Hideo Okawara's
Mixed Signal Lecture Series

DSP-Based Testing – Fundamentals 4
Multi-tone Data Generation

Verigy Japan
August 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 and the measured signal is processed with a 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 of DSP-based testing. FFT (Fast Fourier Transform) is the most powerful tool here. This series 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.

1. Introduction

Multi-tone is a popular stimulus signal in mixed signal tests. By using a multi-tone you can capture gain and phase information at multiple frequency points with a single measurement so that it is very effective in frequency response analyses. As discussed in the last newsletter, when a multi-tone is programmed and generated by a DAC, it has a gain shaping of SINC function. In this issue, let's look at the waveform and think of the point of appropriate multi-tone signal generation.

2. Primitive Method

A multi-tone signal contains multiple frequency waveforms combined in a single waveform. Therefore the very basic method to create a multi-tone signal is a summation of single tone waveforms. The principle is very simple. If you create a multi-tone for the first time, you may write your code as follows.

```
INT       I,j,Ndata,Ntones,Ncycles;
DOUBLE   dP;
ARRAY_D  dWave;

Ntones=100;             // # of tones
Ndata=1024;             // # of data points
dWave.resize(Ndata);
dWave.init(0.0);       // Initialization (DC offset)
for (j=0;j<Ntones;j++) {
      Ncycles=j+1;
      dP=2.0*M_PI*Ncycles/Ndata;
      for (i=0;i<Ndata;i++)
            dWave[i]=dWave[i]+sin(dP*i);
}

DSP_ABS_MINMAX(dWave,&dMin,&dMax,&iMin,&iMax);
DSP_MUL_SCL((0.5/dMax),dWave,dWave); // Scaling to 0.5V
```

List 1. Multi-tone Coding – Primitive Method

This is an example coding of 100 tones combined in the 1024 points of data. When the data generated by this code list is stored in a DAC or an AWG, the generated waveform and spectrum look as Figure 1. The time domain waveform looks something like an impulse signal. Its peak-to-rms ratio (PRR) is 20dB. However, the frequency domain spectrum shows definitely 100 tones. The magnitude is flat. The level is -37.1dB each to the full-scale of the device (DAC or AWG). (SINC effect is not considered here.) This is certainly a multi-tone signal; however the waveform is so extreme that it cannot be appropriate for regular tests.

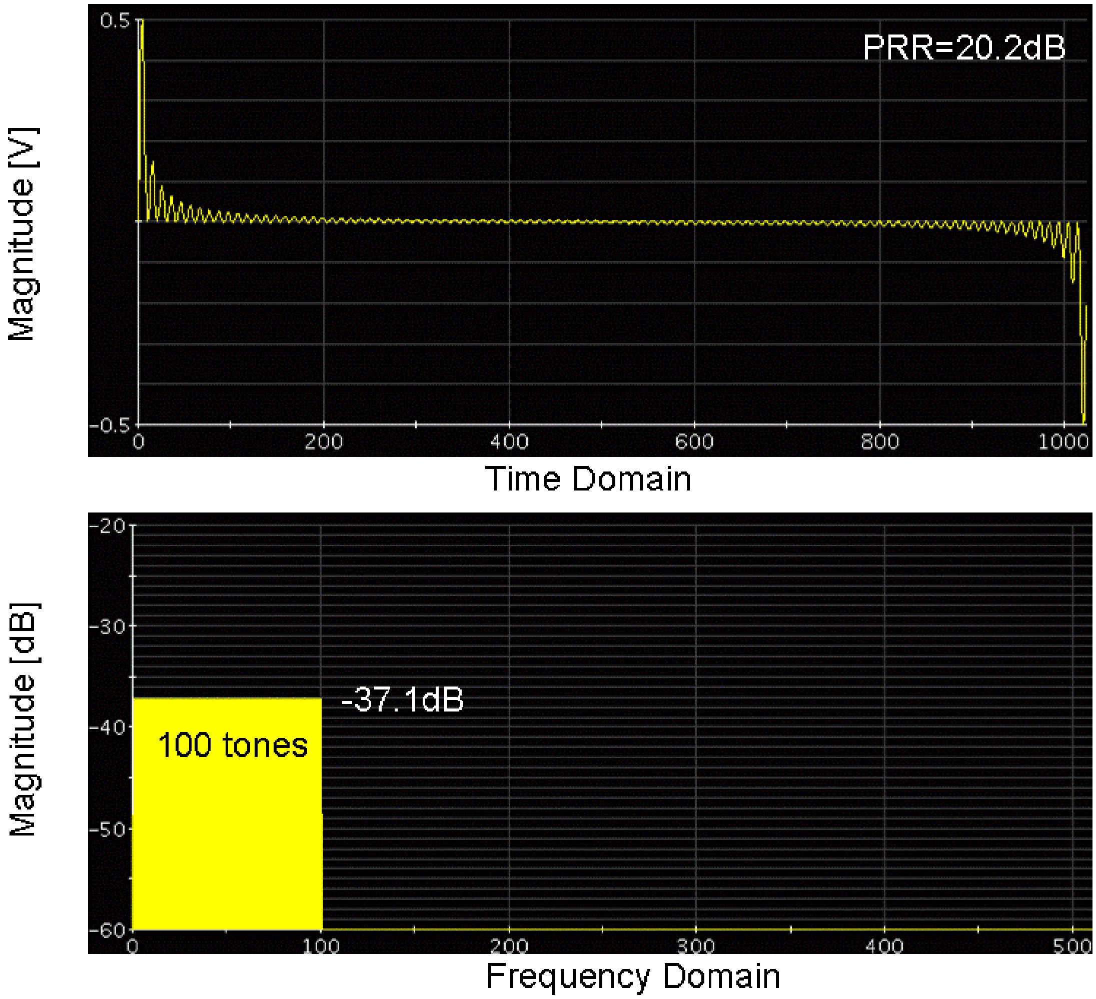

Figure 1. Multi-tone Waveform – Primitive Method

3. Good Practice – Random Phase

The cause of the problem is located in the phase control. In List 1, all sine waves start with the phase “zero” so that the total signal becomes extreme shape.

The point is appropriate seasoning of phase offset in each frequency component. A best practice is to insert random phase offset, employing the random number function in C programming. See the code list below.

```
Ntones=100;
Ndata=1024;
dWave.resize(Ndata);
dWave.init()=0.0;
```

```
for (j=0;j<Ntones;j++) {
      Ncycles=j+1;
      dP=2.0*M_PI*Ncycles/Ndata;
      dQ=2.0*M_PI*drand48(); // random phase
      for (i=0;i<Ndata;i++)
            dWave[i]=dWave[i]+sin(dP*i+dQ);
}

DSP_ABS_MINMAX(dWave,&dMin,&dMax,&iMin,&iMax);
DSP_MUL_SCL((0.5/dMax),dWave,dWave);
```

List 2. Multi-tone Coding – Random Phase

By applying this code, you can create a waveform as in Figure 2. The waveform looks like noise, and it seems to activate the full range of the code. The spectrum is enhanced by 9dB over the case of no phase offset. This signal is a very useful stimulus not only for the frequency response tests but also for a wide-band signal. Modern telecommunications often utilize wide-band signals so that phase-randomized multi-tone is a good substitute of the true modulated signals. PRR might be specified in some cases. Using random phase offset, PRR would be roughly 10 to 15dB.

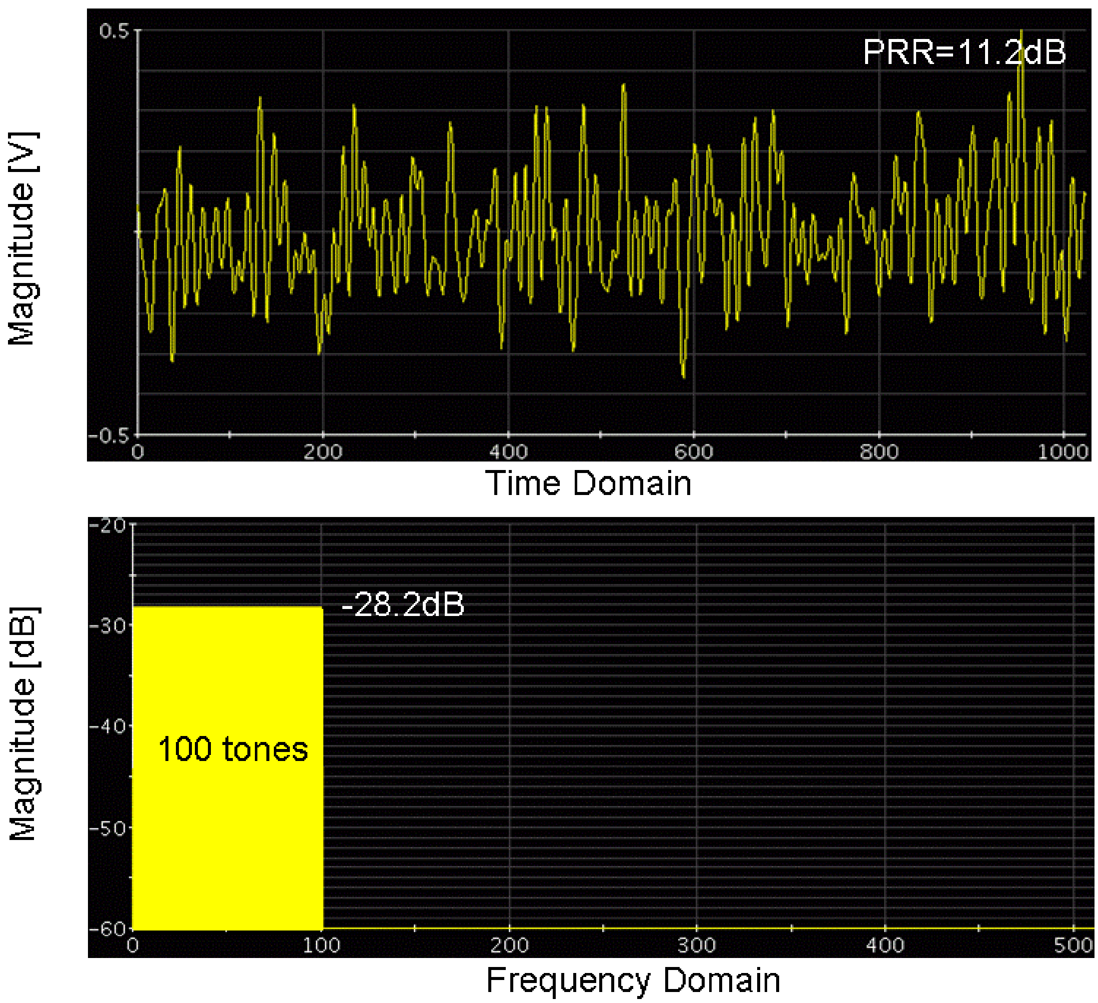

Figure 2. Multi-tone Waveform – Random Phase

A DAC or an AWG has a limited full scale and resolution. When a multi-tone is programmed in such a device, the peak of the signal would be aligned to the full-scale range. The greater the number of tones you make, the lower level or power each tone can convey, and then the less dynamic range the signal can perform. So compromise is needed.

4. Good Practice – Parabolic Phase

The waveform discussed in the previous section looks like a noise. You can use this type of signal for frequency response tests. One of the most typical tests is a filter characterization test. The flatness of the pass band and the loss of the rejection band are popular checkpoints.

As you see in Figure 2, the magnitude of the tones is –28dB in the example. The full-scale range limits the total signal power so that the power is distributed in each one of the spectra equally. If a measurement instrument has the resolution of 16-bit, its AC dynamic range is +/-32768, meaning the minimum signal level is 20*log(1/32768)=-90dBFS. When you apply the multi-tone in Figure 2, you have already lost 28dB of the range. Therefore each one of the tones has the dynamic range of 90-28=62dB, which may not be good enough for your filter characterization.

When applying a multi-tone to filter tests, you may want to distribute as much energy as possible to each tone to achieve the maximum dynamic range within the limited full-scale. In this case the parabolic phase control works effectively. Look at List 3 below.

```
Ntones=100;
Ndata=1024;
dWave.resize(Ndata);
dWave.init()=0.0;
for (j=0;j<Ntones;j++) {
      Ncycles=j+1;
      dP=2.0*M_PI*Ncycles/Ndata;
      dQ=M_PI/Ntones*(1-Ncycles*Ncycles);
      for (i=0;i<Ndata;i++)
            dWave[i]=dWave[i]+sin(dP*i+dQ);
}

DSP_ABS_MINMAX(dWave,&dMin,&dMax,&iMin,&iMax);
DSP_MUL_SCL((0.5/dMax),dWave,dWave);
```

List 3. Multi-tone Coding – Parabolic Phase

The phase offset of each tone is controlled mathematically in this case. Creating the multi-tone using this method, the waveform and the spectrum look as Figure 3. The waveform looks as a swept signal so that the signal may not be appropriate as a substitute test signal for telecommunications tests. However, each of the tones gains 5.6dB more than the case of random phase so that it can test filters that have up to 67.6dB of rejection. It is good for the frequency response test of wide-band devices.

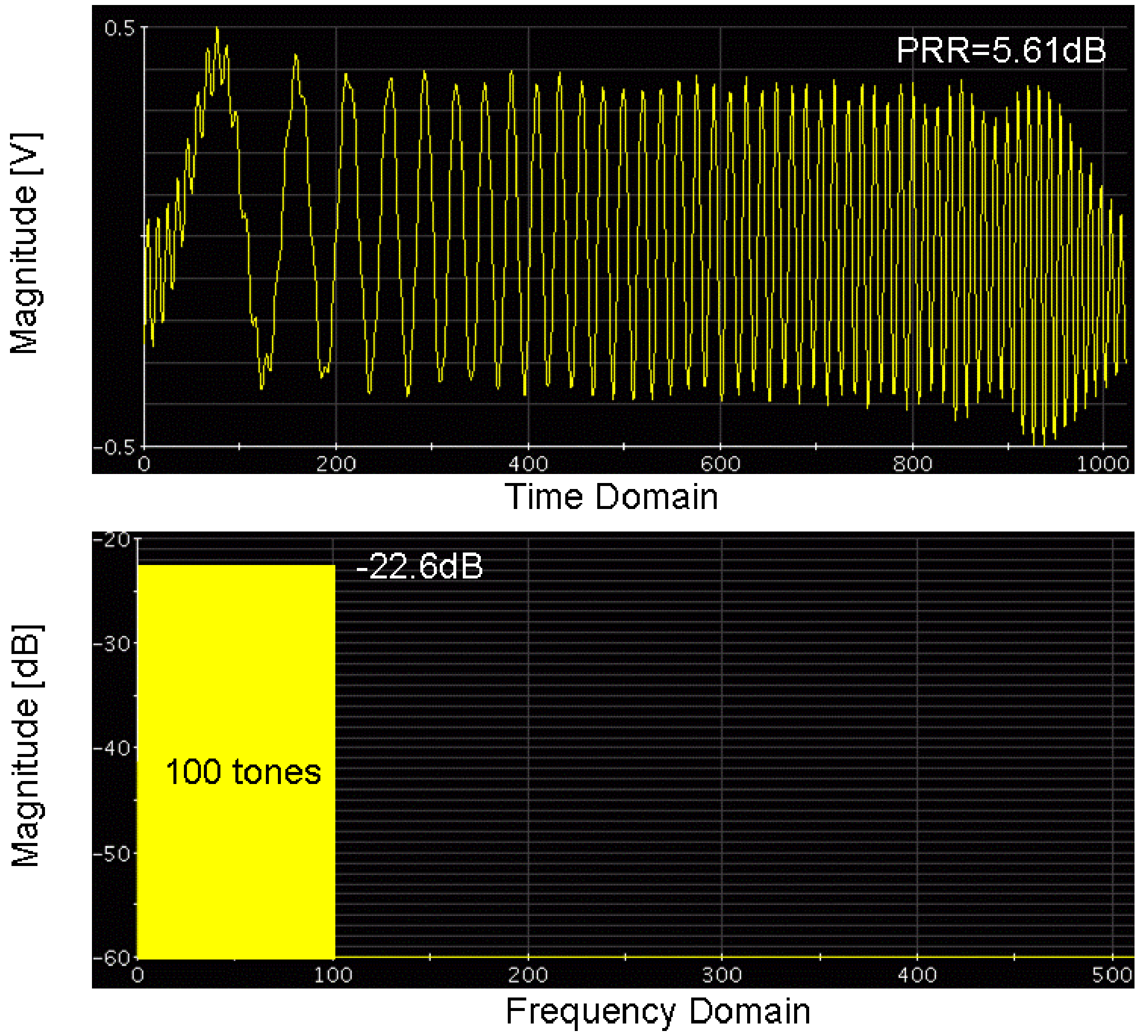

Figure 3. Multi-tone Waveform (3)

Reference

[1] Greg Lowitz, Robert Armitaro, "Predistortion improves digital synthesizer accuracy," Electric Design Mar. 31, 1988, pp. 85-89