line 35, and play the data manipulation on the output data of complex spectrum as discussed in the previous section. Two sets of frequency spectrum *CSp1*[] and *CSp2*[] are reconstructed at Line 37 through to Line 49 which corresponds to the process Figure 5 represents.

```
10:
11:    int     i;
12:    int     Ndata;                  // # of Data
13:    int     Nsp = Ndata/2;          // # of Spectrum (Half Plain)
14:
15:    ARRAY_D       dWave1,dWave2;    // 2 Waveform Containers
16:    ARRAY_COMPLEX CWave,CSpect;     // Complex Number Waveform & Spectrum
17:    ARRAY_COMPLEX CSp1,CSp2;        // 2 Spectrum Containers
18:
19:    // dWave1[] & dWave2[] are supposed to hold waveform data.
20:
30:    CWave.resize(Ndata);
31:    for (i=0;i<Ndata;i++) {
32:       CWave[i].real()=dWave1[i];   // Waveform Data #1
33:       Cwave[i].imag()=dWave2[i];   // Waveform Data #2
34:    }
35:    DSP_FFT(CWave,CSpect,RECT);     // Only One FFT
36:
37:    CSp1.resize(Nsp);               // Spectrum Container #1 (Half Size)
38:    CSp2.resize(Nsp);               // Spectrum Container #2 (Half Size)
39:
40:    CSp1[0]=CSpect[0].real();       // DC of #1
41:    CSp2[0]=CSpect[0].imag();       // DC of #2
42:
43:    for (i=1;i<Nsp;i++) {
44:       CSp1[i].real()= CSpect[i].real()+CSpect[Ndata-i].real();
45:       CSp1[i].imag()= CSpect[i].imag()-CSpect[Ndata-i].imag();
46:
47:       CSp2[i].real()= CSpect[i].imag()+CSpect[Ndata-i].imag();
48:       CSp2[i].imag()=-CSpect[i].real()+CSpect[Ndata-i].real();
49:    }
50:
```

List 1: Sample Program Code for 2 FFT in 1

Since this program performs two spectrum analyses by executing the FFT only once, author expected that this processing could save time than the normal two-FFT situation – hopefully with nearly 200% efficiency. However, the reality was not so happy. The program List 1 took much longer time to complete than the normal two FFT.

There are two reasons in this disappointing result. The FFT routine in the V93000 system is already very well optimized. The time to execute DSP_FFT(Complex Array) on Line 35 was almost the same as the time to execute DSP_FFT(Real Array) twice. The FFT routine is designed to accept any one of an integer number array, a real number array or a complex number array. So when it receives an integer number waveform or a real number waveform, it does not seem to perform redundant processing from the beginning.

Moreover, the procedure in List 1 needs to perform extra processing for data distribution and separation. In order to make up a complex input data, the two waveform data had to be distributed into the real part field and the imaginary part field respectively. (Lines 30 to 34) After the FFT, the combined spectrum result had to be separated into two sets of spectrum. (Lines 37 to 49) The waveform size is usually 1K to 64K in most cases. It took some amount of time to execute those extra processing. Therefore, the "2 FFT in 1" sounds nice but the reality is not so good. The author is sorry that the readers will be disappointed to see the unexpected result.