Hideo Okawara's
Mixed Signal Lecture Series

DSP-Based Testing – Fundamentals 29
Precision Waveform Shift

Verigy Japan
September 2010

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.

Precise Waveform Shift

The target signal of a waveform measurement should always be located at the same location for computer automated waveform parameter analysis. Shifting measured waveform along the time axis was previously discussed in the topic of centering.[1] In this case the waveform data array was simply rotated left or right to be placed in the target location. It was a straightforward rotation of data array so that the shifting resolution is the discrete step of the digitizer's sampling period or sampler's equivalent sampling period.

Modulation/demodulation applications are often performed by using I/Q separated signals, where two AWG and/or two digitizer channels must be well calibrated. However, if the channel calibration would not be good enough or the calibration itself would not be possible to be performed for some reason, you should find any workaround solution for compensating the miss-match between channels. In this case waveform shifting must be performed with very minute resolution.

The topic of the month is how to rotate-shift and align the waveform data to the target location with precision time. And this is a prerequisite to the next month's topic as well.

Delayed Sinusoids

The yellow colored wave in Figure 1(a) illustrates a 3-sinusoid combined waveform. The red colored wave is -1.59us shifted to the yellow wave. (b), (c) and (d) show their 3 component waves. (b) is the 1-cycle sinusoid component, (c) is the 2-cycle sinusoid and (d) is the 3-cycle sinusoid.

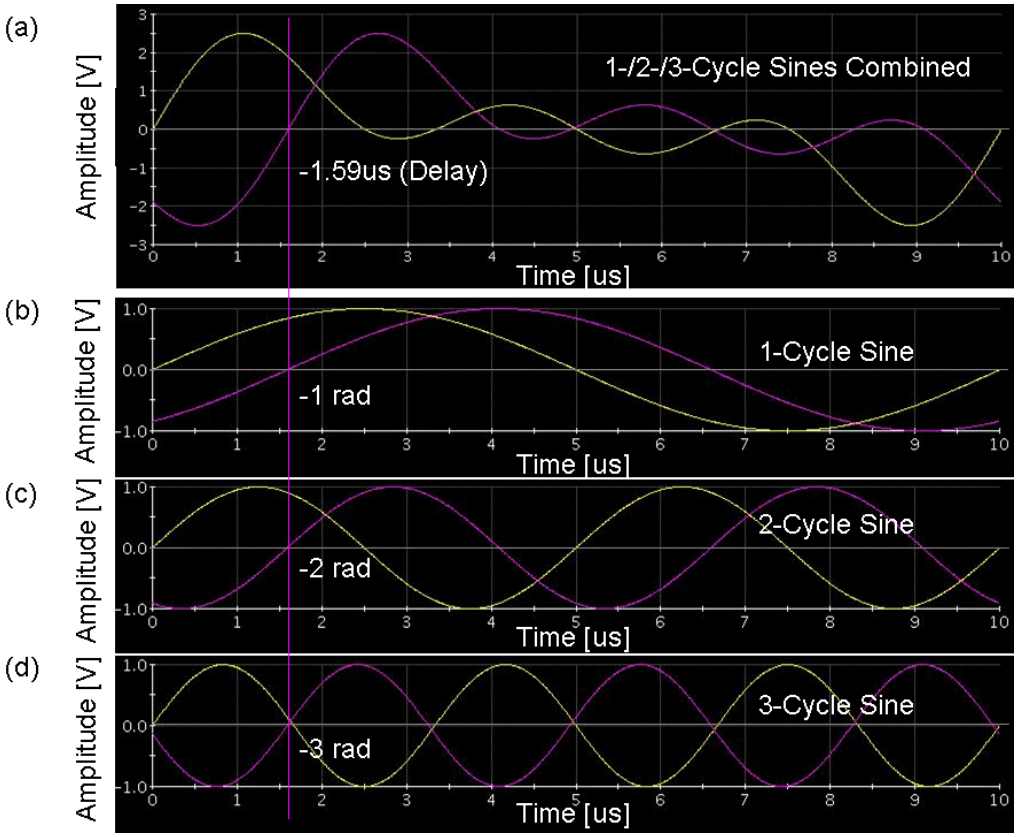

Figure 1: 3 Sinusoids Combined Wave and Its Delayed Wave

[1] Hideo Okawara's Mixed Signal Lecture Series – Fundamentals 23 – Centering

Each red sinusoid in (b), (c) and (d) is -1.59us delayed to the yellow sinusoid. However, examining the phase differences between the yellow and the red sinusoids, each difference is calculated as -1[rad] in (b), -2[rad] in (c) and -3[rad] in (d) respectively. The point of this fact is that the relationship between the time difference and the phase difference. The red waveforms are delayed by the same time -1.59us. However, the phase difference is not the same but proportional to its frequency.

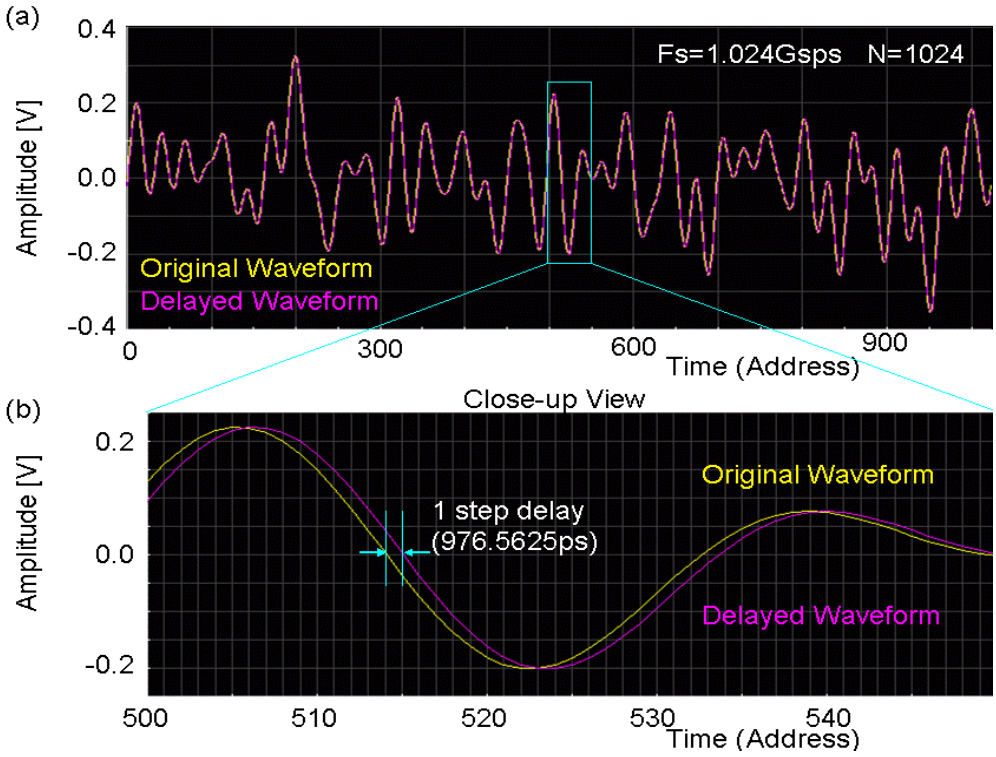

Figure 2 Multi-tone Waveforms

Let's confirm this fact with looking at a multi-tone. Figure 2 illustrates 40-tone waveform containing 1MHz, 2MHz, …, 40MHz, constructed with 1024 points at 1.024Gsps. Actually (a) shows two traces overlaid. The yellow wave is the original waveform and the red wave is delayed by 976.5625ps. (b) shows the close-up view of the highlighted area in (a) so that the difference can be recognized clearly.

Applying FFT to the waves in Figure 2, the phase difference of the delayed wave from the original wave is plotted in Figure 3. The x-axis shows the frequency bin numbers of each tone. It says that the phase difference at bin #1 is -0.006136[rad], and each tone phase of bin #2, 3, …, 40 differs 2x, 3x, …, 40x to the shift of bin #1. Consequently the frequency response of the phase rotation shows linear characteristics. Now the point is obvious. If you want to move the whole waveform forward (advance) or backward (delay), you should give a linear proportional phase rotation to each one of frequency components. When the shifted signal is delayed against the original, the phase difference becomes a negative slope as Figure 3. When the signal is advanced, the phase difference becomes a positive slope.

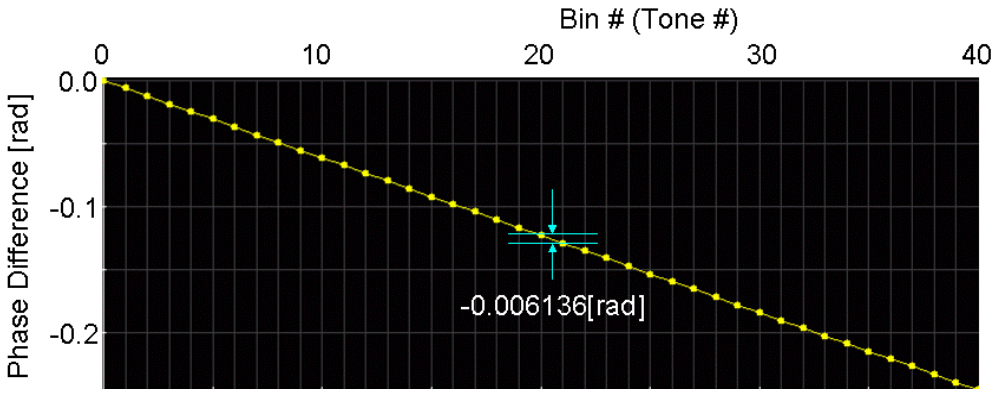

Figure 3 Phase Difference of each tone

FFT / IFFT Application

Now that you understand the point of shifting waveform left or right by arbitrary time period, let's look at a practical procedure to shift the waveform illustrated in Figure 4. It is a 1MHz clock waveform sampled by 1024 points at 1.024Gsps. The sampling period is 976.5625ps and the UTP is 1us. The clock is constructed by the components of the fundamental tone and its harmonics as illustrated in Figure 5.

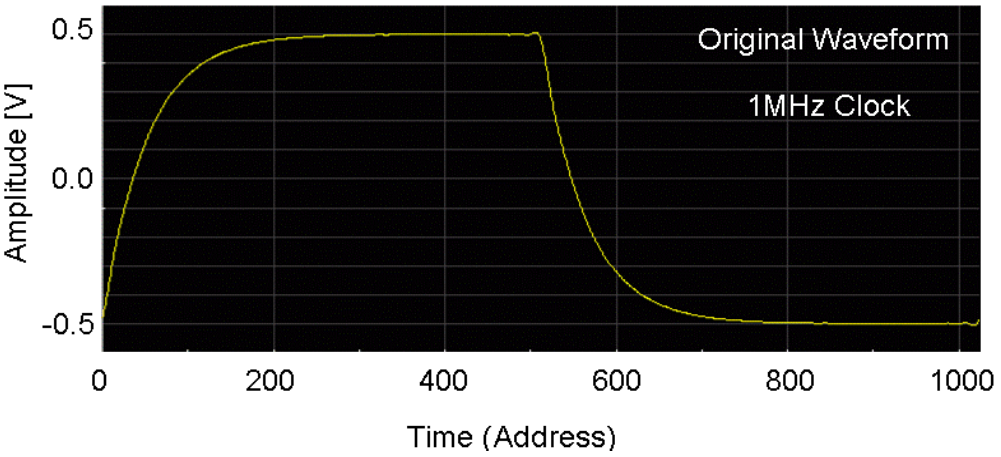

Figure 4 Example Waveform

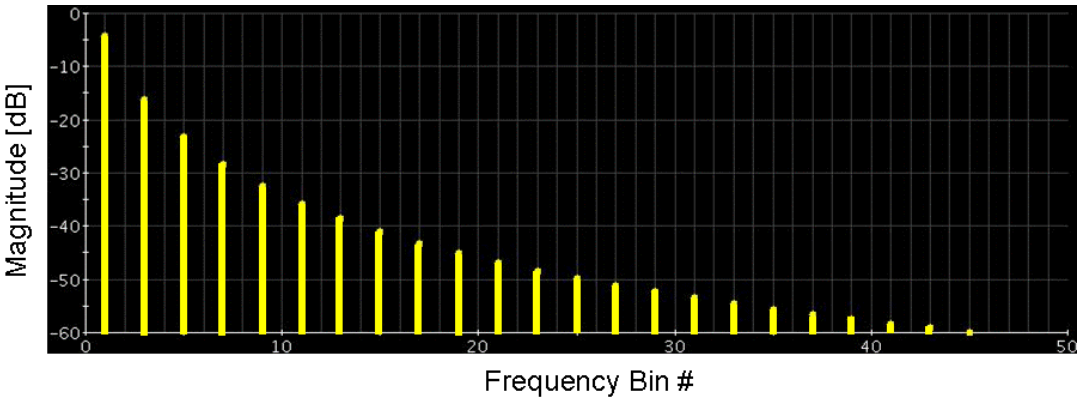

Figure 5 Spectrum of Clock

If you want to move the clock wave by 97.65625ns, you should give a phase rotation of (97.65625ns/1us) x 2π [rad] to the 1MHz component, and give linearly increasing phase rotation to its harmonics components sequentially. Several delays are demonstrated from 97.65625ns to 683.59375ns by every 97.65625ns step. Consequently the phase rotations to the entire frequency components are 7 linear lines as illustrated in Figure 6.

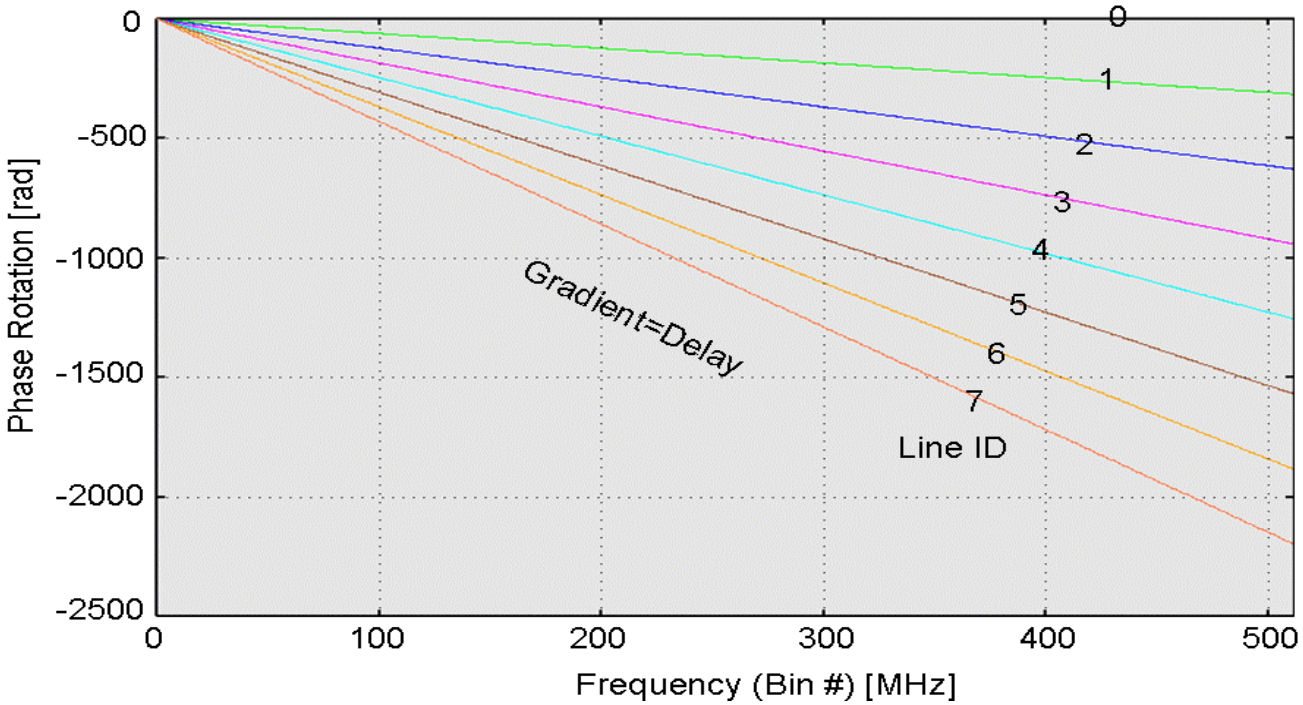

Figure 6 Linear Phase Characteristics (7 Different Delays)

List 1 illustrates the practical coding example for waveform shift. The original waveform is captured in the array "dWave[]" at Line 06, to which FFT is performed at Line 09. The delay time to shift is specified at Line 14. Arrays named "dGain[]" and "dPhase[]" are introduced. However, the gain is always set one at Line 12 because only phase is manipulated. The phase response is programmed linear proportional to the bin numbers at Line 16. The operation is "delay" so that the slope is set negative. If you want to move the target waveform forward, it should be set positive. The original signal is converted into the frequency spectrum "CSp" by FFT at line 09. The gain/phase characteristics are converted into complex numbers "CRot" finally at Line 17. Then the spectrum "CSp" is multiplied by "CRot" at Line 19. The vector multiplication actually rotates each one of the spectrum components as specified at Line 16. From Lines 20 through 25 are routine operations for performing IFFT which is already discussed in a previous newsletter article.[2]

Applying the seven different linear phase shifts illustrated in Figure 6 according to List 1, the original clock waveform is moved backwards (delayed) as overlaid in Figure 7. The red trace is the original waveform and the lines with Line ID 1 through 7 are the shifted results. Colors are consistent in Figures 6 and 7.

[2] Hideo Okawara's Mixed Signal Lecture Series – Fundamentals 13 – Inverse FFT (IFFT)

```
01:   INT            i,N;
02:   DOUBLE         dFs,dTs,dTdelay,dP;
03:   ARRAY_D        dWave,dGain,dPhase,dWaveD;
04:   ARRAY_COMPLEX CSp,CSpD,CRot,CWaveD;
05:   ...
06:   dWave=DGT("AOUT").getWaveform();        // Original Signal Waveform
07:   N=dWave.size();                         // For example N=1024
08:   dTs=DGT("AOUT").getConvClock();         // Sampling Period
09:   DSP_FFT(dWave,CSp,RECT);                // Original Signal Spectrum
10:
11:   dGain.resize(N/2);                      // Gain Container
12:   dGain.init(1.0);                        // Gain=1.0 (Always 1.0)
13:   dPhase.resize(N/2);                     // Phase Container
14:   dTdelay=97.65625 ns;                    // Required Delay Time(0..UTP)
15:   dP=2.0*M_PI*(dTdelay/dTs)/N;            // Phase Increment
16:   for (i=0;i<(N/2);i++) dPhase[i]=-dP*i; // Linear Phase (Delay=Negative):
17:   DSP_POL_RECT(dGain,dPhase,CRot);        // Polar to Rectangular (Vector Exp.)
18:
19:   DSP_MUL_VEC(CSp,CRot,CSpD);             // Phase Rotation Math. (Vector Mult.)
20:   CSpD.resize(N);                         // Delayed Signal Spectrum
21:   for (i=0;i<(N/2);i++) {
22:      CSpD[N-i].real()= CSpD[i].real();    // Complex Conjugate for IFFT
23:      CSpD[N-i].imag()=-CSpD[i].imag();
24:   }
25:   CSpD[N/2]=CZero();                      // Complex Zero
26:   DSP_IFFT(CSpD,CWaveD);                  // IFFT Reconstruction
27:   dWaveD=CWaveD.getReal();                // Imaginary should be garbage.
28:   DSP_MUL_SCL(0.5,dWaveD,dWaveD);         // Scaling Half
29:
```

List 1 Example Coding for Waveform Shift

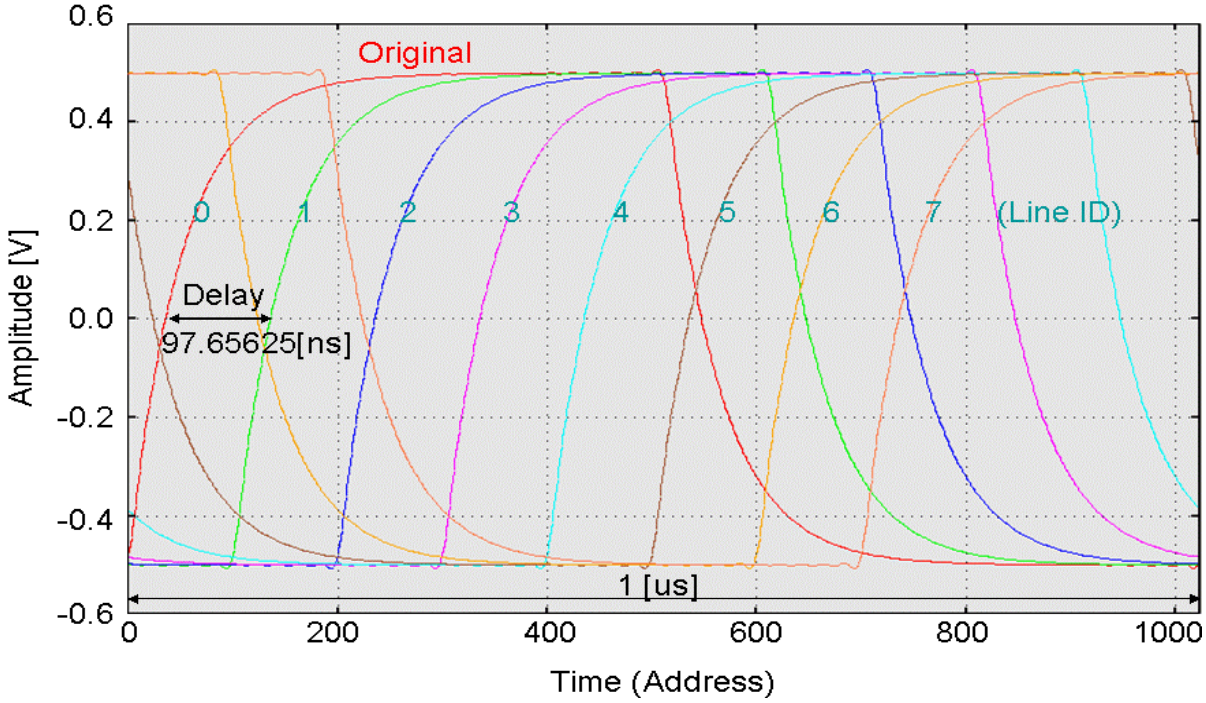

Figure 7 Delayed Results

In order to see how this method works to minute level, let's modify the delay time at Line 14 in List 1 with 7 fine values from 97.65625ps to 683.59375ps by every 97.65625ps step. The phase rotation response to the entire frequency components is illustrated as 7 linear lines in Figure 8. Then the clock waveform is precisely shifted as illustrated in Figure 9. The close-up view (b) shows each one of the traces is exactly delayed as programmed. It is much smaller than the sampling period that is 976.5625ns.

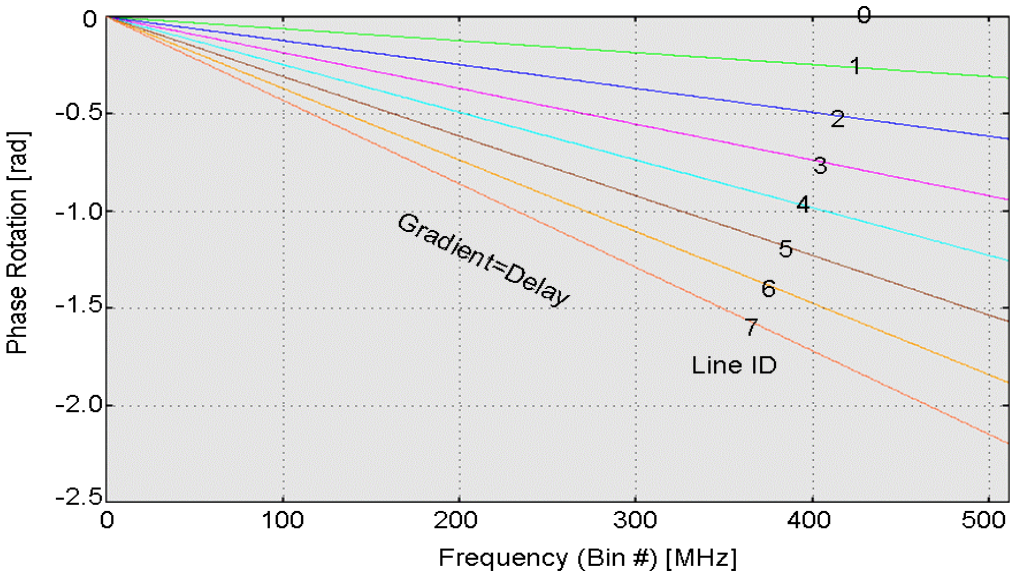

Figure 8 Linear Phase Characteristics

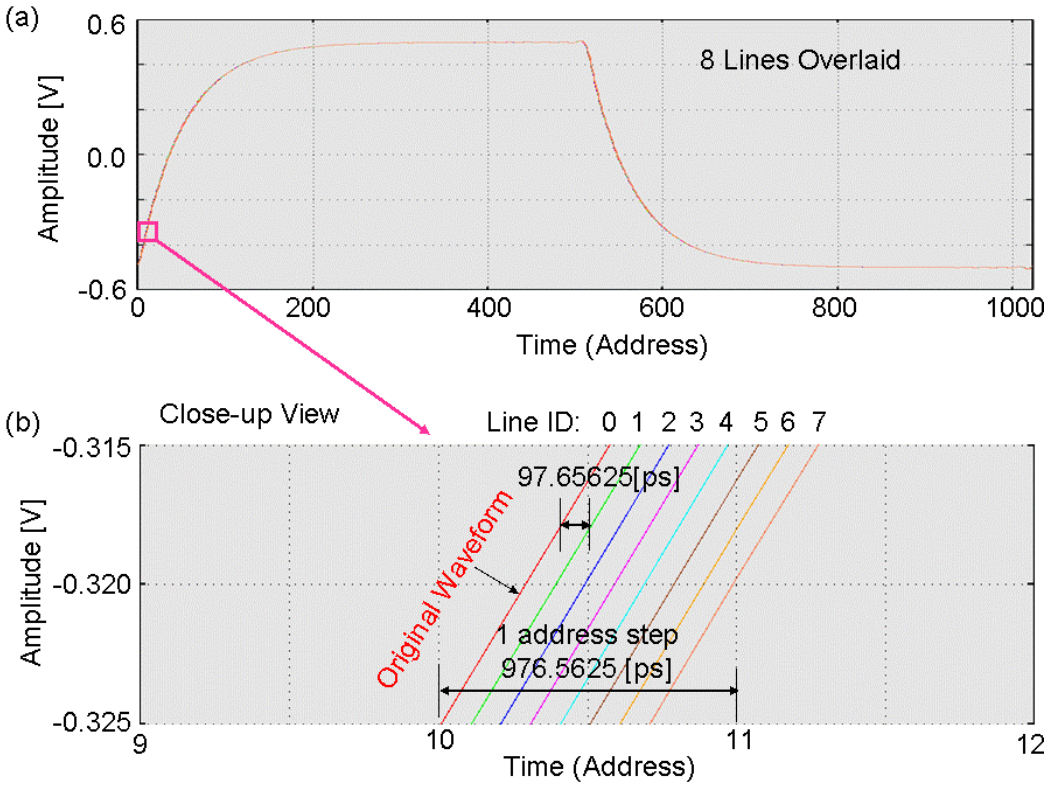

Figure 9 Delayed Results

Conclusion

- Precise waveform shift is performed by a phase rotation of frequency spectrum.
- Phase rotation of each spectrum component is proportional to its frequency.
  - The phase response to the frequency bin number becomes a linear line starting from 0.
  - Delay becomes a negative slope line. Advance becomes a positive slope line.
- Phase rotating operation is realized by complex number multiplication.