Hideo Okawara's
Mixed Signal Lecture Series

DSP-Based Testing – Fundamentals 6
Spectrum Analysis -- FFT

Verigy Japan
October 2008

Preface to the Series

ADC and DAC are the most typical mixed signal devices. In mixed signal testing, analog stimulus signal is generated by an arbitrary waveform generator (AWG) which employs a D/A converter inside, and analog signal is measured by a digitizer or a sampler which employs an A/D converter inside. The stimulus signal is created with 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 in this series, please visit the Verigy web site at www.verigy.com/go/gosemi.
There is a glossary of abbreviations used in this paper at the end of the content.

1. Introduction

In the previous go/semi newsletter, we discussed discrete Fourier transform, or DFT. In this issue we discuss FFT, which is the turbo version of DFT. It is the most powerful and useful tool in test engineering.

2. Fast Fourier Transform (FFT)

The DFT procedure is quite simple as seen in the previous issue; however it contains nested loop calculations so that data processing time increases exponentially according to the number of data. By restricting the number of data to 2n, the "fast Fourier transform" or FFT is developed, that is the very fast and effective algorithm of DFT calculation. In the Verigy V93000 test system, there are several built-in APIs available relating to FFT as follows.

```
DSP_SPECTRUM(dWave,dSp,DB,1.0,RECT,0);
DSP_FFT(dWave,CSp,RECT);
DSP_DFT(dWave,CSp,Ncycles);
DSP_THD(dWave,&THDresult,Ncycles,Nharmonics,DB,0);
DSP_SND(dWave,&dSND,&dSP,&dNP,Ncycles,1.0,0);
DSP_IFFT(CSp,CWave);
etc.
```

List 1: FFT Related APIs

DSP_SPECTRUM() is the most popular API of all. It is very useful to analyze the fundamental component of the test signal, its harmonics and spurious levels numerically. It is often used to show the spectrum on a graph as well.

DSP_THD() and DSP_SND() are used to directly get the numerical data of THD and/or SND (SINAD). FFT is executed inside. You should specify the bin location of the fundamental component, and the API reports the specified parameter value. Even if you do not need a graphical data in your program, it is a good practice to see the spectrum of the captured signal and ensure the validity of what you capture with digitizers/samplers or DUT ADC.

All the APIs except DSP_DFT() require 2n points of data as an input data array. DSP_DFT() can take care of any number of data, but it is not intended to figure out entire spectral data in order not to take huge processing time. It delivers the fundamental, the second and third harmonics components only, however it is good enough to be applied to regular signal analyses such as a signal level, especially when for some reason you cannot set up the test condition for capturing 2n points of data. If you would need to get entire components on the data of non-2n-points of data, you can create your own subroutine for full DFT as discussed in the previous article. [1]

DSP_FFT() is utilized to analyze the phase information of the signal component. For instance, if you need a group delay data in a filter testing, the phase data is necessary. On the other hand, DSP_FFT() is often utilized with the combination with DSP_IFFT() that is inverse fast Fourier transform (IFFT) in filtering by convolution, waveform reconstruction, etc. Further detail about IFFT and related topics will be discussed in separate articles. Also windowing is another intriguing topic in DFT/FFT. It will be discussed in a separate article.

---

[1] DSP_RF_FFTW() is another FFT routine. It is provided for the RF system. It can take care of non-2^n-points of data. For further details, consult the V93000 manual.

3. Spectrum Appearance

For an easier understanding of spectrum analysis, some of the key numbers are defined as follows.

```
Fs      sampling/digitizing frequency
N       total number of sampled data
Ft      test signal frequency
M       number of cycles of the sine wave in the UTP
UTP     unit test period which is the observation time
Fresln frequency resolution or bin spacing
```

List 2: Key Numbers

In the first article of this series in the go/semi newsletter, the Nyquist theorem was described as follows.

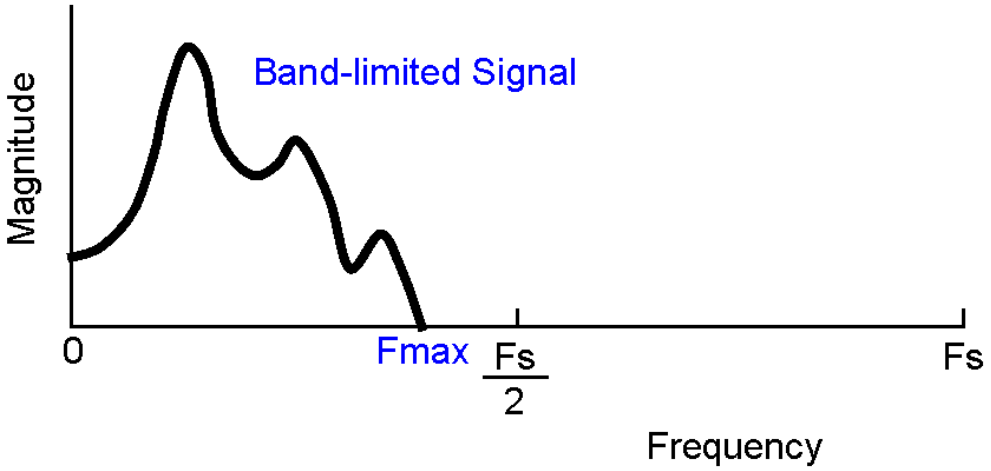

Figure 1: Nyquist Theorem

The test signal frequency (Ft) must be limited within the frequency range of Fs/2. When Ft>Fs/2, the test signal spectrum would be aliased or fallen in the baseband (Fs/2). You lose the true frequency information in the under-sampling condition. The condition of Ft>Fs/2 is called "under-sampling," which is an important and interesting methodology in mixed signal testers. It will be discussed in a separate article.

Figure 2 shows the relationship of the time domain data and its frequency domain data. The UTP includes a whole number (M) of cycles of sinusoidal waveform. The number of total sampling points is N. Then the frequency domain spectrum appears as shown in the lower picture. The number of total spectral bins is N/2. The index of the bins starts from 0 that is DC. The bin step or the frequency resolution is the reciprocal of the UTP. The index of the final bin is N/2-1. The signal spectrum is located at the bin number M. You should be very familiar with these relationships – Ft, Fs, M, N, UTP and Fresln in DSP-based testing.

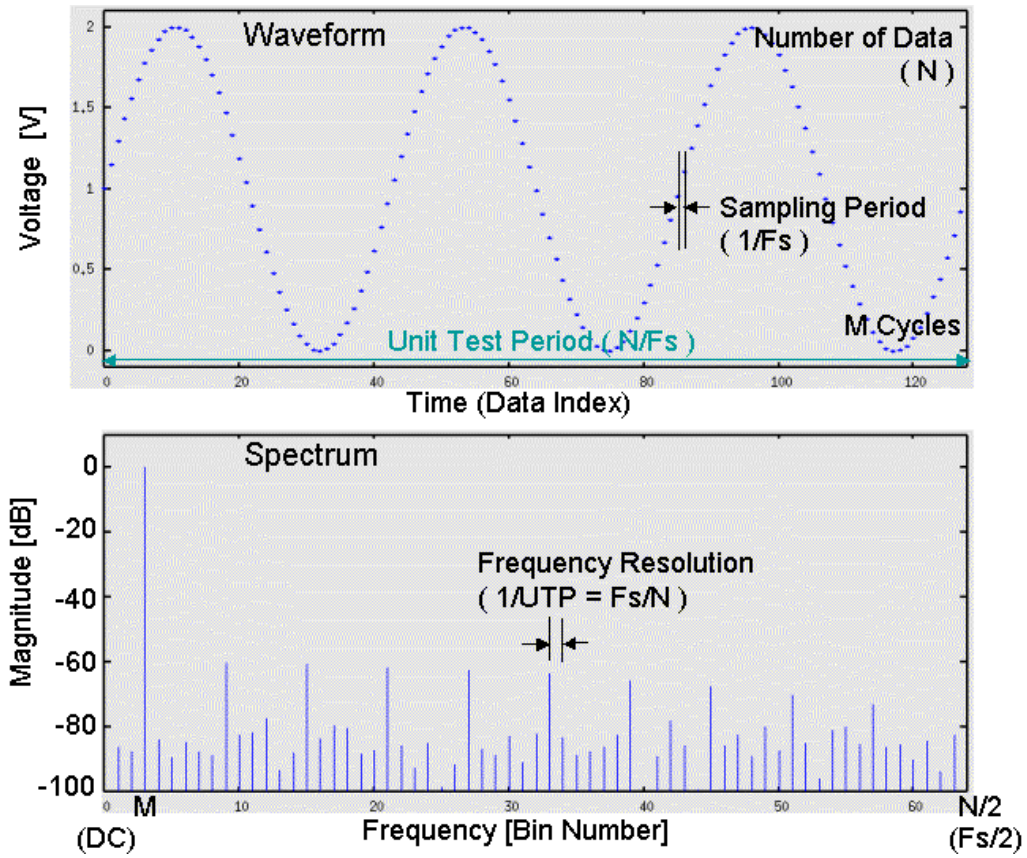

Figure 2: Sampled Waveform and DFT/FFT Result (Spectrum)

In the time domain picture in Figure 2, a whole number or an integer number M cycles of waveform is exactly captured in the N points of data. It can be expressed as Equation (1).

$$\frac{Ft}{Fs} = \frac{M}{N} \qquad (1)$$

It is called "coherent condition." This condition is absolutely important in the DSP-based testing. M and N are whole numbers. They have no common divisors, or are mutually prime.

Further detail will be discussed in separate articles – coherent condition, windowing, under-sampling, etc.

4. Terminologies in Spectrum Analysis

Let's look at terminologies relating to the spectrum display in Figure 3. The outstanding spectrum is usually the fundamental component of the test signal. If M cycles of the sine wave are captured in the UTP that is N/Fs, the signal should be located at the bin number M. The rest of the spectral lines are basically noise. You may notice some remarkable spectral lines in the noise. If the DUT is imperfect in linearity, you will find harmonics distortion at the locations of integer multiples of M. For instance, at bin number 2M, you will find the second harmonic, and at bin number 3M, the third harmonic, and so forth. If there is some noticeable noise not related to the harmonics location, they are called spurious spectra or spurs.

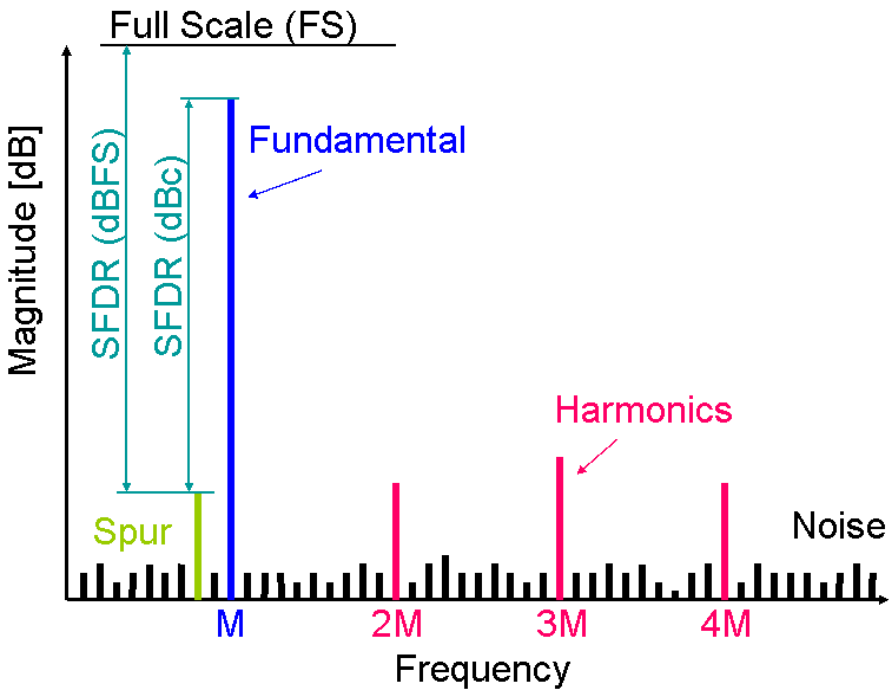

Figure 3: Terminologies in the Spectrum

SFDR or spurious free dynamic range is one of the key parameters in the spectrum analysis. SFDR is defined as the magnitude distance from the fundamental spectrum level to the maximum spur level. The maximum spur may or may not include the harmonics distortion. It depends on the device specification. If the DUT is an ADC, SFDR may refer to the full-scale level instead of the fundamental component level. The unit can show the definition such as [dBFS] or [dBc].

The spectrum display usually shows [dB] magnitude of spectral lines. When calculating ratio-metric parameters, it is a power ratio of signal components. The primitive spectral lines should be figured out in the dimension of [volt], meaning VOLT should be set in DSP_SPECTRUM(). When the length or magnitude of each spectral line is described as V(bin#), the signal power, harmonics power and noise power can be calculated as follows.

$$SignalPower = V(M)^2$$

$$HaronicsPower = V(2M)^2 + V(3M)^2 + V(4M)^2 + ... + V(kM)^2 + ...$$

$$NoisePower = \sum_{i=1}^{\frac{N}{2}-1} V(i)^2 - SignalPower$$

List 3: Powers

The parameter PWR is available in DSP_SPECTRUM() as well, and it may be useful to calculate the squared power.

There are some characteristic parameters defined in terms of the signal integrity shown as Table 1.

Table 1: Characteristic Parameters

SINAD (Signal to Noise and Distortion Ratio) $\text{SINAD [dB]} = 10 \log \frac{\text{SignalPower}}{\text{NoisePower}}$	Caution: SINAD may be referred as SND. SINAD is often referred as SNR. Watch out the definition of "SNR"
SNR (Signal to Noise Ratio) $\text{SNR [dB]} = 10 \log \frac{\text{SignalPower}}{\text{NoisePower-HarmonicsPower}}$	Caution: In audio SNR may be defined as below. $\text{SNR [dB]} = 20 \log \frac{\text{RMS Voltage @ Signal}}{\text{RMS Voltage @ No Signal}}$
THD (Total Harmonics Distortion Ratio) $\text{THD [dB]} = 10 \log \frac{\text{HarmonicsPower}}{\text{SignalPower}}$	Caution: THD [%] is also used.
SFDR (Spurious-Free Dynamic Range) $\text{SFDR [dBc]} = 10 \log \frac{\text{SignalPower}}{\text{WorstSpurPower}}$	Caution: SFDR [dBFS] refers the full-scale level instead of signal level. Harmonics may be excluded from spurious.

SINAD (SND) or signal to noise and distortion ratio is one of the most important parameters in testing converters. It is a ratio of the signal power and the total noise power including everything except the fundamental signal. SINAD is always shown in [dB]. The second important parameter is THD or total harmonic distortion, which is a ratio of the total harmonic power in the band and the signal power. THD is shown in [dB], and may be shown in [%] in especially audio field. The third parameter is SNR or signal to noise distortion, which is a ratio of the signal power and the total noise power without harmonic distortion. Caution! The terminology "SNR" is often used as equivalent of SINAD. Usually when both SINAD and SNR are specified, the SNR does not include harmonics. If SINAD is not specified, probably SNR is equivalent to SINAD. The last parameter SFDR is already discussed in Figure 3.

5. Cautions in SNR Calculation

(1) Definition of SNR

When calculating SNR, you should pay attention to some points. Is the SNR equivalent to SINAD or not? You should check the definition.

(2) Bandwidth of Target

The next point is the bandwidth of calculating noise. Usually the whole bandwidth is accounted for in calculating SNR, meaning bin 1 through bin (N/2-1). This is common. However, the applicable bandwidth may be less than the Nyquist band. For instance, the sampling rate of audio DVD is 48kHz so that the Nyquist bandwidth is 24kHz. If a DAC for audio DVD is tested by using a digitizer with a sampling rate of 768kHz, the noise is distributed from DC to 384kHz. However, the interested bandwidth is 24kHz. Then you may want to calculate the noise power up to 24kHz instead of 384kHz. Probably you will gain more dB of SNR in this case. This may be called "over-sampling."

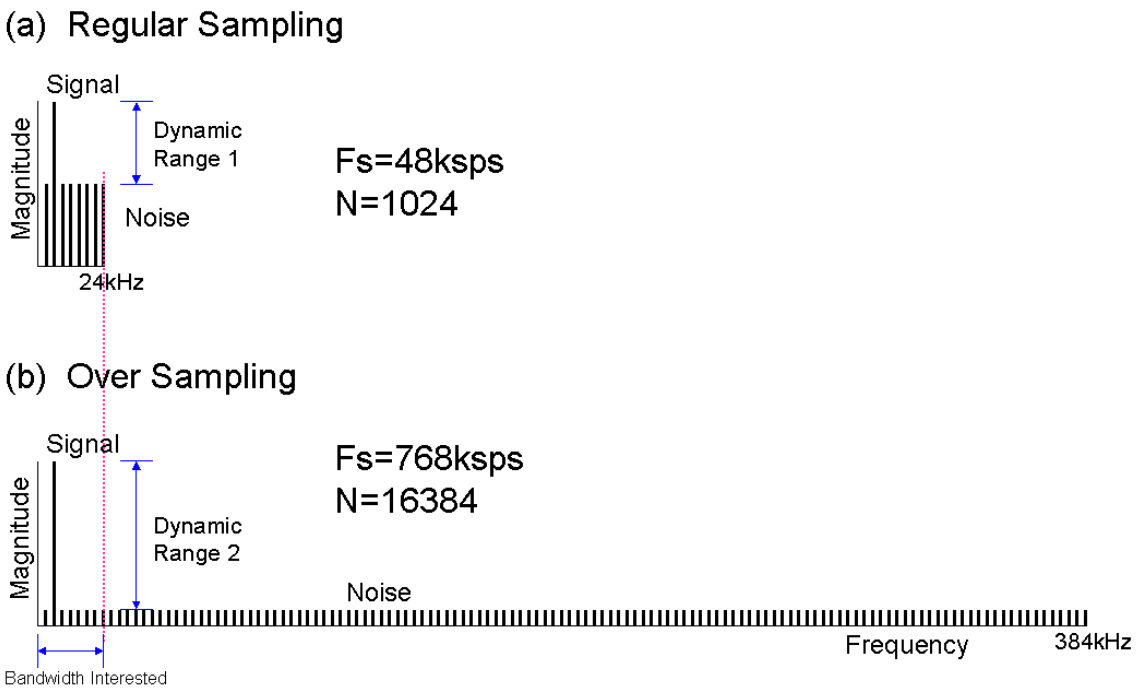

Figure 4: Over-Sampling

Figure 4 depicts the effect of over-sampling. The SNR based on the quantization noise is constant and known as SNR[dB]=6.02*n+1.76. The SNR based on the quantization noise is equal between the case (a) regular sampling of 48ksps and the case (b) over-sampling of 768ksps. The more number of points are captured, the lower the noise floor becomes. In the over-sampling condition, you can get more dynamic range so that you can see much smaller signals. When calculating the SNR performance of the device under test, if your bandwidth interest is 24kHz, you may collect the noise power less than 24kHz so that you could get much better result than regular sampling. This is the advantage of the over-sampling; however you will pay some cost for the big number of data with longer data uploading and processing time.

(3) Weighting Filter

This is a conventional practice. Especially in audio application, a specific weighting filter such as A-weighting, C-message, or psophometric may be applied to the SNR calculation. A-weighting is often used in regular audio devices. C-message and psophometric are typical in telecommunication audio devices such as telephone PCM CODEC. It would gain approximately 2 to 3 dB in the SNR in general so that if you forgot to apply it, you would lose 2 to 3 dB in your test result. The typical weighting factor can be provided by the DSP API as follows;

```
DSP_ASSGIGN_FILTER(WeightingArray, SamplingRate, Type);
```

where Type is one of C_MESSAGE, PSOPHO or A_WEIGHTING. The weighting filter shape is shown in Figure 5.

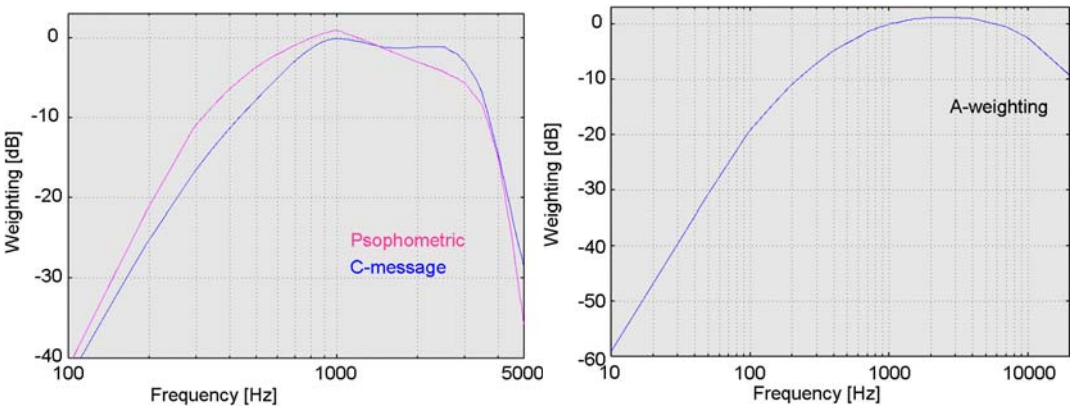

Figure 5: Typical Weighting Filters

You can apply the weighting array in a parameter in the FFT API's as follows.

```
DSP_SPECTRUM(dWave,dSp,DB,1.0,RECT,WeightingArray);
```

(4) Averaging

Averaging is a nice practice to get a stable and reliable result in general. SNR is a noise measurement. Noise is a random phenomenon. If you apply any averaging in the noise measurement, the more averaging you apply, the better SNR you would achieve. So averaging is not appropriate in random phenomena. Do not apply averaging in noise or SNR measurement. By the way, you may apply averaging in THD or SFDR measurement to achieve an accurate result, because distortion or spurious are not a random phenomenon. A THD value may be fluctuated by noise. So averaging can make it a repeatable result.

6. DSP_SPECTRUM()

This is the most popular and useful DSP API in the mixed signal testing with V93000. The third and fourth parameters in the API should be utilized correctly. If you need a plane linear spectrum, apply "VOLT" as the third parameter and assign just a dummy number to the fourth perfunctory. Then spectral lines shows each voltage of the component. (See Figure 6.)

DSP_SPECTRUM(dWave, dSpct, VOLT, 0.0, RECT, 0);
(Dummy)

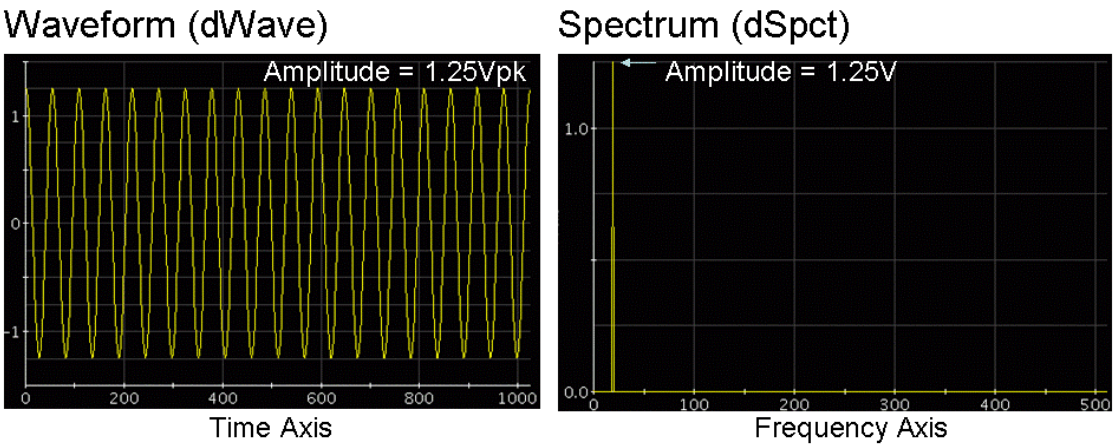

Figure 6: Plane Linear Spectrum with Parameter "VOLT"

When you need a magnitude spectrum in [dB], assign DB to the third parameter, then you will get the spectrum as Figure 7. In this case, if you assign 0.0 to the fourth parameter, the API automatically finds out the maximum spectrum (except DC) in the components, and set the maximum component as 0dB reference magnitude.

DSP_SPECTRUM(dWave, dSpct, DB, 0.0, RECT, 0);

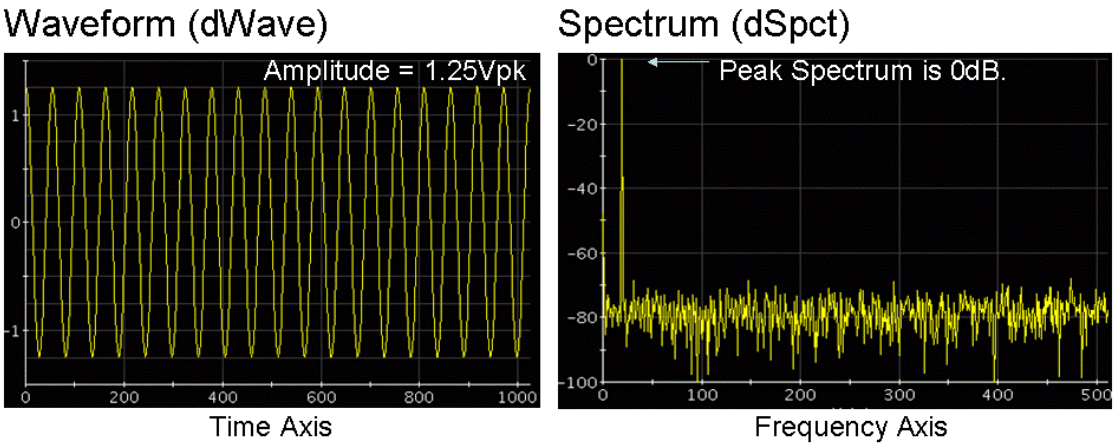

Figure 7: dB Expression of the Spectrum

When the device under test has a specific 0dB reference level, you may indicate the reference value in the API. Then the API takes the specified value as the reference of 0dB, and figures out entire spectra with respect to the reference. Figure 8 shows an example, and the specified value 0.3162V is the 0dBm voltage across 50Ω load. Consequently each spectral line indicates [dBm] values in the spectrum display.

dVref=0.3162 V
DSP_SPECTRUM(dWave, dSpct, DB, dVref, RECT, 0);

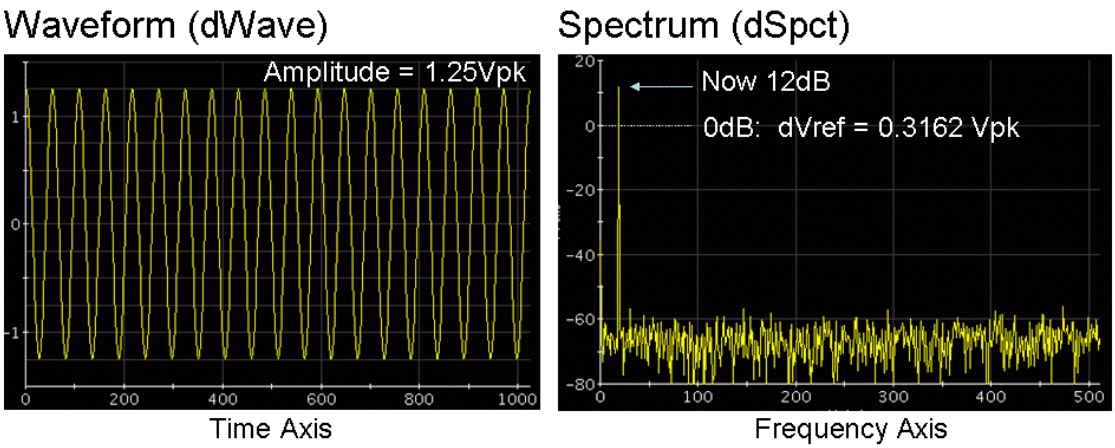

Figure 8: The Fourth Parameter as the Reference Level

Figure 9 shows an example of an 8-bit linear ADC test. When calculating the spectrum, the theoretical full-scale size of the 8-bit is 255/2=127.5 so that this value is specified as the 4th parameter. Then the spectrum display shows the magnitude referred to the full-scale size.

eg. 8-bit A/D Converter (Code 0…255)

dVref=127.5
DSP_SPECTRUM(iADC, dSpct, DB, dVref, RECT, 0);

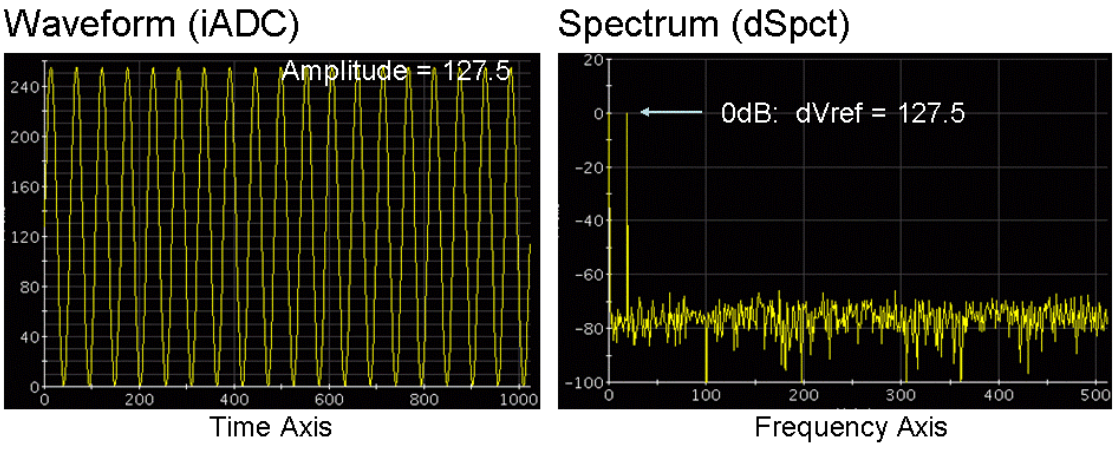

Figure 9: [dB] Magnitude for 8-bit ADC

The waveform data in Figure 9 is the fullscale data so that in this case the fundamental spectrum meets the level of 0dB in the frequency domain.

7. Glossary of Abbreviations

DFT – Discrete Fourier Transform
FFT – Fast Fourier Transform
IFFT – Inverse Fast Fourier Transform

ADC – Analog to Digital Converter
API – Application Programming Interface
DSP – Digital Signal Processor or Processing
DUT – Device Under Test
SFDR – Spurious Free Dynamic Range
SND – Signal to Noise and Distortion Ratio
THD – Total Harmonic Distortion
UTP – Unit Test Period