CHAPTER 8

Sampling Theory

Up to this point in the textbook, we have been discussing DC and other very low-frequency measurements of electronic devices. It is at this juncture that we will turn our attention to the testing of electronic devices with AC signals using digital signal processing (DSP) techniques. Such testing techniques have profoundly changed the design and use of automatic test equipment (ATE) found in most, if not all, production facilities. At the heart of these methods is the principle of sampling. While sampling has been known since the late 1920s in its present form, its widespread use in the design community came into being in the early 1970s with advancements in monolithic circuit integration techniques. Today, most electronic equipment, such as cell phones, personal handheld devices, computers, and so on, make use of these same design principles owing to its flexibility, programmability, testability, and cost.

In this chapter we describe the theory of sampling and a related concept of reconstruction. Much of the mathematical notation that we use for the remainder of this textbook will be defined here. We shall discuss the sampling theorem and the effects of aliasing. We shall also describe the impact of quantization effects and clock jitter (noise) on the overall conversion process. We will then turn our attention to the creation of repetitive analog signals to be used as the test stimulus and the need for a coherent sample set. Signal coherence is one of the most important system conditions for a fast and accurate DSP-based testing scheme. The chapter closes with a short discussion of clock synchronization across different sampling systems.

8.1 ANALOG MEASUREMENTS USING DSP

8.1.1 Traditional Versus DSP-Based Testing of AC Parameters

AC measurements such as gain and frequency response can be measured with relatively simple analog instrumentation, as mentioned in Section 2.4. To measure gain, an AC continuous sine wave generator can be programmed to source a single tone at a desired voltage level, V_{in}, and at

a desired frequency. A true RMS voltmeter can then measure the output response from the DUT, V_{out}. Then gain can be calculated using a simple formula: gain = V_{out}/V_{in}.

The pure analog approach to AC testing suffers from a few problems, however. First, it is relatively slow when AC parameters must be tested at multiple frequencies. For example, each frequency in a frequency response test must be measured separately, leading to a lengthy testing process. Second, traditional analog instrumentation is unable to measure distortion in the presence of the fundamental tone. Thus the fundamental tone must be removed with a notch filter, adding to test hardware complexity. Third, analog testing measures RMS noise along with RMS signal, making results unrepeatable unless we apply averaging or bandpass filtering.

In the early 1980s, a new approach to production testing of AC parameters was widely adopted in the ATE industry. The new approach became known as *DSP-based testing*.[1] Digital signal processing (DSP) is a powerful methodology that allows faster, more accurate, more repeatable measurements than traditional AC measurements using an RMS voltmeter. A mixed-signal test engineer will never be fully competent without a strong background in signal processing theory. Unfortunately, a full treatment of sampling theory and DSP is well beyond the scope of this book. Other texts have covered the subject of signal processing in much more detail.[2–4]

The reader is assumed to already have a strong theoretical background in DSP, although this book will undoubtedly fall into the hands of the DSP novice as well. We will review the basics of sampling theory and DSP as they apply to mixed-signal testing, without giving the subject an in-depth treatment. Hopefully, this introductory coverage will both refresh the experienced reader's memory of DSP and allow the novice to understand the fundamentals of DSP-based testing.

Before we can discuss DSP-based testing, we must first understand sampling theory for both analog-to-digital converters and digital-to-analog converters. In this chapter, we will examine the basics of sampling theory before proceeding to a more detailed study of DSP-based testing in Chapter 9.

8.2 SAMPLING AND RECONSTRUCTION

8.2.1 Use of Sampling and Reconstruction in Mixed-Signal Testing

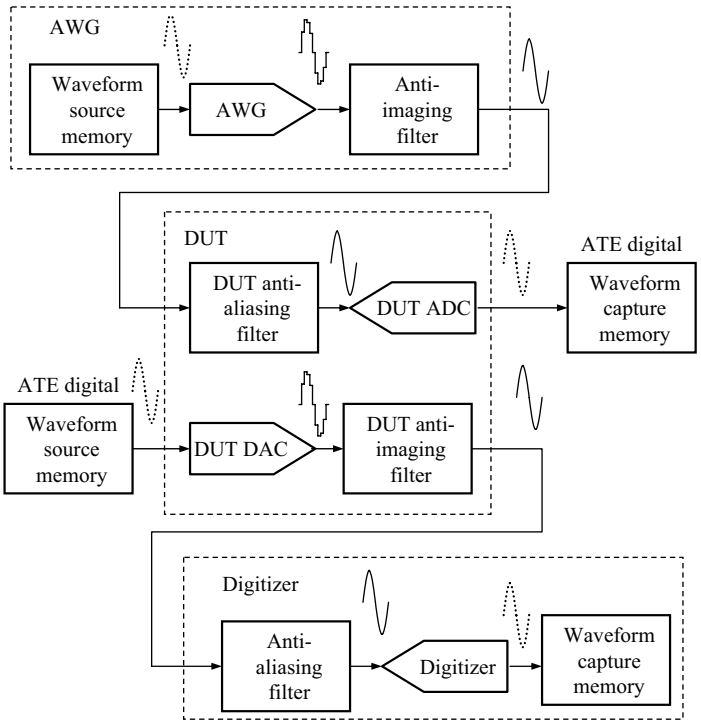
Sampling and reconstruction are the processes by which signals are converted from the continuous (i.e., analog) signal domain to the discrete (i.e., digital) signal domain and back again. Both sampling and reconstruction are used extensively in mixed-signal testing. The ATE tester samples and reconstructs signals to stimulate the DUT and measure its response. The DUT may also sample and reconstruct signals as part of its normal operation. Both mathematical and physical sampling and reconstruction occur as the DUT is tested. Figure 8.1 illustrates the various types of sampling and reconstruction that occur when the voice-band interface circuit of Figure 1.2 is tested.

In a purely mathematical world, a continuous waveform can be sampled and then reconstructed without loss of signal quality, as long as a few constraints are met. Unfortunately, a number of imperfections are introduced in the physical world that makes the conversion between continuous time and discrete time fall short of the mathematical theory. Many of these imperfections will be discussed in this section.

8.2.2 Sampling: Continuous-Time and Discrete-Time Representation

Many signals in the physical world around us are continuous (i.e., analog) in nature. Familiar examples of real-world analog signals include sound waves, light intensity, temperature, and pressure. Many modern electronic systems, such as the cellular telephone example in Chapter 1, must convert the continuous signals in the physical world into discrete digital representations

Figure 8.1. Various test signals associated with a voice-band interface circuit.



compatible with digital storage, digital transmission, and mathematical processing. Continuous signals are often described by mathematical equations, such as

$$v(t) = A\ \sin\left(2\pi f_o t + \phi\right) \tag{8.1}$$

where $v(t)$ is a continuous function of time t, whose value in this particular case changes in a sinusoidal manner with amplitude A, frequency f_o, and phase shift ø.

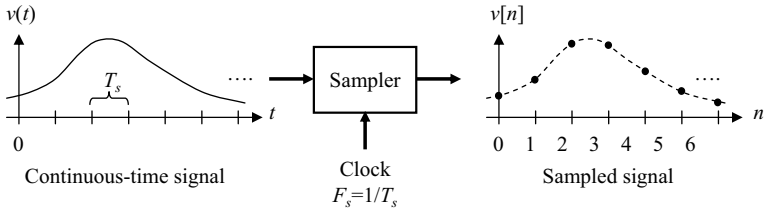
Sampling is a process in which a continuous-time signal is converted into a sequence of discrete samples uniformly spaced at intervals of T_s seconds, often written as

$$v\left[n\right] = v\left(t\right)\Big|_{t = nT_s} \tag{8.2}$$

where $v[n]$ defines the values of $v(t)$ at the sampling instants defined at $t = nT_s$. Such a process is depicted in Figure 8.2. We refer to T_s as the *sampling period* and its reciprocal $F_s = 1/T_s$, as the *sampling frequency* or *sampling rate,* and n as an arbitrary integer. To simplify our notation, it is common practice to drop the T_s term in the argument of Eq. (8.2) because it is assumed to be constant for all time. The continuous waveform $v(t)$ is said to exist in continuous time, while the sampled waveform $v[n]$ is said to exist in discrete time. r example, substituting Eq. (8.1) into (8.2), we can write

$$v\left[n\right] = A\ \sin\left(2\pi f_o n T_s + \phi\right) = A\ \sin\left(2\pi \frac{f_o}{F_s} n + \phi\right) \tag{8.3}$$

Figure 8.2. Continuous-time signal and its sampled equivalent.



For reasons that will become clear later in this chapter, we often impose the condition that the ratio f_o/F_s be a rational fraction, $f_o/F_s = M/N$, where M and N are integers, allowing one to write

$$v[n] = A \sin\left(2\pi \frac{M}{N} n + \phi\right) \tag{8.4}$$

Discrete signals such as this can then be stored in computer arrays and processed using DSP functions.

Up to this point we have defined a sampled waveform in the discrete-time domain as a sequence of numbers defined by $v[n]$. We can also define a sampled waveform as a continuous function of time. The use of this alternative notation is important in the next section where the samples are converted back into the original continuous-time signal. To enable such a description, we must make use of the concept of impulse functions. Mathematically, an impulse function, denoted by $\delta(t)$, is defined as having zero amplitude everywhere except at $t = 0$, where it is infinitely large in such a way that it contains unit area under its curve, as depicted by the following two rules

$$\delta(t) = 0, \; t \neq 0 \tag{8.5}$$

and

$$\int_{-\infty}^{\infty} \delta(t)\, dt = 1 \tag{8.6}$$

It is important to realize that no function in the ordinary sense can satisfy these two rules. However, we can imagine a sequence of pulse like functions that have progressively taller and thinner peaks, with the area under the curve remaining equal to unity as illustrated inFigure 8.3a. If we take this argument to the limit, letting the pulse width go to zero while the pulse height goes to infinity, then we have what we refer to as an impulse function. It should be obvious from this description that we are going to encounter some difficulty in graphing the impulse function. Hence, an impulse is graphically represented by an arrow whose height is equal to the area (voltage × time) under the impulse, as shown in Figure 8.3b.

An important property of impulse functions is the so-called sifting property, defined by

$$\int_{-\infty}^{\infty} v(t)\, \delta(t - t_0) dt = v(t_0) \tag{8.7}$$

Here the impulse function selects or sifts out a particular value of the function $v(t)$, namely, the value at $t = t_0$, in the integration process. If $v_a(t)$ denotes a signal that has been uniformly sampled every T_s seconds, then we can make use of the sifting property and write the following mathematical representation for $v_a(t)$ in terms of a series of evenly spaced, equally sized

Figure 8.3. Impulse definition.

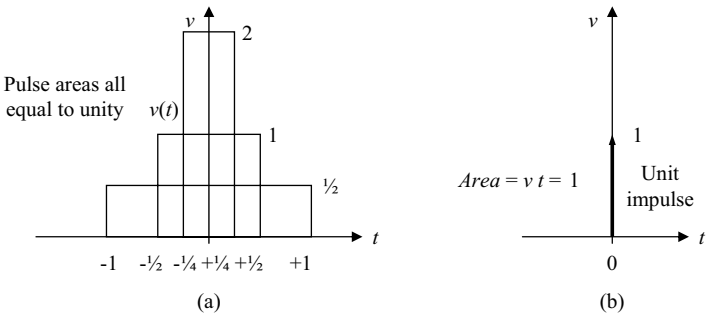
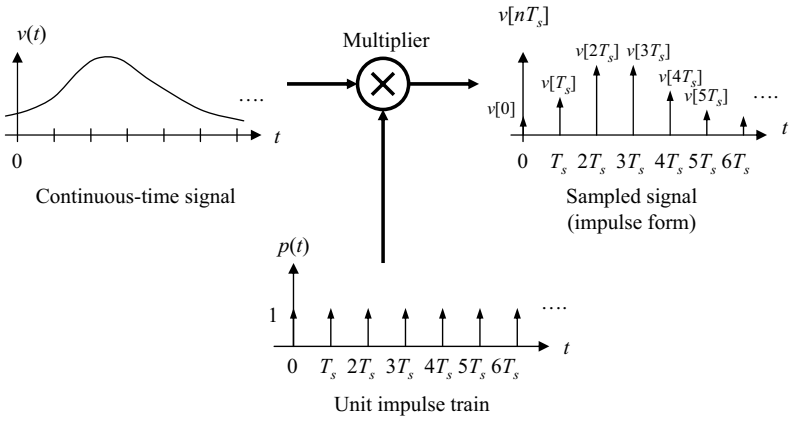


Figure 8.4. Continuous-time representation of a sampled signal as a series of impulses created by multiplying the original continuous-time signal by a unit impulse train.



impulse functions, commonly referred to as a *unit impulse train*:

$$v_a(t) = \sum_{n=-\infty}^{\infty} v(t)\,\delta(t - nT_s) \tag{8.8}$$

Figure 8.4 illustrates the impulse representation of a sequence of samples from a continuous-time signal. Mathematically, the impulses are equal to the multiplication of the continuous-time signal times a unit impulse train.

Equivalently, through direct application of the sifting property of the impulse function, we can write Eq. (8.8) as

$$v_a(t) = \sum_{n=-\infty}^{\infty} v(nT_s)\,\delta(t - nT_s) \tag{8.9}$$

Note that $v_a(t)$ is not defined at the sampling instants because $\delta(t - nT_s)$ is not defined at $t = nT_s$. However, one must keep in mind that the values of $v_a(t)$ at the sampling instants are embedded in the area carried by each impulse function. It should now be clear that $v_a(t)$ and $v[n]$ are different but equivalent models of the sampling process in the continuous-time and discrete-time domains, respectively. In order to keep track of which domain we are working in (i.e., continuous

or discrete), we shall make use of parentheses to encompass the argument of a continuous-time signal, $v(nT_s)$, and square brackets, $v[n]$, to denote a discrete-time signal.

8.2.3 Reconstruction

The inverse operation of sampling is reconstruction. Reconstruction is a process in which a sampled waveform (impulse form) is converted into a continuous waveform by a circuit such as a digital-to-analog converter (DAC) and an anti-imaging filter. In effect, reconstruction is the operation that fills in the missing waveform that appears between samples. In essence, the combined effect of the DAC and filter can be modeled as a single reconstruction operation denoted with impulse response $p(t)$ as shown in Figure 8.5. Mathematically speaking, the reconstruction operation performs interpolation between sampled values. The relationship between the input and output of the reconstruction process can be described by a convolution integral given by

$$v_R(t) = \int_{-\infty}^{\infty} v_a(t-\tau)\,p(\tau)\,d\tau \tag{8.10}$$

Substituting the impulse representation for the input signal $v_a(t)$ from Eq. (8.9), we can write the reconstructed signal several ways as

$$v_R(t) = \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} v(nT_s)\,\delta(t-nT_s-\tau)\,p(\tau)\,d\tau = \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} v(nT_s)\,\delta(t-nT_s-\tau)\,p(\tau)\,d\tau \tag{8.11}$$

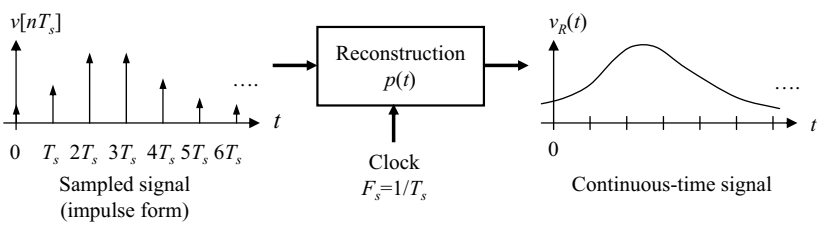
Through the direct application of the sifting property of the impulse function, the output signal from the reconstruction process can then be described as

$$v_R(t) = \sum_{n=-\infty}^{\infty} v(nT_s)\,p(t-nT_s) \tag{8.12}$$

The shape of the impulse response defines the shape of the waveform between adjacent samples. Thus $p(t)$ is commonly referred to as the *characteristic pulse shape* of the reconstruction operation. Equation (8.12) states that each sample is multiplied by a delayed version of $p(t)$ and the resulting waveforms are added together to form $v_R(t)$. In other words, at each sample time $t = nT_s$, a pulse $p(t-nT_s)$ is generated with an amplitude proportional to the sample value $v(nT_s)$. Collectively, all the pulses are summed to form the output continuous signal $v_R(t)$. The general form of Eq. (8.12) appears often in the study of linear, time-invariant continuous-time systems. It is a special case of *convolution* and we say that the output is obtained by *convolving* the continuous-time equivalent signal of $v(nT_s)$ with $p(t)$. The following example will help to illustrate this concept.

Most DACs make use of a square characteristic pulse, as it is easiest to realize in practice. The sum of all shifted and scaled square pulses will result in a "staircase" continuous-time waveform, as shown in Figure 8.7. It is also evident that the staircase waveform is a rather poor

Figure 8.5. Reconstructing a continuous-time signal from a data sequence.



EXAMPLE 8.1

An input sequence $v[n]$ derived from a sinusoid has the following sampled values {0, 0.50, 0.87, 1.00, 0.87, 0.50, 0} corresponding to $n = 0, \ldots, 6$. Everywhere else the sequence is assumed to be zero. Using a triangular reconstruction pulse shape $p(t)$ defined as follows

$$p(t) = \begin{cases} 1 - |t-1|, & 0 \le t < 2 \\ 0 & \text{elsewhere} \end{cases} \tag{8.13}$$

plot the output waveform $v_R(t)$. Assume a sampling period, T_s, of 1 s.

Solution:

To begin, a plot of the characteristic pulse $p(t)$ is shown in Figure 8.6a. As is evident, $p(t)$ is a triangular waveform with a pulse duration that lasts for 2 s and has a peak value of 1. Following Eq. (8.12), with the limits of summation changed from 0 to 6 (as all other sample values are assumed equal to zero), we can write the reconstructed waveform as

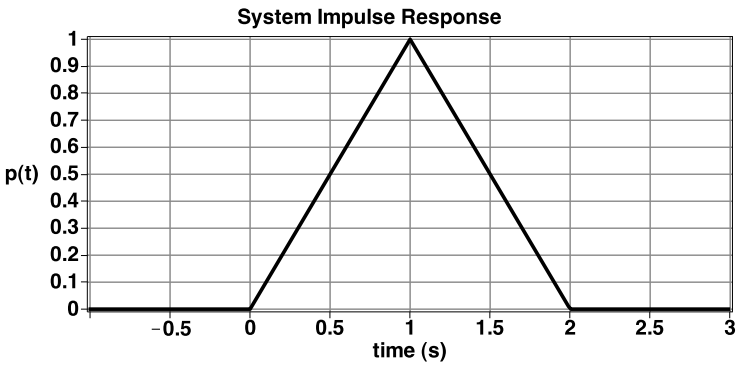
$$v_R(t) = \sum_{n=0}^{6} v(nT_s)p(t - nT_s)$$

or when expanded as

$$v_R(t) = v(0)p(t) + v(T_s)p(t - T_s) + v(2T_s)p(t - 2T_s) + \cdots + v(6T_s)p(t - 6T_s)$$

Now we can substitute an expression for each shifted $p(t)$ according to Eq. (8.13). However, it is more instructive to demonstrate this by superimposing all the pulses weighted by the sampled value on one time axis as shown in Figure 8.6b. At any particular time point, we can add up the contribution from each pulse, and form a single point on the reconstructed waveform. This is shown in the figure for $t = 3.6$ s. This same operation can be repeated for all the remaining time points. The result is a straight-line interpolation between adjacent sampled values.

Figure 8.6. Convolving a triangular pulse with a sequence of sampled values. (a) Triangular impulse system response. (b) System output response.



(a)

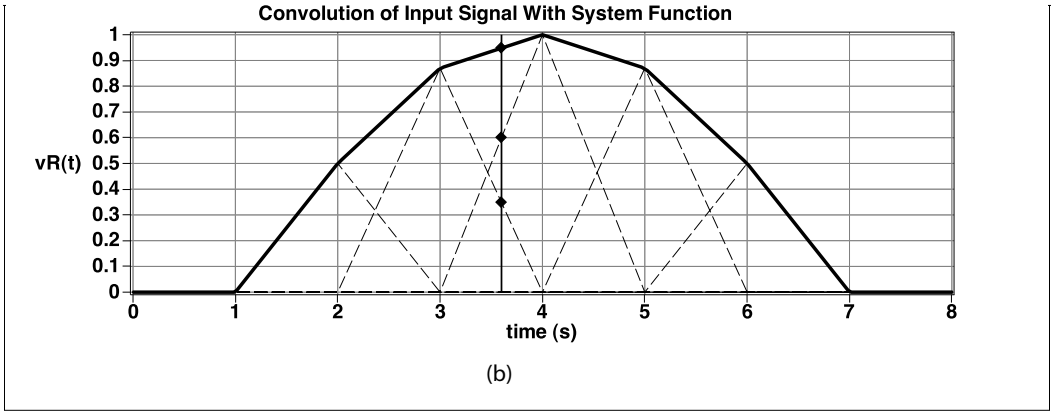
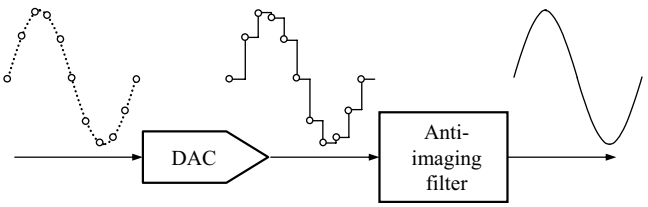


Figure 8.7. Illustrating the reconstruction operation with a DAC and an anti-imaging filter circuit.



approximation of the original waveform. A better approximation would certainly be obtained by increasing the number of steps per period used to reconstruct the waveform. However, the upper frequency range of the DAC limits this approach. It is also clear from Figure 8.7 that the reconstructed waveform $v_R(t)$ contains a large amount of undesirable high-frequency energy, as the reconstructed signal is made up of various sized pulses. To eliminate this high-frequency energy, the DAC is usually followed by a post-filtering circuit, typically one with a low-pass characteristic having a cutoff frequency of at most one-half F_s. Such a filter is known under different names as a smoothing or anti-imaging filter. Collectively, the DAC and the anti-imaging filter are called a *reconstruction filter*.

Cascading a filter after the DAC effectively alters the characteristic pulse $p(t)$ of the reconstruction process and provides a much better approximation to the original waveform. In fact, perfect reconstruction can be obtained if the characteristic pulse of the overall reconstruction process has the following form:

$$p(t) = \begin{cases} 1, & t = 0 \\ \dfrac{\sin\left(\pi t / T_s\right)}{\left(\pi t / T_s\right)}, & \text{otherwise} \end{cases} \tag{8.14}$$

This is a very long pulse, and its infinite length implies that to reconstruct a signal at time t exactly requires all the samples, not just those around that time. Substituting Eq. (8.14) into (8.12) allows us to write an exact interpolation formula for recovering the continuous-time information from the sampled values as

$$v_R(t) = \begin{cases} v(nT_s), & t = nT_s \\ \sum_{n=-\infty}^{\infty} v(nT_s) \dfrac{\sin\left[\dfrac{\pi}{T_s}(t - nT_s)\right]}{\dfrac{\pi}{T_s}(t - nT_s)}, & \text{otherwise} \end{cases} \tag{8.15}$$

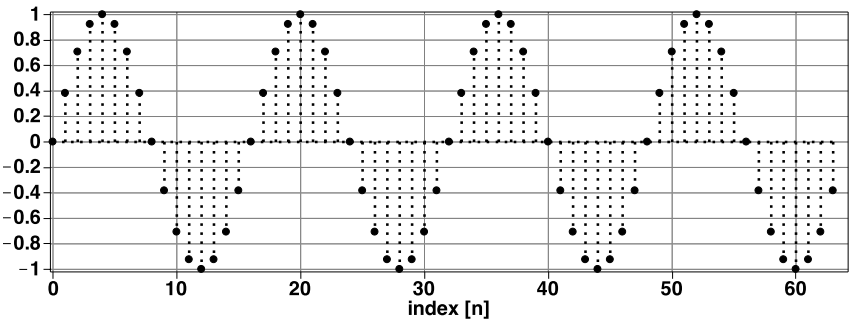
It is interesting to note that $v_R(t)$ is equal to $v(nT_s)$ at all the sampling instants.

In practice, a perfect reconstruction operation can only be approached, not actually realized. Consequently, some imperfections are introduced in the reconstruction process. There are two main sources of errors: (1) aperture effect due to the characteristic pulse shape and (2) magnitude and phase errors related to the anti-imaging filter. Both types of errors lead to

Exercises

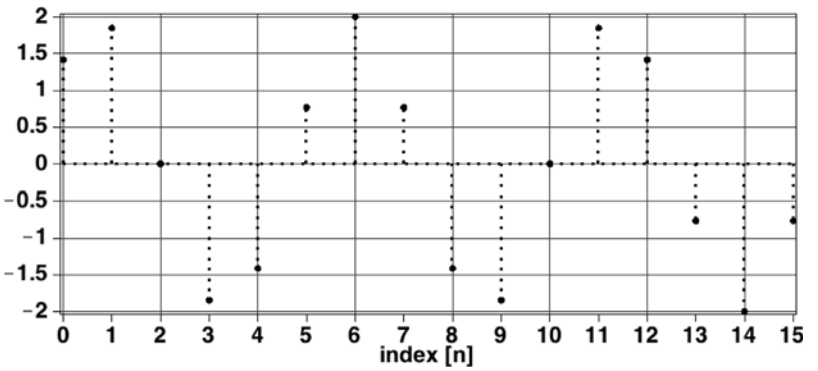
8.1. Using MATLAB or an equivalent software program, plot 64 samples of a sine wave having unity amplitude, zero phase shift, and a period of 16 samples. We shall refer to this plotting range as the observation interval. (The stem command in MATLAB is an effective method for plotting discrete samples as a function of time.)

ANS. Setting $A = 1$, $\phi = 0$, $N = 16$, and $M = 1$, we get:



8.2. Using MATLAB or an equivalent software program, plot 16 samples of a sine wave having amplitude of 2, $\pi/4$ phase shift, and a period of 16/3 samples.

ANS. Setting $A = 2$, $\phi = \pi/4$, $N = 16$, and $M = 3$, we get:



frequency-dependent magnitude and phase errors. If either error is an important parameter of a particular test, then they would need to be measured and corrected using a focused calibration procedure.

8.2.4 The Sampling Theorem and Aliasing

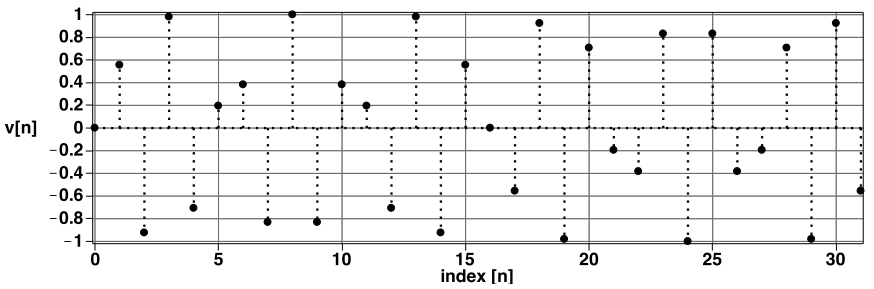
The sampling examples of the previous subsections are all performed in accordance with the *sampling theorem*. Shannon introduced the idea back in 1949 for application in communication systems. For this reason, it is sometimes referred to as the *Shannon sampling theorem*. However, interest and knowledge of the sampling theorem in engineering applications can be traced back to Nyquist in 1928 and as far back as 1915 in the literature of mathematicians. For a historical account of the sampling theorem, interested readers can refer to Jerri[6] for a detailed account. Specifically, the sampling theorem for band-limited signals can be stated in two separate but equivalent ways:

EXAMPLE 8.2

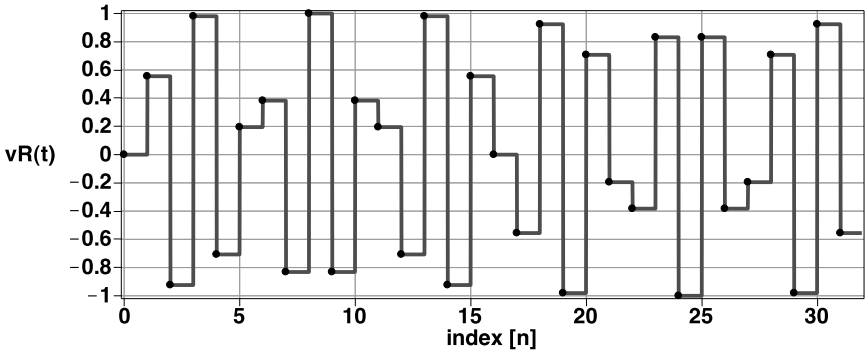
Reconstruct the sample set that results from a single sine wave with parameters: $A = 1$, $\phi = 0$, N = 32, and M = 13 using (a) zero-order hold, (b) first-order hold, and (c) perfect reconstruction.

Solution:

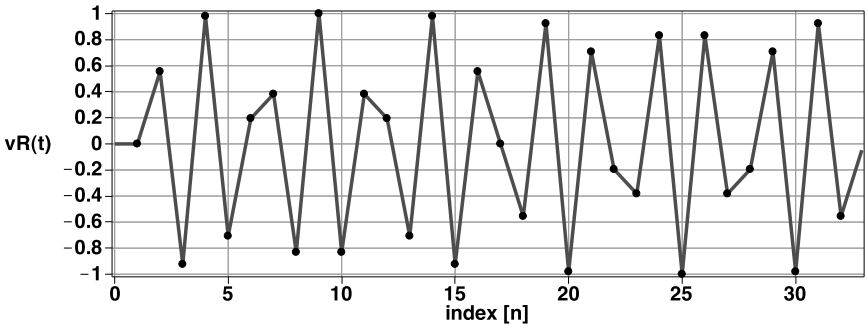
A plot of the 32 samples from the sine wave is shown below:



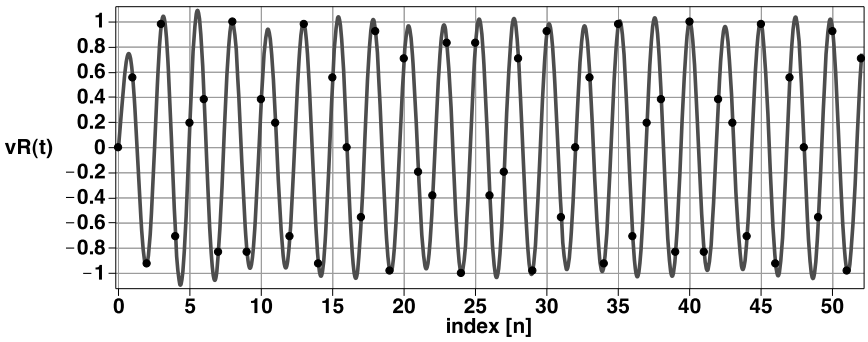
As is evident from this plot, no apparent appearance of a sine wave is present in the data set. This happens when the frequency of the sine wave approaches the frequency $F_s/2$, or in terms of M and N, as M approaches $N/2$. This does not mean that the data set is in error; it is just difficult to see the sine wave shape. If the data set is convolved with a zero-order hold operation, then we obtain the continuous-time waveform shown below. Superimposed on the plot are the original data points. Clearly, the data samples lie on the interpolated waveform. Looking at the interpolated waveform, it would be a stretch to say that this waveform is sinusoidal in shape. However, the information related to the sine wave is contained in the interpolated waveform. We just have to do more post-processing (e.g., linear filtering). We shall discuss this further in the next subsection.



If the data set is convolved with a first-order hold operation, then we obtain the continuous-time waveform shown below:



This interpolated waveform is quite different than that corresponding to the zero-order hold operation, but it too does not exhibit a sinusoidal shape. Finally, we perform the perfect reconstruction operation using the interpolation function described by Eq. (8.14) and obtain the waveform shown below:

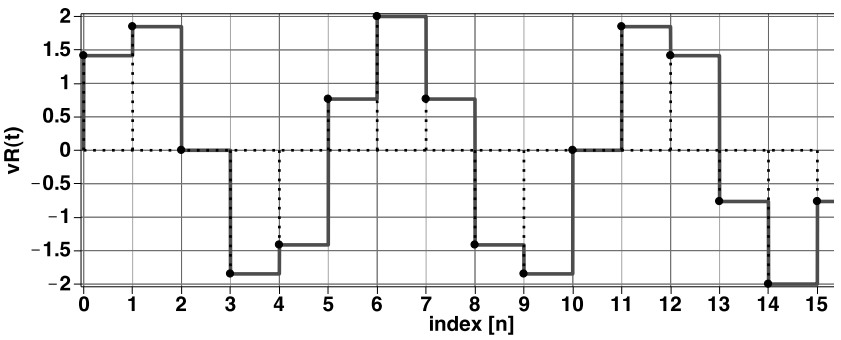


Here the reconstructed waveform has a very distinct sinusoidal shape completing 13 full cycles of the sine wave over the 32 sample points. We note that the waveform has a slight transient component at the start of the waveform but disappears after a few cycles. This waveform is almost identical to the unity amplitude sine wave with zero phase with parameters M = 13 and N = 32. We therefore conclude that zero-order and first-order hold interpolation is not sufficient to recover the original analog waveform from a set of samples. Additional filtering or post processing is necessary.

Exercises

8.3. Reconstruct the sampled signal displayed in Exercise 8.2 using the rectangular pulse described by $p(t)=\begin{cases}1, & 0 \le t < 1\\ 0, & \text{elsewhere}\end{cases}$

ANS.



The Sampling Theorem

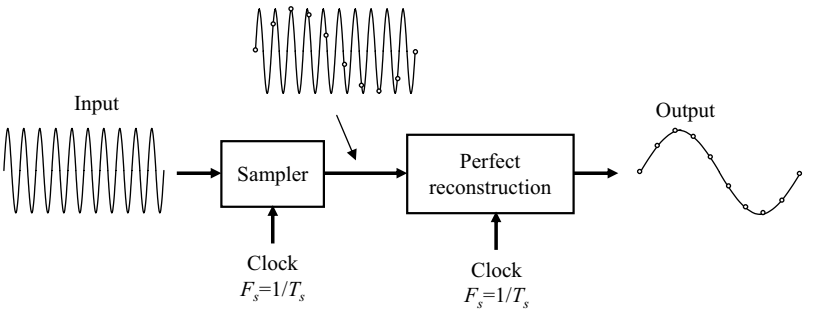
1. A continuous-time signal with frequencies no higher than F_{max} is completely described by specifying the values of the signal at instants of time separated by $1/(2F_{max})$ seconds.
2. A continuous-time signal with frequencies no higher than F_{max} may be completely recovered from knowledge of its samples taken at the rate of $2F_{max}$ per second.

The sampling rate $2F_{max}$ is called the *Nyquist rate*, and its reciprocal is called the *Nyquist interval*. The Nyquist rate is the minimum sampling rate allowable by the sampling theorem. Although somewhat confusing at times, the *Nyquist frequency* refers to F_{max}.

The first part of the sampling theorem is exploited by ATE digitizers. Part 2 of the theorem is exploited by waveform generators. For example, a 10-kHz sine wave appearing at the output of a DUT can theoretically be sampled by the digitizer at 20.1 kHz with no loss of signal information. However, if it is sampled at a slightly lower frequency of 19.9 kHz, specific information about its characteristics are lost. To better understand this, consider the setup shown in Figure 8.8 consisting of a sampler driven by a sine wave, followed by a perfect reconstruction operation.

Ideally, if Shannon's theorem is satisfied, the output of this arrangement should correspond exactly with the input signal (i.e., have exactly the same amplitude, phase, and frequency). In the case shown here, less than two samples per period are taken from the input sine wave; hence it violates the sampling theorem. Such a waveform is said to be undersampled. Subsequently, the signal reconstructed from these samples, shown on the right-hand side in Figure 8.8, has the same amplitude as the input signal but has a much lower frequency (as an estimate of the reconstruction operation consider joining adjacent samples with straight lines). The sampling and reconstruction process has distorted the input signal. The phenomenon of a higher frequency sinusoid acquiring the identity of a lower-frequency sinusoid after sampling is called *aliasing*.

Figure 8.8. Undersampled sine wave and its reconstructed image.



To avoid aliasing in practice, it is important to limit the bandwidth of the signals that appear at the input to the digitizer to less than the one-half the Nyquist rate. In general, practical signals are not limited to a fixed range of frequencies, but have a frequency spectrum that decay to zero as the frequency approaches infinity. As a result, it is not always clear how to satisfy the sampling theorem. To avoid this ambiguity a low-pass *anti-aliasing* filter is placed before the digitizer to attenuate the high-frequency components in the signal so that their aliases become insignificant.

While aliasing is generally an effect that is to be avoided, the process of undersampling has been used to an advantage in several applications. As we will see in Section 11.3, undersampling is used to extend the measurement capabilities of an arbitrary waveform digitizer. Aliasing may also be advantageously utilized by a DUT as part of its normal operation. The cellular telephone base-band interface is one such example that might use undersampling to convert high-frequency inputs to lower-frequency signals to be digitized by a slow ADC.

Finally, we would like to address a commonly asked question: What happens if we sample a sinusoidal at exactly twice its frequency? The answer is that information may be lost. To understand this, consider that an arbitrary sinusoidal (i.e., one with arbitrary amplitude and phase) can always be represented as the sum of a sine and cosine signal operating at the same frequency:

$$C\sin\left(2\pi f_o t+\phi\right)= A\cos\left(2\pi f_o t\right)+ B\sin\left(2\pi f_o t\right) \tag{8.16}$$

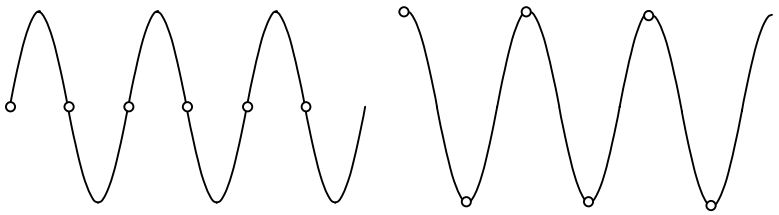
Therefore, analyzing the effect of sampling a sine and cosine signal allows us to generalize the result for a signal having an arbitrary phase, ϕ. Figure 8.9 illustrates the samples derived from a sine and cosine signal sampled at twice their frequency. As is evident, all the samples from the sine wave are zero, whereas those from the cosine signal are not. Clearly, any information contained in the sine wave such as its amplitude would be lost and unobtainable from the samples. We can therefore conclude that one should not attempt to sample at exactly twice the Nyquist rate.

8.2.5 Quantization Effects

Mathematical sampling can be achieved with no loss of signal quality. A computer can come very close to mathematical perfection. For example, the following MATLAB routine can be used to create 64 samples of a sine wave with unity amplitude and zero phase shift:

```
pi = 3.14159265359;
for k = 1:64,
   v(k) = 1*sin(2*pi/64*k);
end;
```

Figure 8.9. Sine and cosine waves sampled at twice the signal frequency.



As the time index k is incremented in unit steps, the sampling period is by default equal to unity, resulting in a unity sampling frequency. Therefore, the frequency of the sampled sinusoid is 1/64 Hz, as $M = 1$ and $N = 64$. The quality of the sine wave is limited only by the tiny amounts of mathematical error in the computation process. This sampling process would result in a nearly perfect sampled representation of the sine wave. It would have almost no distortion and very little noise. The ADC included in a digitizer, on the other hand, will always introduce some amount of noise and distortion. The noise introduced by an ADC can be classified as (1) quantization noise and (2) circuit related noise such as thermal and shot noise. Distortion, on the other hand, is a result of nonlinear circuit behavior and component mismatches.

In a perfectly designed and manufactured ADC, the majority of the noise will be caused by the quantization error of the conversion process. Figure 8.10 shows a set of samples obtained from a sine wave that has been digitized by a 4-bit ADC. For example, the quantized waveform in Figure 8.10 could be stored in a computer memory as the sample set {7,11,14,14,11,7,4, 1,1,4, 7,11,14,14,11,7,4,1,1,4,7,11,14,14,11,7,4,1,1,4}. Also shown in Figure 8.10 is the original analog waveform superimposed on a regular spaced set of grid lines, together with an expanded view of a single sample shown on the right-hand side.

The vertical grid lines correspond to the sampling instances, with time increasing from left to right. The horizontal grid lines correspond to the ADC input decision levels with corresponding output code level identified. The distance between adjacent horizontal grid lines is the LSB step size (V_{LSB}). A single LSB step size sets the largest distance that the ADC output will be from a sample obtained directly from the original waveform (see the expanded view on the right of Figure 8.10). In general, an ideal D-bit ADC with a full-scale analog input range of V_{FSR} whose quantization operation is based on rounding or truncation has an LSB step size of

$$V_{LSB} = \frac{V_{FSR}}{2^D - 1} \tag{8.17}$$

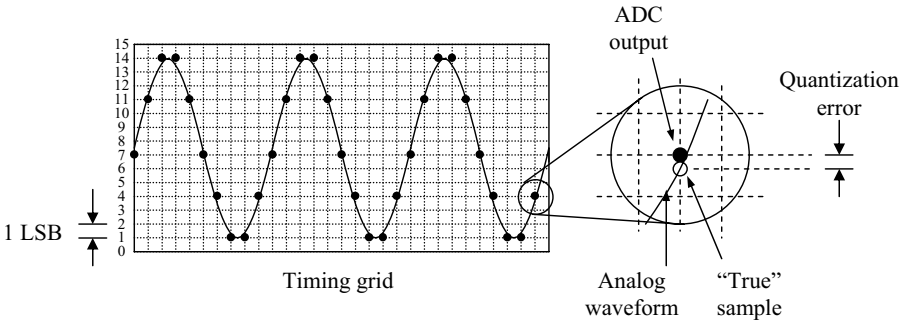
If the ADC quantization process is one where all ideal code widths are of equal length, then the LSB step size will be given by

$$V_{LSB} = \frac{V_{FSR}}{2^D} \tag{8.18}$$

See Section 6.1 of DAC Testing for further details about the various ADC transfer characteristics.

The quantization errors in Figure 8.10 do not look especially severe at first glance. However, if we were to reconstruct a continuous-time waveform from these samples, the analog waveform would contain a significant noise component as illustrated in Figure 8.11. If we separate the errors from the quantized samples, we can see how much noise the quantization process has introduced.

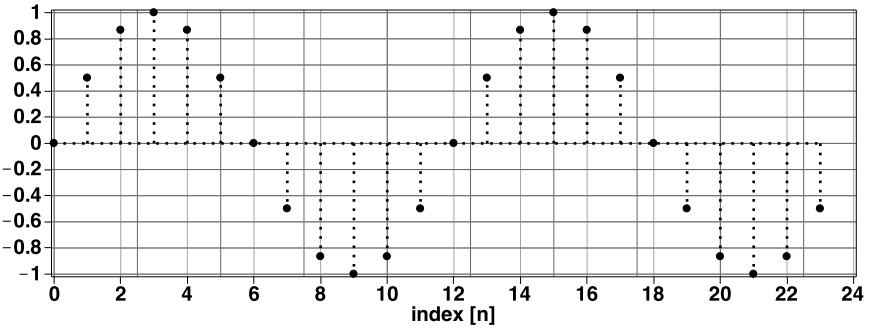
Figure 8.10. Quantized sine wave samples.



Exercises

8.4. To illustrate the effects of aliasing, compare 24 samples of a sinusoid with unity amplitude, zero phase shift, and a period of 12 s derived using a sampling rate of 1 Hz and a sampling rate of 1/8 Hz. Use MATLAB or an equivalent software program for your analysis.

ANS. Setting A=1, φ=0, f_o=1/12, and F_s=1, we get:



SETTING A = 1, $\phi = 0$, f_o = 1/12, and F_s = 1/8, we get

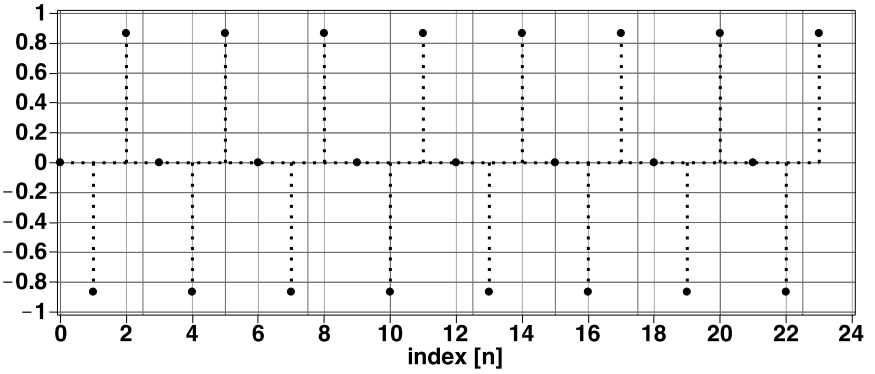


Figure 8.11. Illustrating the noise component that is associated with a quantization operation.

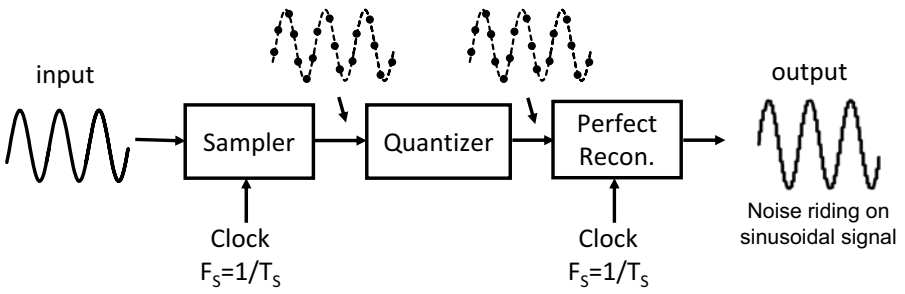
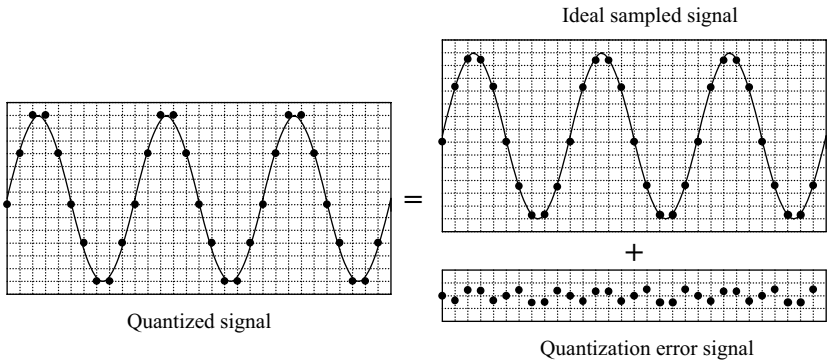


Figure 8.12. Representing the quantized waveform as a sum of the original sampled signal and a quantization error signal.



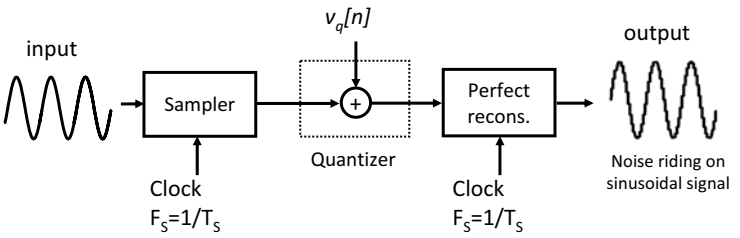
In Figure 8.12, the quantized waveform is equal to the sum of the ideal sampled waveform and an error waveform. The error waveform $v_q[n]$ is the quantization error or noise added by ADC quantization process based on rounding. Statistically speaking, the quantization errors of a random input signal exhibit a uniform probability distribution[5] from $-\frac{1}{2}\, V_{LSB}$ to $+\frac{1}{2}V_{LSB}$, assuming a perfect ADC. Moreover, the ideal quantization error sequence $v_q[n]$ resembles a random sequence having an average and RMS value given by

$$v_{q-AVE} = 0 \quad \text{and} \quad v_{q-RMS} = \frac{V_{LSB}}{\sqrt{12}} \tag{8.19}$$

A truncating ADC has the exact same RMS error value but the average value will be $V_{LSB}/2$. In a statistical sense, we can model the behavior of the quantizer as a summation element as illustrated in Figure 8.13 where one input to the quantizer is the sampled signal and the other the error sequence with average and RMS values given by Eq. (8.19). The corresponding ADC output signal will contain a noise component with an RMS value of $V_{LSB}/\sqrt{12}$. This same level of error will also appear with the signal after perfect reconstruction.

Obviously, quantization error can be reduced by using an ADC with more bits of resolution (consider combining Eqs. (8.17) and (8.19)). Higher resolution would provide more

Figure 8.13. Modeling the quantization operation with a summation element with one input coming from the sampler, the other input corresponding to the error sequence $v_q[n]$.



vertical graticules on the plots in Figure 8.10, reducing the size of each LSB step. Adding an extra bit of ADC resolution reduces the size of each LSB step by one-half, thereby reducing the RMS value of the quantization noise by a factor of two, or 6 decibels (6 dB). A 16-bit ADC is theoretically capable of a 97.76-dB signal-to-noise ratio (SNR) with a full-scale sine wave input. A 15-bit ADC would therefore be capable of 91.76 dB SNR, and so on. (See Chapter 10, "Analog Channel Testing" for an explanation of the decibel unit and SNR measurements.)

EXAMPLE 8.3

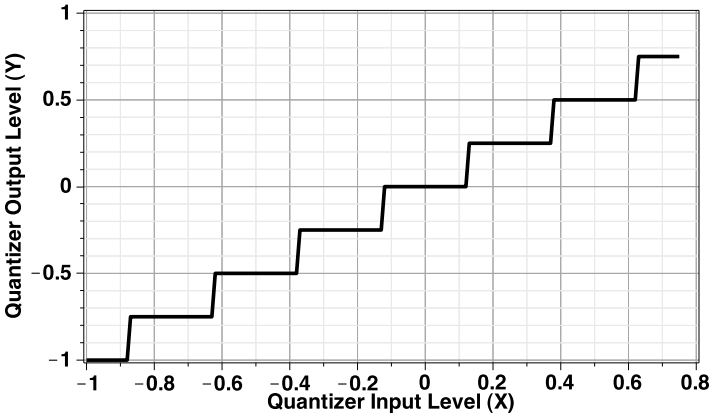
Compute the quantization error sequence that results from exciting a 3-bit ADC with a full-scale amplitude sinusoidal signal of unity amplitude, zero mean, zero phase, ***M*** = 1, and ***N*** = 64. Also, compute the RMS value of the quantization error and compare this result with its theoretical predicted value.

Solution:
To aid us in this investigation we shall make use of the following MATLAB routine for an ideal 3 bit quantizer performing a rounding operation typical of an ADC having a full-scale input range between –1 and +0.75:

```
% 3-Bit Quantizer (-1 <= X <= +0.75)
D = 3;                          % # of bits of resolution
VFSR = 1.75;                    % Full scale voltage range
VLSB = FS/(2^D-1);              % Least significant bit step size
Y = round(X/VLSB)*VLSB;         % rounds to nearest integer level, then scale by VLSB
```

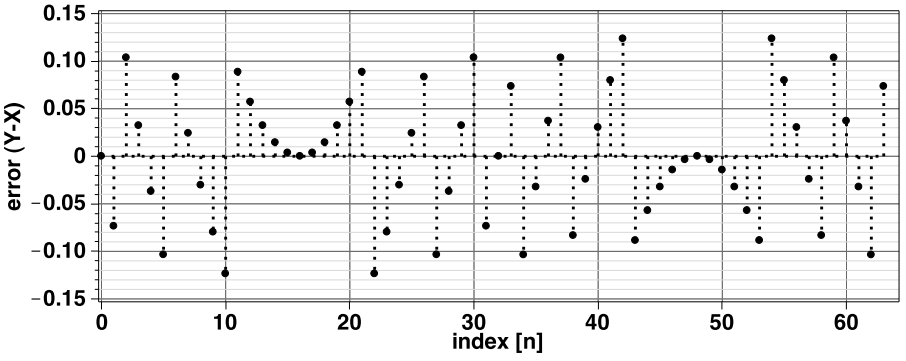
A quantizer is the element of the ADC that limits the continuous input signal, say ***X***, to discrete values denoted by ***Y***–n this case, values of –1, –0.75, –0.5, –0.25, 0, 0.25, 0.5, and 0.75. The ADC would then interpret these levels and provide an output digital representation, for example in a two's complement form. The transfer characteristic, ***Y*** vs. ***X***, for this quantizer is shown in Figure 8.14.

Figure 8.14. Ideal quantizer transfer characteristic.



Now passing a near full-scale sinusoid having the parameters, $A = 0.75$, $\phi = 0$, $N = 64$, and $M = 1$ through the 3-bit quantizer, we get the error sequence shown in Figure 8.15.

Figure 8.15. Quantization error sequence.



Here we see that the error sequence has symmetrical response bounded between –0.125 or +0.125 and has a mean value of –1.0842e–18 or nearly zero. The RMS value of the error is computed to be 0.0670. According to the quantization theory presented earlier, the error sequence should have an average value of 0 and an RMS value of $0.25/\sqrt{12} = 0.0722$ based on an LSB step size of 0.25 V. For all intents and purposes, the results of this simulation agree reasonably well. The discrepancy is largely a result of the quantizer's low resolution of 3 bits. If we increased its resolution, we would discover a much closer correspondence between experiment and theory.

Exercises

8.5.	What is the LSB of an ideal 8-bit ADC that has a full-scale input range of 0–1 V? What is the expected RMS value of the corresponding quantization noise?	ANS. 3.9 mV, 1.13 mV.
8.6.	If an ideal 7-bit ADC has an RMS quantization noise component of 1.4 mV, what is the quantization noise for a 5-bit ADC having an identical full-scale input range?	ANS. 2%.
8.7.	A 4-bit ADC with an analog input range from –1.5 to +1.5 V gives an output of code of 4 for a code range beginning at 0 and ending at 15. What are the minimum and maximum values of the input voltage corresponding to this output code?	ANS. –0.7 V, –0.5 V.

8.2.6 Sampling Jitter

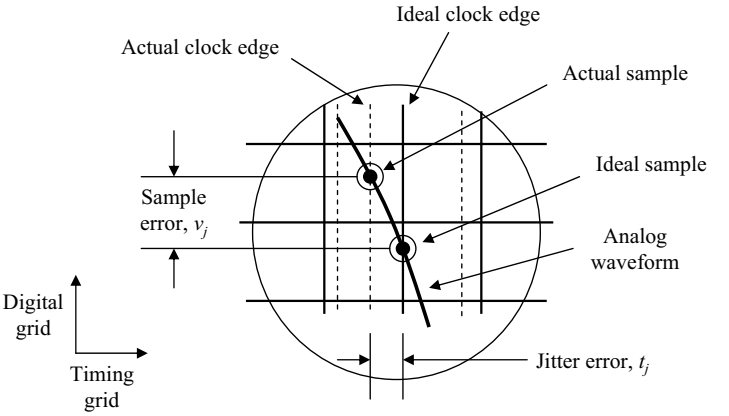
Another source of signal quality degradation is sampling jitter. Jitter is the error in the placement of each clock edge controlling the timing of each ADC or DAC sample. Figure 8.16 illustrates the effect of jitter on the sampling process of an ADC. Here we make use of the same regular spaced grid as that used in Section 8.2.5 except that this time we added an additional set of vertical dotted lines to indicate the actual clock edge subject to random clock jitter.

As is evident in this situation, the actual sample can differ quite significantly from the ideal sample and the size of this error is proportional to the magnitude of the jitter. Mathematically, we can calculate the effects of jitter on the samples obtained by an ADC by associating jitter with a random timing variable, which we shall denote as t_j, and adding it to the sampling expression given in Eq. (8.2) according to

$$v[n] = v(t)\big|_{t = nT_s + t_j} \tag{8.20}$$

Due to the nature of t_j, $v[n]$ is now a random variable as well. Calculating the effects of jitter can become mathematically complicated in all but the simplest examples. One example that allows us to draw some useful conclusions is the study of jitter on the sample points of a single

Figure 8.16. Illustrating the effect of clock jitter on the sampling process.



sinusoid with peak amplitude A_o and frequency f_o. The phase shift is assumed equal to zero without loss of generality. Without jitter, the sample points are

$$v[n] = v(t)\Big|_{t = nT_s} = A_o \sin(2\pi f_o n T_s) \tag{8.21}$$

With jitter present, according to Eq. (8.20), the samples become

$$v[n] = v(t)\Big|_{t = nT_s + t_j} = A_o \sin\left(2\pi f_o \left(nT_s + t_j\right)\right) \tag{8.22}$$

We can separate this expression into two parts, one that includes the deterministic component and the other due to jitter. To see this, consider using the trigonometric identity $\sin(A+B) = \sin(A)\cos(B) + \cos(A)\sin(B)$ so that we can rewrite Eq. (8.22) as

$$v[n] = A_o \sin(2\pi f_o n T_s)\cos(2\pi f_o t_j) + A_o \cos(2\pi f_o n T_s)\sin(2\pi f_o t_j) \tag{8.23}$$

Since the magnitude of the jitter t_j is assumed to be small compared to the sampling period T_s, we can approximate Eq. (8.23) as

$$v[n] \approx A_o \sin(2\pi f_o n T_s) + A_o 2\pi f_o t_j \cos(2\pi f_o n T_s) \tag{8.24}$$

Here we made use of the fact that when x is small, $\cos(x) \approx 1$ and $\sin(x) \approx x$. Now we have the jitter term separated from the deterministic term, allowing us to claim that the error in the sample due to jitter, denoted as v_j, is

$$v_j[n] \approx A_o 2\pi f_o t_j \cos(2\pi f_o n T_s) \tag{8.25}$$

Recognizing that the derivative of a sine wave is a cosine wave further allows us to write the jitter-induced error in terms of the magnitude of the jitter and the slope of the signal at the sample point

$$v_j[n] \approx \left[\left. \frac{dv(t)}{dt} \right|_{t = nT_s} \right] \cdot t_j \tag{8.26}$$

This result should be readily apparent from Figure 8.16. It suggests that a timing error will induce a larger sample error at the rapidly rising or falling points of a sine wave than at its peak or trough.

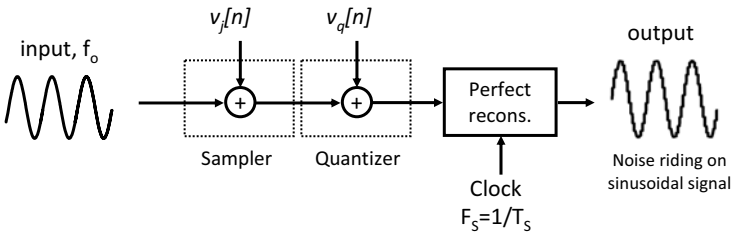
Assuming that the jitter t_j has an RMS value of t_{j-RMS} and is independent of $v(t)$, we can approximate the RMS value of the error sequence $v_j[n]$ as the product of the RMS value of t_j and the RMS value of the derivative of $v(t)$ at each sampling instant. For a sampled sinusoidal signal with peak amplitude A_o and frequency f_o, the RMS value of the jitter-induced error is

$$v_{j-RMS} \approx \frac{2\pi A_o f_o}{\sqrt{2}} t_{j-RMS} \tag{8.27}$$

Assuming the jitter-induced error is independent of the quantization-induced error, we can model each effect separately as shown in Figure 8.17. The total combined RMS error can then be combined in a statistical sense as

$$v_{ERROR-RMS} = \sqrt{\left(\frac{2\pi \cdot A \cdot f_o}{\sqrt{2}} t_{j-RMS} \right)^2 + \left(\frac{V_{LSB}}{\sqrt{12}} \right)^2} \tag{8.28}$$

Figure 8.17. Modeling the sampling and quantization process as a linear combination of two independent error sequences.



Consider the following ADC example.

EXAMPLE 8.4

A 12-bit ADC operating at a sampling rate of 20 MHz is used to sample a signal with a bandwidth of 500 kHz. The ADC operates off a single supply of 5 V. What is the worst-case RMS value of the error associated with the sampled signal when the clock signal has an RMS noise component of 500 ps?

Solution:
A 12-bit ADC assumed to have a full-scale range of 5 V has an LSB step size given by

$$V_{LSB} = \frac{5\text{ V}}{2^{12}-1} = 1.221\text{ mV}$$

The RMS error associated with the ADC conversion process accounting for both quantization error and clock jitter with a worst-case amplitude signal given by 5 V peak-to-peak is therefore

$$v_{ERROR-RMS} = \sqrt{\left(\frac{2\pi \cdot 5\text{ V}/2 \cdot 500\text{ kHz} \cdot 500\text{ ps}}{\sqrt{2}}\right)^2 + \left(\frac{1.221\text{ mV}}{\sqrt{12}}\right)^2} = 2.79\text{ mV}$$

It is interesting to note that if the above error was induced by quantization error alone, then the effective number of bits this ADC would achieve is found from

$$\sqrt{12} \times v_{ERROR-RMS} = \frac{V_{FSR}}{2^{D_{effective}}-1} \Rightarrow D_{effective} = \log_2\left[\frac{V_{FSR}}{\sqrt{12} \times v_{ERROR-RMS}} + 1\right] \quad \textbf{(8.29)}$$

Substituting the appropriate parameters, we find $\boldsymbol{D_{effective}}$ = 9 bits. Hence, clock jitter has induced a loss of 3 bits from ideal ADC operation.

At this point in our discussion we can use this result to set a limit on the maximum tolerable jitter allowable based on the ADC's speed and resolution. We first have to define the amount of jitter-induced noise that we are willing to tolerate. Let us define an LSB step size upper limit on the tolerable amount of jitter-induced noise, that is,

$$v_{j-RMS} < V_{LSB} = \frac{V_{FSR}}{2^D - 1} \quad \textbf{(8.30)}$$

Substituting Eq. (8.27) and rearranging allows us to bound the jitter according to

$$t_{j-RMS} < \frac{V_{FSR}}{\sqrt{2}\pi A_o f_o \left(2^D - 1\right)} \tag{8.31}$$

Further more, if we assume a full-scale input sinusoid, $V_{FSR} = 2A_o$, then we can find a lower limit on the maximum allowable jitter given by

$$t_{j-RMS} < \frac{\sqrt{2}}{\pi f_o \left(2^D - 1\right)} \tag{8.32}$$

Conversely, for a D-bit ADC having an RMS sampling jitter $t_{j\text{-}RMS}$, the maximum sampling frequency that can be used (i.e., $F_{s\text{-}MAX} = 2 f_{o\text{-}MAX}$) is

$$F_{s-MAX} < \frac{2\sqrt{2}}{\pi t_{j-RMS} \left(2^D - 1\right)} \tag{8.33}$$

or we can conclude that the maximum conversion resolution (expressed in number of bits) available with a maximum sampling frequency $F_{s\text{-}MAX}$ and RMS sampling jitter $t_{j\text{-}RMS}$ is

$$D_{MAX} < \log_2 \left(\frac{2\sqrt{2}}{\pi t_{j-RMS} F_{s-MAX}} + 1 \right) \tag{8.34}$$

Figure 8.18 illustrates the trade-off between ADC effective resolution and RMS clock jitter for a range of sampling frequencies under worst-case conditions of full-range input signal swing and an input frequency set at the Nyquist frequency of $F_s/2$. As is quite evident form this plot, the effective ADC resolution decreases dramatically with increasing clock jitter. Furthermore, to achieve high ADC resolution at high sampling rates requires very low level of clock jitter.

The effect of sampling jitter on the operation of a DAC can be described by similar mathematical expressions derived for the ADC. Consider that the effect of clock jitter on the output of

Figure 8.18. Worst-case tradeoff of the effective ADC resolution and the RMS clock jitter for a range of ADC sampling frequencies.

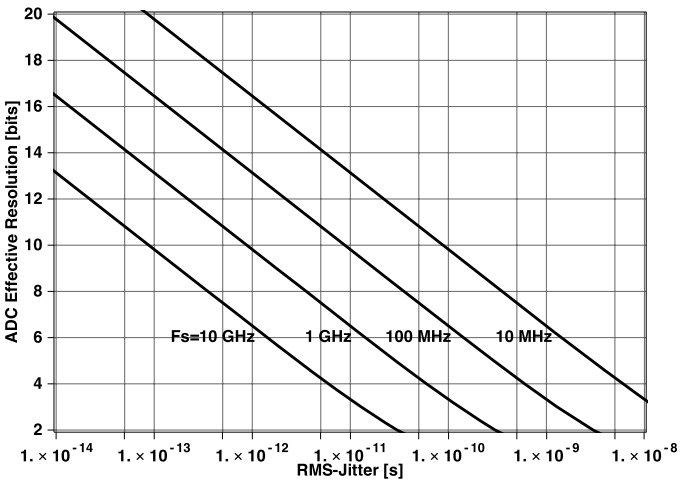


Figure 8.19. The effect of clock jitter on the actual DAC output can be separated into an ideal output and a jitter-induced error signal.

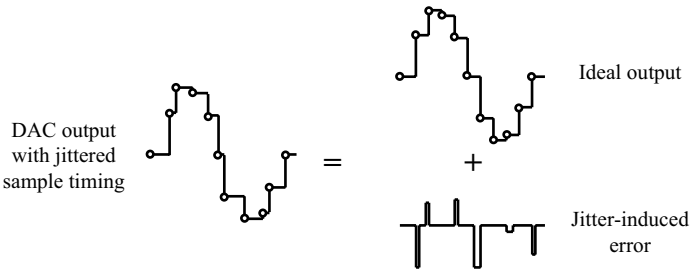
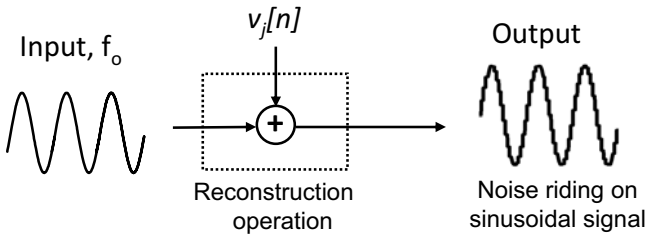


Figure 8.20. Modeling the jitter-induced error that occurs during the reconstruction process.



a DAC can be separated from its ideal operation as shown in Figure 8.19. Here the actual output waveform is separated into an ideal waveform and one that contains the jitter induced noise. Mathematically, the jitter-induced error can be described as

$$v_j(t) = \left[v(nT_s) - v\left((n-1)T_s\right) \right] \left[u\left(t - nT_s - t_j\right) - u\left(t - nT_s\right) \right] \tag{8.35}$$

where $u(t)$ is a unit step function. With an error pulse occurring on average once every clock period, we can consider that the effective energy contributed by each pulse at the sampling instant is

$$e_p[n] = \left(v[n] - v[n-1]\right)^2 t_j \tag{8.36}$$

Furthermore, we can relate this energy back to the original sample value by dividing Eq. (8.36) by T_s; that is, the pulse energy is distributed over a full clock period, and taking the square-root value, then we can write the jitter-induced error as

$$v_j[n] = \left(v[n] - v[n-1]\right)\sqrt{\frac{t_j}{T_s}} \tag{8.37}$$

Recognizing that the difference operation normalized by T_s is a discrete-time representation of differentiation allows us to approximate the jitter-induced error (for high oversampling ratios) as

$$v_j[n] \approx \left[\left. \frac{dv(t)}{dt} \right|_{t=nT_s} \right] \sqrt{t_j T_s} \tag{8.38}$$

This expression is similar to that given for the jitter-induced error of the ADC, except that t_j is replaced by $\sqrt{t_j T_s}$. Hence we can make use of Eqs. (8.27)–(8.34) with the appropriate change of variable. For instance, the RMS value of the jitter-induced error voltage error would be written as

$$v_{j-RMS} = \frac{2\pi \cdot A \cdot f_o}{\sqrt{2}} \cdot \sqrt{t_{j-RMS} T_s} \tag{8.39}$$

In an identical manner as for the ADC conversion process, the DAC reconstruction process can be modeled with as a linear combination of the digital signal and an error sequence whose RMS value is given by Eq. (8.39).

This example serves to illustrate the sensitivity of the DAC to clock jitter and how bad the measurement accuracy can become if clock jitter is not properly accounted for.

If the jitter-induced voltage error is less than the LSB step size of the DAC, then the clock jitter will have little effect on the operation of the DAC. This condition is met when

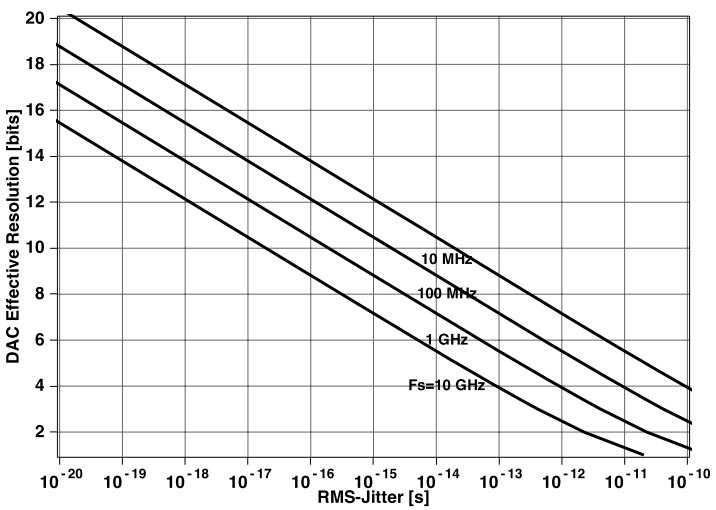
$$t_{j-RMS} \le F_s \left[\frac{\sqrt{2}}{2\pi} \frac{V_{FSR}}{\left(2^D - 1\right)} \frac{1}{A \cdot f_o} \right]^2 \tag{8.40}$$

This was obtained by combining (8.30) with (8.39) and rearranging to obtain Eq. (8.40). Under worst-case conditions (i.e., input amplitude of $V_{FSR}/2$ and $f_o = F_s/2$), one can show that the maximum conversion resolution achievable for a DAC with an RMS clock jitter of t_{j-RMS} is given by

$$D_{MAX} < \log_2 \left(\frac{2}{\pi} \sqrt{\frac{2}{t_{j-RMS} F_{s-MAX}}} + 1 \right) \tag{8.41}$$

Plotting this expression for a range of sampling frequencies we obtain the graph shown in Figure 8.22. In comparison to the equivalent ADC worst-case effective number of bits shown in Figure 8.18, we see that the DAC is much more sensitive to clock jitter. This was also observed in Example 8.5.

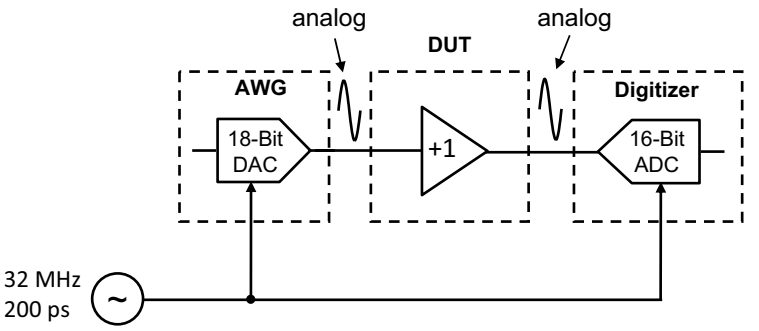
Figure 8.22. Worst-case tradeoff of the effective DAC resolution and the RMS clock jitter for a range of DAC sampling frequencies.



EXAMPLE 8.5

An AWG and DIG in a particular ATE is constructed from an 18-bit DAC and a 16-bit ADC with a full-scale value of 10 V. A 1 V peak sinusoidal signal is to be sourced at 4.5 MHz and applied to a DUT with an expected gain of unity. The AWG and DIG are clocked at 32 MHz as shown in the test setup of Figure 8.21. Measurements have shown that the clock source has a jitter noise component of 200 ps RMS. What is the expected RMS error in the samples collected by the digitizer? What is the effective resolution of the AWG-Digitizer combination?

Figure 8.21. An AWG-Digitizer setup for measuring the AC gain of a DUT.



Solution:

We begin by determining the jitter—induced error associated with the DAC of the AWG. Using Eq. (8.39), we compute the AWG. DAC induced RMS error as

$$v_{AWG-RMS} = \frac{2\pi \cdot A \cdot f_o}{\sqrt{2}} \cdot \sqrt{\frac{t_{j-RMS}}{F_s}} = \frac{2\pi \cdot 1\,\text{V} \cdot 4.5\ \text{MHz}}{\sqrt{2}} \cdot \sqrt{\frac{200\ \text{ps}}{32\ \text{MHz}}} = 49.9\ \text{mV}$$

Next, we compute the corresponding jitter induced RMS error associated with the digitizer-ADC as

$$v_{DIG-RMS} = \sqrt{\left(\frac{2\pi \cdot A \cdot f_o}{\sqrt{2}} t_{j-RMS}\right)^2 + \left(\frac{V_{LSB}}{\sqrt{12}}\right)^2}$$

$$= \sqrt{\left(\frac{2\pi \cdot 1\,\text{V} \cdot 4.5\ \text{MHz} \cdot 200\ \text{ps}}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{12}} \frac{10}{2^{16}-1}\right)^2} = 3.99\ \text{mV}$$

Finally, assuming that the errors are independent, we find that the total RMS error associated with the measurement setup is

$$v_{DIG-RMS} = \sqrt{\left(v_{AWG-RMS}\right)^2 + \left(v_{DIG-RMS}\right)^2} = \sqrt{(49.9\ \text{mV})^2 + (3.99\ \text{mV})^2} = 50.1\,\text{mV}$$

Any sequence of measurements will therefore contain about a 50 mV RMS error. It is obvious that this error will largely be from the AWG reconstruction process (i.e., 50-mV error from the AWG compared to 4 mV from the DIG).

If the AWG-digitization process is assumed to be ideal and is only limited by the quantization error associated with the ADC, then the effective number of bits can be found from

$$D_{effective} = \log_2\left[\frac{V_{FSR}}{\sqrt{12} \times v_{ERROR-RMS}} + 1\right] = \log_2\left[\frac{10\ \text{V}}{\sqrt{12} \times 50\ \text{mV}} + 1\right] = 5.9\ \text{bits}$$

In either the DAC or ADC case, according to Eq. (8.27) doubling the timing jitter doubles the noise level. Also, doubling the signal amplitude or signal frequency doubles the jitter-induced noise. Testers often have particular sampling frequencies or other conditions that produce minimum sampling jitter. For instance, a particular tester may produce minimum jitter if the digital pattern is exercised at the tester's master clock frequency divided by 2N, where N is any integer. As another example, a particular digitizer may operate with minimum jitter when its phase-locked loop phase discriminator input is near 16 kHz. If extremely low noise measurements are to be performed, the test engineer should understand which sampling rates provide the least jitter in each of the tester's instruments and subsystems.

8.3 REPETITIVE SAMPLE SETS

8.3.1 Finite and Infinite Sample Sets

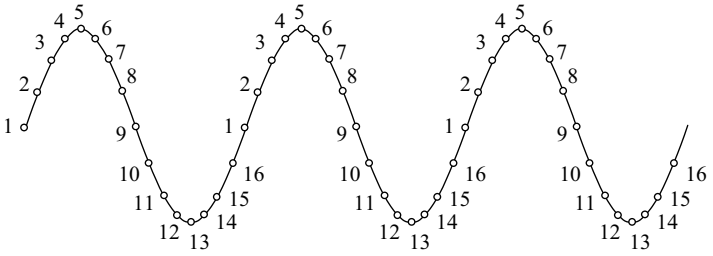
In many mixed-signal systems such as a cellular telephone, the waveforms sampled by the system's ADC sub-blocks are nonrepetitive. In the cellular telephone example, the caller's voice is a random signal that seldom, if ever, repeats. The cellular telephone digitizes the caller's voice and processes the samples in real time in a continuous process. For all intents and purposes, we can consider the cellular telephone sample sets to be infinite in length.

In the DSP-based testing environment, on the other hand, signals are often created and measured using a finite sample set of a few hundred or a few thousand samples. If desired, the finite sample sets in mixed-signal testers can be repeated endlessly, allowing easier debugging with spectrum analyzers and oscilloscopes. During production testing, however, the sample sets are only allowed to repeat long enough to collect the necessary measurement information. The use of repetitive, finite sample sets drives a number of ATE-specific limitations which the test engineer must understand. For example, Figure 8.23 shows a short sequence of 16 samples that repeats endlessly. Notice how sample 16 feeds smoothly into sample 1 at the end of each sequence. This smooth wraparound results from a property known as *coherence*. Coherence is one of the most important enabling factors for fast and accurate DSP-based test.

Exercises

8.8.	What is the RMS value of the error induced by an ADC having an RMS sampling jitter of 100 ps while measuring a 1-V amplitude sinusoid with a frequency of 100 kHz?	ANS. 44.4 μ*V*.
8.9.	What is the maximum sampling jitter that a 6-bit ADC can tolerate when it has a full-scale input range of 0–3 V and is converting a 100-kHz, 1-V peak sinusoid?	ANS. 0.107 μs.
8.10.	What is the maximum sampling jitter that a 5-bit DAC can tolerate when it has a maximum sampling rate of 10 MHz?	ANS. 84.3 ps.
8.11.	If a 6-bit DAC has a sampling jitter of 500 ps RMS, what is its maximum sampling rate?	ANS. 408.5 kHz.
8.12 .	If an ADC is controlled by a clock circuit with a minimum clock period of 1 μs and RMS jitter of 2.5 ns, what is the maximum conversion resolution possible with the ADC?	ANS. 8.5 bits.

Figure 8.23. Finite sample set, repeated indefinitely.



8.3.2 Coherent Signals and Noncoherent Signals

In the example waveform of Figure 8.23, the last sample of the first iteration wraps smoothly into the first sample of the second iteration because there is exactly one sine wave cycle represented by the 16 samples. If we reconstruct this sample set at a sampling frequency F_s, then the sine wave would have a frequency of $F_s/16$. This frequency is known as the *fundamental frequency* or *primitive frequency*, F_f. In general, the fundamental frequency F_f of N samples collected at a sampling rate of F_s is

$$F_f = \frac{F_s}{N} \tag{8.42}$$

The period of the fundamental frequency is called the *primitive period* or *unit test period* (UTP)

$$UTP = \frac{1}{F_f} \tag{8.43}$$

The amount of time required to collect a set of N samples at a rate of F_s is also equal to one UTP

$$UTP = \frac{N}{F_s} \tag{8.44}$$

In practice, it usually takes an extra fraction of a UTP to allow the DUT and ATE hardware to settle to a stable state before a sample set is collected.

The fundamental frequency is often called the *frequency resolution*. The reason for this alternate terminology is that the only coherent frequencies that can be produced with a repeating sample set are those frequencies that are integer multiples of the fundamental frequency. Hence, in terms of N and the sampling frequency, the coherent frequencies F_c are

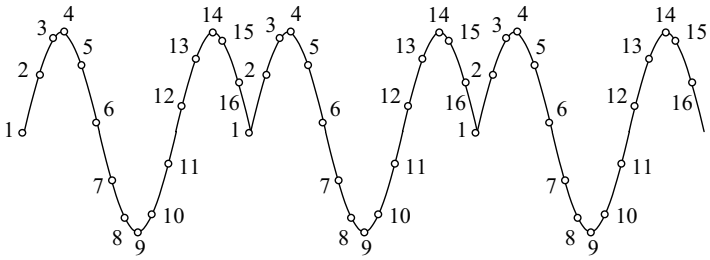
$$F_c = M\frac{F_s}{N} \tag{8.45}$$

where M is an integer 0, 1, 2,…, $N/2$. The astute reader will recognize that we first made use of coherent frequencies in Section 8.2.1 in the development of Eq. (8.4), where $f_o = F_c$.

As an example, if we source the samples in Figure 8.23 at a rate of 16 kHz, then the fundamental frequency would be 16 kHz/16 = 1 kHz. The sine wave in Figure 8.23 would appear at 1 kHz. The next-highest frequency we could produce with 16 samples at this sampling rate is 2 kHz. If we wanted to produce a 1.5-kHz sine wave, then we would have a noncoherent sample set as shown in Figure 8.24.

If we wanted to produce a 1.5-kHz sine wave using a coherent sample set, then we would have to choose a sampling system with a fundamental frequency equal to 1.5 kHz/N, where N is

Figure 8.24. Noncoherent sample sets cannot be looped properly.



any integer. We might choose F_f = 500 Hz, for example, and then use the third multiple of the fundamental frequency to produce the 1.5-kHz sine wave. A fundamental frequency of 500 Hz could be achieved using 32 samples instead of 16 (16 kHz/32 = 500 Hz). We would then calculate a sine wave with three cycles in 32 samples according to

```
pi = 3.14159265359;
for k = 1:32,
   sinewave(k) = sin(2*pi*3/32*(k-1));
end
```

Since the fundamental frequency determines the frequency resolution of a measurement, it might seem that minimizing the fundamental frequency would be a great idea. In the absence of test time constraints, a fundamental frequency of 1 Hz would provide good flexibility in test frequency choice. Remember, though, that the UTP drives the test time. Since one UTP is equal to $1/F_f$, a 1-Hz frequency resolution would require 1 s of data collection time. For most production tests, this would be unacceptable.

Many test situations call for the application of a coherent multitone signal to excite a device. Such a signal is created by simply adding together a set of *P* unique sine waves (i.e., having different coherent frequencies) according to the following formula

$$v[n] = \sum_{i=1}^{P} A_i \sin\left(2\pi \frac{M_i}{N} n + \phi_i\right) \tag{8.46}$$

Here each sine wave is assigned a unique amplitude A_i, phase shift ϕ_i, and frequency designated by $(M_i/N)F_s$. The integers represented by M_i are commonly referred to as the *Fourier spectral bins*.

Any signal made up of a sum of coherent signals is also coherent. If one or more of the frequency components are noncoherent, though, the entire waveform will be noncoherent. Although noncoherent sample sets cannot be used to generate continuous signals through a looping process, they can be analyzed with DSP operations using a preprocessing operation called *windowing*. However, windowing is an inferior production measurement technique compared to coherent, nonwindowed testing. Windowing will be discussed in Chapter 9, "DSP-Based Testing."

Returning to the 16-kHz sampling example, we could create a multitone signal with frequencies at 1.5, 2.5, and 3.5 kHz using an expanded calculation given by the following MATLAB routine

```
pi = 3.14159265359;
phase1 = 0, phase2 = 0, phase3 = 0;
```

```
for k = 1:32,
   multitone(k) = sin(2*pi*3/32*(k-1) + phase1*pi/180)...
       + sin(2*pi*5/32*(k-1) + phase2*pi/180)...
       + sin(2*pi*7/32*(k-1) + phase3*pi/180;
end
```

The endpoints of this waveform would wrap smoothly from end to beginning because the waveform is coherent. The multitone signal calculated would be described as a three-tone multitone waveform with equal amplitudes at the third, fifth, and seventh spectral bins.

8.3.3 Peak-to-RMS Control in Coherent Multitones

Notice that in the multitone example in Section 8.3.2, all the frequency components are created at the same phase (0 degrees). The problem with this type of waveform is that it may have an extremely large peak-to-RMS ratio, especially as the number of tones increases. Consider the 7-tone multitone signal in Figure 8.25. The first waveform consists entirely of sine waves, while the second waveform consists entirely of cosine waves. These waveforms exhibit a spiked shape that is unacceptable for most testing purposes since it tends to cause signal clipping in the DUT's circuits.

The peak-to-RMS ratio of a multitone can be adjusted by shifting the phase of each tone to a randomly chosen value. The waveform in Figure 8.26 shows how randomly selected phases for

Figure 8.25. Pure sine and pure cosine seven-tone multitones.

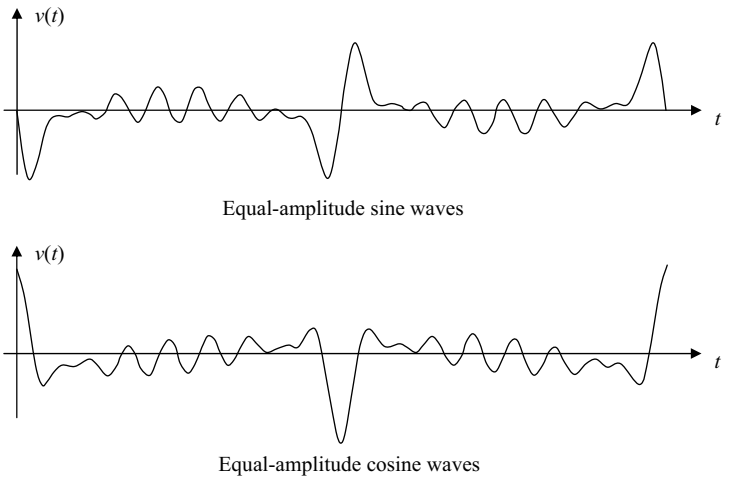
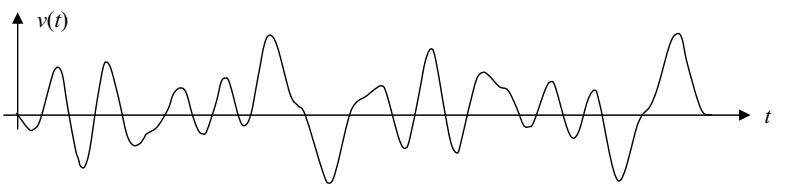


Figure 8.26. Seven-tone multitone created with random phases.



the seven tones of Figure 8.25 produces a much less "spikey" waveform. Unfortunately, there is no equation to calculate phases to give a desired peak-to-RMS ratio. In many test programs, phases are chosen using a pseudorandom number generator with a uniform probability distribution between 0 and 2π radians. If the desired peak-to-RMS ratio is not achieved with one set of pseudorandom phases, then the program tries again until the desired ratio is found. These phase values can be generated each time the program loads, or they can be hard-coded into the test program once they have been determined through trial and error. This second approach prevents a pseudorandom algorithm from choosing one set of phases on a given day and another set of phases at a later date. For example, this might happen when an upgraded tester operating system includes a change to the pseudorandom algorithm. Theoretically, a different set of phases should not cause any shift in measurement results, but the use of hard-coded phases removes one more unknown factor from the measurement correlation effort.

What is the ideal peak-to-RMS ratio for a multitone signal? At first it might seem that it would be best to let the pseudorandom process search for a minimum peak-to-RMS ratio. This would provide the largest RMS voltage for a given peak-to-peak operating range. Larger RMS signals provide better noise immunity and improved repeatability. But this kind of signal is susceptible to large shifts in peak-to-RMS ratio if any of the filters in the ATE tester or DUT cause frequency-dependent phase shifts. A change in peak-to-RMS ratio could lead to a clipped signal, which would ruin the measurement accuracy.

In many end applications, the DUT will usually see a peak-to-RMS ratio of about 10 to 11 decibels (a ratio of about 3.35:1). Although the 10- to 11-dB range appears in many data sheets without explanation, it is based on the approximate peak-to-RMS levels encountered in typical analog signals. This range is roughly equal to the peak-to-RMS ratio of broadband signals having near-Gaussian-distributed amplitudes and random phases. As it happens, this ratio also tends to produce a multitone whose peak-to-RMS ratio is least sensitive to phase shifts from filters. For this reason, the pseudorandom phase selection process should be set to search for a peak-to-RMS ratio of between 10 and 11 decibels. A multitone signal must contain at least six tones to hit a peak-to-RMS ratio of 3.35:1. For signals having fewer tones, the target ratio is not terribly important, but it is still a good idea to use pseudorandom phase shifts for the tones rather than adding pure sine or cosine waveforms.

8.3.4 Spectral Bin Selection

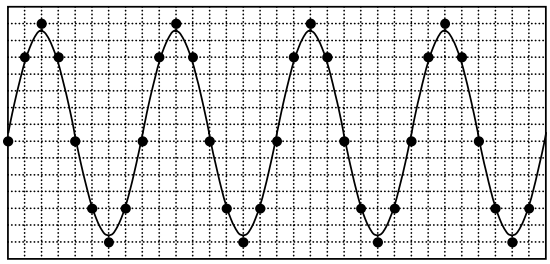
One of the common mistakes a novice test engineer makes is to choose spectral bins by simply calculating the nearest integer multiple of the fundamental frequency. For example, if the test engineer wanted a 2-kHz sine wave using a 16-kHz sampling rate and 32 samples, then according to Eq. (8.45) the nearest Fourier spectral bin is

$$M = \frac{2\text{ kHz}}{\left(16\text{ kHz}/32\right)} = 4$$

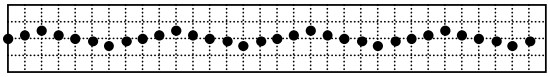
which corresponds exactly with spectral bin 4. The problem with bin 4 is that it is not mutually prime with the number of samples, 32. (Mutually prime numbers are ones containing no common factors.) The number $4 = 2^2$ is not mutually prime with the number $32 = 2^5$, so this choice of bin, sampling frequency, and number of points is a poor one.

One of the problems with non-mutually-prime spectral bins is that they may cause the quantization noise of a coherent signal to contain periodic errors instead of errors that are randomly distributed over the UTP. Consider the sine example with 4 cycles in 32 samples. If we look at a quantized version of this signal from a 4-bit ADC in Figure 8.27, we see that the quantization errors repeat four times in the sample set. The same problem occurs with DAC converters as well. Furthermore, a nonprime spectral bin hits fewer code levels on the DAC and ADC; Thus we are

Figure 8.27. Non-mutually prime spectral bin selection leads to periodic errors.



Non-mutually-prime spectral bin



Periodic quantization errors
appear as gain error and
harmonic distortion

just testing the same points repeatedly. Repetitively exercising the same code levels results in less robust fault coverage in the DAC and ADC circuits.

Another problem with non-mutually-prime bins is that they tend to lead to overlaps between test tones, harmonic distortion components, and intermodulation distortion components. The use of mutually prime bins does not necessarily prevent intermodulation distortion overlaps, but it makes them less likely. Whether mutually prime bins are chosen or not, one should verify that all distortion components fall into spectral bins that do not coincide with bins containing important signal information.

Consider the example of a three-tone multitone at 1, 2, and 3 kHz. The problem with this multitone is that there is a great deal of distortion overlap. The second and third harmonic distortion of the 1-kHz tone falls on top of the 2- and 3-kHz test tones, respectively. Also, the second order intermodulation distortion between the 2-kHz tone and the 3-kHz tone appears at 1 and 4 kHz, corrupting the 1-kHz test tone. All these overlaps would cause errors in any measurement involving the 1-, 2-, or 3-kHz tones (gain, frequency response, distortion, etc.). A better approach is to use test frequencies close to the desired frequencies, but located at spectral bins that do not cause any intermodulation or harmonic distortion overlaps.

EXAMPLE 8.6

Select the spectral bins for a three-tone signal at 1, 2, and 3 kHz with no more than ±50-Hz error in the signal frequencies. The signal should take no more than 50 ms to repeat. Use a 16-kHz sampling rate.

Solution:

With a maximum UTP of 50 ms and a sampling rate of 16 kHz, the number of sample points is found from Eq. (8.44) to be

$$N \leq 50 \text{ ms} \times 16 \text{ kHz} = 800$$

An important constraint on the number of sample points used in most test systems is that ***N*** must be a power of two (i.e., 2P, where ***P*** is an integer). The reason for this will be explained in more detail in Chapter 9, but it is because we will ultimately use the Fast Fourier Transform (FFT) algorithm to measure the response of the DUT to the three-tone signal. Therefore, we shall select ***N*** equal to 512. We want the highest possible ***N*** in order to achieve the greatest frequency resolution or the smallest fundamental frequency. Working with 512 samples, the fundamental frequency becomes

$$F_f = \frac{16\text{ kHz}}{512} = 31.25\text{ Hz}$$

Subsequently, the closest spectral bin numbers that correspond to the 1-, 2-, and 3-kHz signals are found using Eq. (8.45), together with Eq. (8.42), to be

$$M_1 = \frac{1\text{ kHz}}{31.25\text{ Hz}} = 32\text{ (non-mutually-prime, shift to 31)}$$
$$M_2 = \frac{2\text{ kHz}}{31.25\text{ Hz}} = 64\text{ (non-mutually-prime, shift to 63)}$$
$$M_3 = \frac{3\text{ kHz}}{31.25\text{ Hz}} = 96\text{ (non-mutually-prime, shift to 97)}$$

In all three cases, the computed spectral bin values were all even numbers sharing a factor of 2 with the number of samples, 512. Shifting the result by one in either a positive or negative direction eliminates their dependence on the common factor of 2. The resulting test frequencies, f_1, f_2, and f_3, are then

$$f_1 = 31 \times 31.25\text{ Hz} = 968.75\text{ Hz}$$
$$f_2 = 63 \times 31.25\text{ Hz} = 1968.75\text{ Hz}$$
$$f_3 = 97 \times 31.25\text{ Hz} = 3031.25\text{ Hz}$$

In all three cases, the chosen test frequencies are within the desired ±50-Hz error margin and are therefore acceptable.

We now have to verify that there are no distortion overlaps using spectral bins 31, 63, and 97. First we list the harmonics of the three test tone bins (stopping at the Nyquist bin, which is located at 8 kHz, or bin 256). Harmonics are defined as all the frequencies at an integer multiple of the test tone and computed according to the following table:

Harmonic Distortion Terms

Harmonic/Test Tone	M_A	$2M_A$	$3M_A$	$4M_A$	$5M_A$	$6M_A$	$7M_A$	$8M_A$
M_1	31	62	93	124	155	186	217	248
M_2	63	126	189	256	—	—	—	—
M_3	97	194	—	—	—	—	—	—

None of these harmonics overlaps with the other harmonics or with the test tones; Thus the harmonic distortion o verlap criterion is met. Next we look for intermodulation components. Intermodulation components appear at the sum and difference of any two tones, that is, $2F_1 - F_2$,

second- and third-order distortions. (Second-order distortions are those in which the magnitude of the integers in front of F_1 and F_2 add up to 2; third-order distortions are those in which the magnitude of the integers add up to 3; etc.) The following is a series of tables listing the intermodulation interaction between the three test tones:

Second-Order IMD Sum Terms ($M_A + M_B$)

Test Tone Bin	$M_1 = 31$	$M_2 = 63$	$M_3 = 97$
$M_1 = 31$	62	94	128
$M_2 = 63$	94	126	160
$M_3 = 97$	128	160	194

Second-Order IMD Difference Terms ($M_A - M_B$)

Test Tone Bin	$M_1 = 31$	$M_2 = 63$	$M_3 = 97$
$M_1 = 31$	0	32	66
$M_2 = 63$	32	0	34
$M_3 = 97$	66	34	0

Third-Order IMD Sum Terms ($2M_A + M_B$)

Test Tone Bin	$M_1 = 31$	$M_2 = 63$	$M_3 = 97$
$M_1 = 31$	93	125	159
$M_2 = 63$	157	189	223
$M_3 = 97$	225	—	—

Third-Order IMD Difference Terms ($2M_A - M_B$)

Test Tone Bin	$M_1 = 31$	$M_2 = 63$	$M_3 = 97$
$M_1 = 31$	31	1	35
$M_2 = 63$	95	63	29
$M_3 = 97$	163	131	97

Here the letter *A* and *B* represent the row and column spectral bin, respectively. By cycling through all possible combinations of the sums and difference terms, a complete listing of the intermodualtion products can be easily identified. As is evident from this table, none of the intermodulation components falls on top of a test tone located in bins 31, 63, and 97 (except possibly along the diagonal of the matrix corresponding to the third order intermodulation distortion products where these can be ignored). Thus, the chosen spectral bins are good ones.

To avoid overlaps between harmonic distortion components and signal components, we should guarantee not only that the tones are mutually prime with the number of samples, but that they are also mutually prime with one another. For example, bins 3 and 9 are both mutually prime with 512 samples, but they are not mutually prime with one another. Consequently, the third harmonic of spectral bin 3 coincides with the tone at bin 9, resulting in an overlap.

The lack of overlap between harmonic distortion components and test tones in this example is guaranteed by a choice of mutually prime bins. In addition, none of the harmonics interferes with any of the intermodulation components. The choice of mutually prime bins does not guarantee a lack of overlap between intermodulation components and test tones or distortion components, but it does reduce the likelihood of such overlaps. Since there are no overlaps in this example, we can measure gain, frequency response, harmonic distortion, intermodulation distortion, and signal-to-noise ratio with the same set of collected samples. The ability to measure multiple parameters using a single data collection cycle is an advantage of multitone testing. This technique saves a tremendous amount of test time compared with single-tone testing approaches.

As we have seen, multitone DSP-based testing only provides accurate measurements if the test engineer is careful with the selection of test tones. Careless selection of spectral bins will lead to

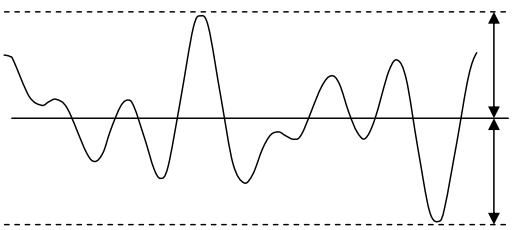
Exercises

8.13.	What is the fundamental frequency of 512 samples collected at a rate of 1 MHz? What is the corresponding UTP?	ANS. 1/512 MHz, 512 μs.
8.14.	How many cycles of a 2.1375-kHz sine wave are completed in a 7.953216-ms UTP?	ANS. 17 cycles.
8.15.	What is the nearest coherent frequency to 20 kHz when 512 samples are collected at a rate of 44 kHz? How many cycles are completed in one UTP?	ANS. 20.023 kHz, 233 cycles.
8.16.	Using a hand analysis, compute the peak-to-RMS ratio of a two-tone multitone described by $A\sin(\omega_1 t) + B\sin(\omega_2 t)$.	ANS. $\sqrt{2}\dfrac{A+B}{\sqrt{A^2+B^2}}$
8.17.	Select the spectral bins of a two-tone signal at 15 and 30 kHz such that minimum distortion overlap occurs. Assume that the sampling rate is 44.8 kHz and that the UTP must be less than 100 ms.	ANS. 1501, 2999.

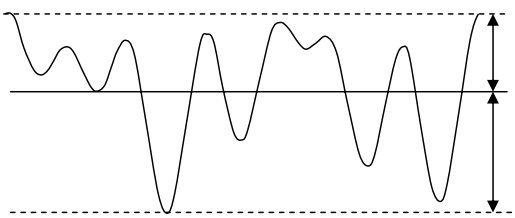
answers that may be slightly incorrect. If we had chosen bin 62 for the 2-kHz tone, for example, then the second harmonic distortion from the 1-kHz tone would have affected the measured level of the 2-kHz tone by a small but significant amount.

In most cases, we choose a sample set consisting of an even number of samples. Thus the mutally prime rule prevents us from using even-numbered spectral bins. In the previous example, we chose bin 63 instead of 62 because it was mutually prime with the number of samples. There is a second reason that we chose 63 and not 62. Combinations of odd

Figure 8.28. Even spectral bins lead to asymmetrical peaks relative to the signal's DC offset.



Spectral bins 3, 5, and 7 (symmetrical peaks)



Spectral bins 2, 5, and 7 (asymmetrical peaks)

harmonics and even harmonics in a multitone signal result in a signal with asymmetrical positive and negative peaks relative to the DC offset of the signal. The DC offset of such an asymmetrical multitone is not centered between the maximum and minimum voltages. This gives poor fault coverage for the circuit under test because it exercises one side of the signal range more than the other.

Figure 8.28 shows the difference between an all-odd multitone and a mixed odd-even multitone. Of course a single-tone signal with an even harmonic does not have the asymmetry problem, but it may lead to the kind of quantization noise modulation illustrated in Figure 8.27. The bottom line is that odd-numbered, mutually prime spectral bins should always be used whenever possible. If the situation is truly desperate, non-mutually-prime or even-numbered bins can be used as a last resort.

8.4 SYNCHRONIZATION OF SAMPLING SYSTEMS

8.4.1 Simultaneous Testing of Multiple Sampling Systems

Many DUTs contain both ADC and DAC channels, as in the case of the voice-band interface of Figure 8.1. These channels are often tested simultaneously in an ATE test program to minimize test time. Simultaneous testing requires a digital pattern loop containing the appropriate samples to excite the DAC channel. At the same time, an AWG converts another set of samples into analog form to excite the ADC channel. The response of the ADC is collected directly into ATE memory for later processing. The DAC response must be digitized before being stored into ATE memory. The response of each channel would then be analyzed through a post-processing frequency-domain operation and judged suitable or not. For example, we might test the gain of the ADC channel and the DAC channel simultaneously using this approach.

Unfortunately, crosstalk between the ADC and DAC channels can lead to small gain errors if we use the same test tones in both channels. Often, the gain errors are small enough that we can live with them. However, if the DAC and ADC have channel-to-channel crosstalk specifications, we can save some test time by measuring the crosstalk during the gain test. All we have to do is select slightly different test tones on the DAC side from those used on the ADC side. Then the feedthrough from DAC to ADC will show up in different bins from the ADC signals and vice versa. This is made possible by operating the various components of the ATE and DUT at different test frequencies but ensuring that they have the same UTP. In turn, this also implies that the fundamental frequency F_f is the same for all components and will guarantee coherent sampling sets in both signal paths. Let us consider the following example.

The preceding example demonstrates one of the reasons we prefer to use the same fundamental frequency for both ADC and DAC. It allows us to make coherent crosstalk measurements between two supposedly isolated signal paths. The other reason is that the UTP for the ADC and DAC is identical by definition, assuming we drive the ADC and DAC from the same digital pattern loop. For instance, if it takes 30 ms to collect the DAC channel samples, then it also takes 30 ms to collect the ADC channel samples. Identical UTPs drive identical fundamental frequencies, since the UTP is the inverse of the fundamental frequency. Sometimes, though, the ADC and DAC are not designed to sample at the same frequency. Fortunately, the sampling frequencies are often related by a simple integer multiple (i.e., 16 and 32 kHz). In these cases, we can simply collect more samples on one channel than on the other to achieve identical fundamental frequencies.

Matching all the fundamental frequencies in a particular test would be easy if we could simply request any arbitrary sampling rate from the ATE instruments. Unfortunately, many ATE testers have a limited choice of sampling frequencies. Henry Ford once said that you could purchase a Model T automobile in any color you wanted, as long as you wanted black. Sometimes particular tester architecture gives us a similar choice of sampling rates. For example, we might

EXAMPLE 8.6

A DUT's DAC and ADC both operate at a 32-kHz sampling rate. Find a sampling system that tests the gain of a DAC channel and an ADC channel simultaneously, as shown in Figure 8.1. Use three tones at 1, 2, and 3 kHz, with a maximum test tone error of 100 Hz.

Solution:
Let us start by setting the number of samples in the waveform exciting the DUT's DAC and ADC at $N = 1024$. Subsequently, the fundamental frequency F_f will be

$$F_f = \frac{32 \text{ kHz}}{1024} = 31.25 \text{ Hz}$$

The desired three tones will then be located in spectral bins of 32, 64, and 96, resulting in the desired test frequencies of 1, 2 and 3 kHz.

Beginning with the ADC channel test, we shall select the sampling rate of the AWG to be 16 kHz and impose the constraint that it has a fundamental frequency of 31.25 Hz. This in turn requires that the sample set consist of 512 samples. Using the sampling rate and spectral bins from the prior example, we will create an AWG waveform with 512 samples, having test tones at bins 31, 63, and 97. We will source this signal from the AWG at a 16-kHz sampling rate. To achieve the same fundamental frequency as the AWG signal, the ADC must collect 1024 samples (32 kHz sampling rate/1024 points = 16 kHz sampling rate/512 points). Using this sampling system we know that the ADC samples will form a coherent sample set.

Next let us consider the DAC channel test. By feeding 1024 samples to the DAC at 32 kHz, we would create a sampling system with the same fundamental frequency as the ADC. If we then set the digitizer sampling rate to 16 kHz and collect 512 samples from the DAC output, we would again achieve a fundamental frequency of 31.25 Hz, guaranteeing coherence. To allow simultaneous testing of the ADC and DAC gain, we need to select different spectral bins than those used to test the ADC. We can choose bins 33, 67, and 95 to meet all our testing criteria.

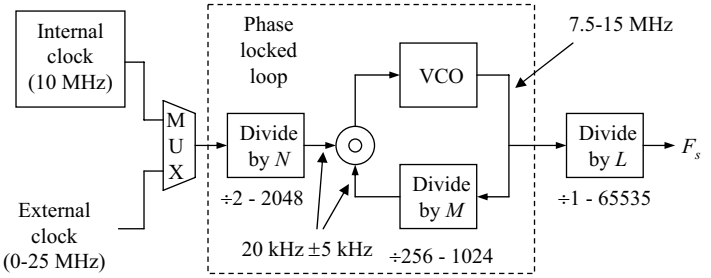
have a choice of any sampling frequency we want as long we want a multiple of 4 Hz. In the remaining sections we shall examine some of the ATE clocking architectures that the test engineer might encounter.

8.4.2 ATE Clock Sources

Mixed-signal testers use a variety of different approaches to clock generation. The most common clock generation schemes involve phase-locked loops, frequency synthesizers, or flying adders. Each of these has strong points and weak points that the test engineer will have to deal with. Ultimately, though, all the clocks in a mixed-signal tester should be referenced to a single master clock so that all instrumentation can be synchronized to achieve coherent sampling systems during each test.

The phase-locked loop (PLL) frequency generator is a circuit that produces an output clock equal to a reference clock times M over N, where M and N are integers. An example ATE PLL-based clocking architecture is shown in Figure 8.29. It consists of several counter stages and a voltage-controlled oscillator (VCO). This PLL is used to generate the sampling clock for a digitizer. It can use either a fixed 10-MHz internal frequency reference or an externally supplied reference frequency.

Figure 8.29. ATE PLL-based digitizer clocking source.



The external reference is required if a DAC output is to be digitized, since a reference clock would have to come from the same digital pattern generator feeding the DAC its samples, frame syncs, and other digital signals. The PLL shown in Figure 8.29 operates by first dividing the reference frequency F_{REF} by N, then by multiplying the result by M through the divide-by-M counter in the negative feedback loop around the VCO. Finally, the frequency of the VCO output can be further divided by another counter stage, which divides the output by integer L resulting in the output sampling frequency F_s given by

$$F_s = \frac{M}{NL} F_{REF} \tag{8.47}$$

This particular example imposes a number of restrictions on the test engineer. First, the externally supplied reference clock must be between 0 Hz and 20 MHz. Next, the value of N must be between 2 and 2048. The output of the divide-by-N stage should be as close to 20 kHz as possible for maximum stability of the PLL. Other frequencies will work, but will introduce additional jitter into the clock. The VCO output must be between 5 and 10 MHz. The value of M must be between 256 and 1024. Finally, the value of L must be between 1 and 65535. Every time the PLL is reconfigured, it must be allowed to settle to a stable state, adding a bit of wait time between tests. Clearly, this clocking architecture is very inflexible and puts a large burden on the test engineer.

More modern testers allow the test engineer to select a wider range of frequencies using a frequency synthesizer. Frequency synthesizers work by taking a reference clock (10 MHz, for example) and passing it through a series of dividers and frequency mixers to produce a very stable output frequency with very little jitter. These synthesizers also take significant time to stabilize (25–50 ms), but that is the price paid for low jitter. Synthesizers are not entirely flexible either. For instance, a particular synthesizer may only be able to produce integer multiples of 4 Hz.

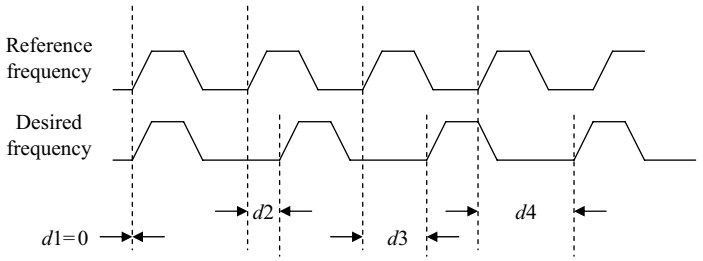
Flying adders can allow an even more flexible clocking source with little settling time, but they may introduce a little more jitter than a frequency synthesizer. A flying adder works by using a

Exercises

8.18. An ATE PLL-based clock source such as the one in Figure 8.29 is set with the divide-by-***N*** counter equal to 1024, the divide-by-***M*** equal to 512, and the divide-by-***L*** equal to 64. With a reference frequency of 25 MHz, what is the output sampling frequency? Are all frequency constraints met in this configuration?

ANS. 195.3125 kHz. Yes, intermediate frequencies are 24.414 kHz and 12.5 MHz.

Figure 8.30. Clock generation using flying adder delays.



high-frequency reference clock and calculating the difference between the desired clock edges and the clock edges produced by the reference clock. Each desired clock edge is generated by delaying each reference clock edge by a carefully calculated amount of time as shown in Figure 8.30.

A new delay time has to be calculated for each delayed clock edge. The calculation is performed on the fly by an adder circuit, thus the terminology "flying adder." Because the edges are generated by programmable delay circuits, flying adders are sometimes more prone to jitter than frequency synthesizers. However, the frequency stabilization time for a flying adder clock circuit is nearly instantaneous.

8.5 SUMMARY

In this chapter, we have presented an introduction to sampling theory and coherent sampling systems as they are applied in mixed-signal ATE testing. While the treatment of this material is not as thorough as one might encounter in a signal processing textbook, this level of coverage should be adequate for beginning mixed-signal test engineers. Thus far, we have only seen how coherent multitone sample sets are created, sourced, and captured. In the next chapter, we will explore the use of digital signal processing algorithms, such as the fast Fourier transform (FFT), in the analysis of the samples collected during a coherent mixed-signal test. As we will see, digital signal processing allows a combination of low test time and high accuracy not possible with conventional, purely analog instrumentation.

PROBLEMS

8.1. For the following parameters of a sampled sine wave (A, ϕ, N, and M), calculate the frequency of the resulting sine wave assuming a sampling rate of 1 Hz. What are the UTP and the fundamental frequency related to this collection of samples?

(a) $A = 2$, $\phi = 0$, $N = 32$, and $M = 1$
(b) $A = 1$, $\phi = \pi/2$, $N = 64$, and $M = 13$
(c) $A = 2$, $\phi = \pi/8$, $N = 64$, and $M = 5$
(d) $A = 1$, $\phi = 0$, $N = 64$, and $M = 33$
(e) $A = 1$, $\phi = 0$, $N = 128$, and $M = 65$
(f) $A = 0.5$, $\phi = 5\pi/8$, $N = 32$, and $M = 2.5$

8.2. Using MATLAB or an equivalent software program, plot the samples of the signals described in Problem 8.1 over its UTP.

8.3. Repeat Problems 8.1 and 8.2 with a sampling rate of 8 kHz.

8.4. Using a square characteristic pulse, reconstruct the samples obtained in Problem 8.1 and determine the frequency of the resulting sine wave. Identify any situation where aliasing occurs.

8.5. Using perfect interpolation reconstruct the samples generated from Problem 8.1. How does the reconstructed waveform compare to a zero-order hold function (square pulse) and a first-order hold function (triangular pulse)?

8.6. If a digital sinusoidal signal described by $A = 1$, $\phi = 0$, $N = 32$, and $M = 5$ is played through a DAC and speaker arrangement whose sampling rate is 8 kHz, what analog frequency will be heard? Repeat for $A = 1$, $\phi = 0$, $N = 32$, and $M = 25$. *Hint:* Reconstruct the discrete signal using the perfect interpolation formula in MATLAB and observe the frequency of the reconstructed signal.

8.7. What is the LSB step size of an ideal 12-bit rounding ADC that has a full-scale input range of 0–3 V? What is the expected RMS value of the corresponding quantization noise?

8.8. If an ideal 8-bit rounding ADC has a 400-μV RMS quantization noise component, what would be the noise component for a 5-bit ADC having the same input range?

8.9. A 6-bit truncating ADC with an analog input range from –1.5 to +1.5 V gives an output of code of 37 for a code range beginning at 0 and ending at 63. What are the minimum and maximum values of the input voltage corresponding to this output code?

8.10. A 6-bit rounding ADC with an analog input range from –1.5 to +1.5 V gives an output of code of 37 for a code range beginning at 0 and ending at 63. What are the minimum and maximum values of the input voltage corresponding to this output code?

8.11. What is the RMS value of the error induced by an ADC having an RMS sampling jitter of 250 ps while measuring a 1-V amplitude sinusoid with a frequency of 20 kHz?

8.12. What is the maximum allowable sampling jitter that a 10-bit ADC can tolerate when it has a full-scale input range of 3 V and converting a 1-V amplitude sinusoid with a frequency of 20 kHz? Assume a 1-LSB step size maximum allowable voltage error.

8.13. What is the maximum allowable sampling jitter that an 8-bit DAC can tolerate when it has a maximum sampling rate of 10 MHz? Assume a 1-LSB step size maximum allowable voltage error.

8.14. If a 6-bit DAC has an RMS sampling jitter of 100 ps, what is the maximum sampling rate? Assume a 1-LSB step size maximum allowable voltage error.

8.15. If an ADC or DAC is controlled by a clock circuit with a minimum clock period of 10 ns and RMS jitter of 250 ps, what is the maximum conversion resolution possible with either the ADC or DAC? Assume a 1-LSB step size maximum allowable voltage error.

8.16. What is the RMS value of the uncertainty associated with a signal sampled by a 10-bit ADC having an RMS clock jitter of 100 ps. The ADC has a full-scale range of 5 V and Nyquist frequency of 1 MHz. *Hint:* Independent errors add in a mean-squared sense, that is, $v^2_{total-RMS} = v^2_{q-RMS} + v^2_{j-RMS}$.

8.17. What is the RMS value of the uncertainty associated with a signal sampled by a 6-bit ADC having an RMS clock jitter of 1 ns. The ADC has a full-scale range of 5 V and Nyquist frequency of 1 MHz.

8.18. Plot the quantization error sequence that results after exciting a 6-bit rounding ADC with a full-scale amplitude sine wave. Use the MATLAB routine given in Example 8.3 for the quantizer. Compute the mean and RMS value of the quantization noise sequence. Repeat for an 8-bit quantizer. How does the mean and RMS value compare with theory in the two cases?

8.19. What is the fundamental frequency of 1024 samples collected at a rate of 20 kHz? What is the corresponding UTP?

8.20. How many cycles of a 20-kHz sine wave are completed in a 0.8-ms UTP? How many cycles of a 20-kHz sine wave are completed in a 0.79-ms UTP? If the signals are repeated indefinitely, which ones are coherent?

8.21. What are the coherent frequencies associated with 16 samples collected at a rate of 1 kHz?

8.22. Using MATLAB or an equivalent software program, plot a multitone signal consisting of three unity amplitude sine waves with frequencies of 1, 5, and 11 kHz over a UTP of 1 ms. Assume that the phase shifts are all zero.

8.23. Using the random selection method described in Section 8.3.3, search for the phases of fifty 1-mV amplitude tones such that the peak-to-RMS value is in a ratio of approximately 3.35:1.

Provide a plot to illustrate your result. Investigate the sensitivity of the peak-to-RMS value by changing the phase of each tone by 1% and computing the change in the peak-to-RMS value.

8.24. The Newman phase selection criterion[7] relies on selecting the phase of the k^{th} tone of a multitone signal according the quadratic expression given by $\phi_k = (\pi/N)(k-1)^2$. Using this equation, determine the phases of the 50-tone multitone signal of Problem 8.23 and compare its peak-to-RMS value to the ideal value of 3.35. Investigate the sensitivity of the peak-to-RMS value by changing the phase of each tone by 1% and computing the change in the peak-to-RMS value.

8.25. Using MATLAB or an equivalent software program, plot a multitone signal consisting of 100 unity-amplitude sine waves distributed across a frequency range of 100 kHz using a UTP of 1 ms. Determine the phases of each tone such that the peak-to-RMS value approaches the ideal value of 3.35.

8.26. Select the spectral bins of a four tone signal at 1, 2, 3, and 4 kHz such that minimum distortion overlap occurs. Assume that the sampling rate is 44.8 kHz and that the UTP must be less than 100 ms. The accuracy of the test frequencies should be less than ±100 Hz. Justify your answer by providing the appropriate distortion tables.

8.27. Create the appropirate distortion tables for the following signalling situations:

(a) $N = 1024$, $M_1 = 15$, $M_2 = 23$, $M_3 = 27$.
(b) $N = 1024$, $M_1 = 15$, $M_2 = 23$, $M_3 = 27$, $M_4 = 33$.
(c) $N = 1024$, $M_1 = 15$, $M_2 = 23$, $M_3 = 27$, $M_4 = 33$, $M_5 = 39$, $M_6 = 49$.

8.28. An ATE PLL-based clock source is set with the divide-by-*N* counter equal to 4096, the divide-by-*M* equal to 512, and the divide-by-*L* equal to 128. With a reference frequency of 200 MHz, what is the output sampling frequency?

8.29. An ATE PLL-based clock source has a divide-by-*N* counter with a range from 1 to 1024, a divide-by-*M* counter with a range from 1 to 256, and a divide-by-*L* with a range of 1 to 65535. With a reference frequency of 200 MHz, what is the range of the output sampling frequency?

REFERENCES

1. M. Mahoney, *Tutorial DSP-Based Testing of Analog and Mixed-Signal Circuits*, The Computer Society of the IEEE, Washington D.C., 1987, ISBN 0818607858.

2. A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1989, ISBN 013216292X.

3. A. V. Oppenheim et al., *Signals and Systems*, Prentice Hall, Englewood Cliffs, NJ, 1997, ISBN 0138147574.

4. W. McC. Siebert, *Circuits, Signals, and Systems*, The MIT Press, Cambridge, MA, 1985, ISBN 0262192292.

5. P. G. Hoel, S. C. Port, and C. J. Stone, *Introduction to Probability Theory*, Houghton Mifflin Company, Boston, 1971, ISBN 039504636X, p. 118.

6. A. J. Jerri, The Shannon sampling theorem—Its various extensions and applications: A tutorial review, *Proceedings of the IEEE*, **65**, pp. 1565–1596, 1997.

7. E. V. Ouderaa, J. Schoukens, and J. Renneboog, Peak factor minimization of the input and output signals of linear systems, *IEEE Transactions on Instrumentation and Measurement*, **37**(2), pp. 207–212, June 1988.