CHAPTER 6

DAC Testing

Data converters (digital-to-analog and analog-to-digital) are used in all aspects of system and circuit design, from audio and video players to cellular telephones to ATE test hardware. When used in conjunction with computers and microprocessors, low-cost mixed-signal systems and circuits have been created that have high noise immunity and an ability to store, retrieve, and transmitted analog information in digital format. Such systems have fueled the growth in the use of the Internet, and this growth continues to push data converter technology to higher operating frequencies and larger bandwidths, along with higher conversion resolution and accuracy.

In this chapter, we will focus on testing the *intrinsic parameters* of a digital-to-analog converter (DAC). The next chapter will look at testing the intrinsic parameters of an analog-to-digital converter (ADC). Intrinsic parameters are those parameters that are intrinsic to the circuit itself and whose parameters are not dependent on the nature of the stimulus. This includes such measurements as absolute error, integral nonlinearity (INL), and differential nonlinearity (DNL). For the most part, intrinsic measurements are related to the DC behavior of the device. In contrast, the AC or transmission parameters, such as gain, gain tracking, signal-to-noise ratio, and signal to harmonic distortion, are strongly dependent on the nature of the stimulus signal. For instance, the amplitude and frequency of the sine wave used in a signal-to-distortion test will often affect the measured result. We defer a discussion of data converter transmission parameters until Chapter 11, after the mathematical details of AC signaling and measurement are described.

When testing a DAC or ADC, it is common to measure both intrinsic parameters and transmission parameters for characterization. However, it is often unnecessary to perform the full suite of transmission tests and intrinsic tests in production. The production testing strategy is often determined by the end use of the DAC or ADC. For example, if a DAC is to be used as a programmable DC voltage reference, then we probably do not care about its signal-to-distortion ratio at 1 kHz. We care more about its worst-case absolute voltage error. On the other hand, if that same DAC is used in a voice-band codec to reconstruct voice signals, then we have a different set of concerns. We do not care as much about the DAC's absolute errors as we care about their end effect on the transmission parameters of the composite audio channel, comprising the DAC, low-pass filter, output buffer amplifiers, and so on.

This example highlights one of the differences between digital testing and specification-oriented mixed-signal testing. Unlike digital circuits, which can be tested based on what they *are* (NAND gate, flip-flop, counter, etc.), mixed-signal circuits are often tested based on what they *do* in the system-level application (precision voltage reference, audio signal reconstruction circuit, video signal generator, etc.). Therefore, a particular analog or mixed-signal subcircuit may be copied from one design to another without change, but it may require a totally different suite of tests depending on its intended functionality in the system-level application.

6.1 BASICS OF DATA CONVERTERS

6.1.1 Principles of DAC and ADC Conversion

Digital-to-analog converters, denoted as DACs, can be considered as decoding devices that accept some input value in the form of an integer number and produces as output a corresponding analog quantity in the form of a voltage, current or other physical quantity. In most engineering applications such analog quantities are conceived as approximation to real numbers. Adapting this view, we can model the DAC decoding process involving a voltage output with an equation of the form

$$v_{OUT} = G_{DAC} D_{IN} \tag{6.1}$$

where D_{IN} is some integer value, v_{OUT} is a real-valued output value, and G_{DAC} is some real-valued proportionality constant. Because the input D_{IN} is typically taken from a digital system, it may come in the form of a D-bit-wide base-2 unsigned integer number expressed as

$$D_{IN} = b_0 + b_1 2^1 + b_2 2^2 + \cdots + b_{D-1} 2^{D-1} \tag{6.2}$$

where the coefficients $b_{D-1}, b_{D-2}, ..., b_2, b_1, b_0$ have either a 0 or 1 value. A commonly used symbol for a DAC is that shown in Figure 6.1. Coefficient b_{D-1} is regarded as the most significant bit (MSB), because it has the largest effect on the number and the coefficient b_0 is known as the least significant bit (LSB), as it has the smallest effect on the number.

For a single LSB change at the input (i.e., $\Delta D_{IN} = 1$ LSB),we see from Eq. (6.1) that the smallest voltage change at the output is $\Delta v_{OUT} = G_{DAC} \times 1$ LSB. Because this quantity is called upon frequently, it is designated as V_{LSB} and is referred to as the *least significant bit step size*. The transfer characteristic of a 4-bit DAC with decoding equation

$$v_{OUT} = \frac{1}{10} D_{IN}, \qquad D_{IN} \in \{0,1,...,15\} \tag{6.3}$$

is shown in Figure 6.2a. Here the DAC output ranging from 0 to 1.5 V is plotted as a function of the digital input. For each input digital word, a single analog voltage level is produced, reflecting the one-to-one input–output mapping of the DAC. Moreover, the LSB step size, V_{LSB}, is equal to 0.1 V.

Figure 6.1. A D-bit digital-to-analog converter.

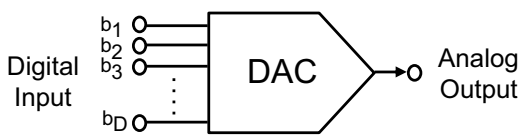
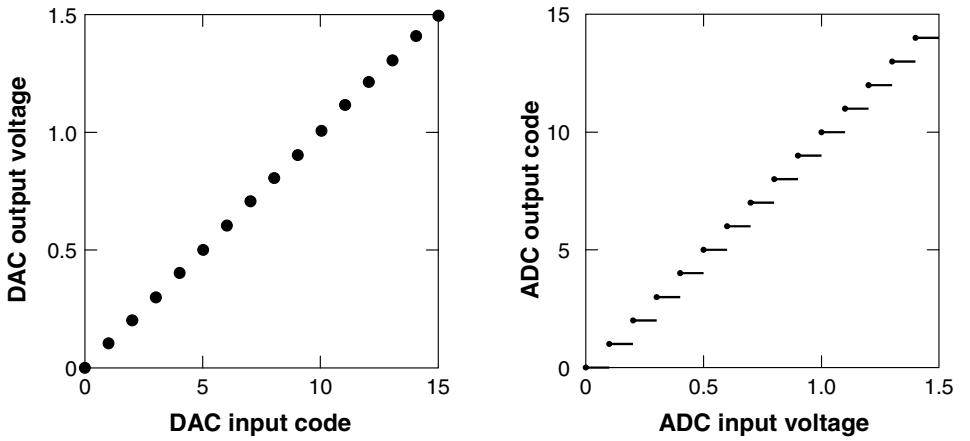


Figure 6.2. (a) DAC code-to-voltage transfer curve (b) ADC voltage-to-code transfer curve.



Alternatively, we can speak about the gain of the DAC (G_{DAC}) as the ratio of the range of output values to the range of input values as follows

$$G_{DAC} = \frac{v_{OUT,\max} - v_{OUT,\min}}{D_{IN,\max} - D_{IN,\min}} \tag{6.4}$$

If we denote the full-scale output range as $V_{FSR} = v_{OUT,\max} - v_{OUT,\min}$ and the input integer range as 2D–1 (for above example, 15–0 = 24–1), then the DAC gain becomes

$$G_{DAC} = \frac{V_{FSR}}{2^D - 1} \tag{6.5}$$

expressed in terms of volts per bit. Consequently, the LSB step size for the ideal DAC in volts can be written as

$$V_{LSB} = \frac{V_{FSR}}{2^D - 1} \tag{6.6}$$

Interesting enough, if the terms $v_{OUT,\min}$ and $v_{OUT,\max}$ covers some arbitrary voltage range corresponding to an arbitrary range of digital inputs, then the DAC input-output behavior can be described in identical terms to the ideal DAC described above except that an offset term is added as follows

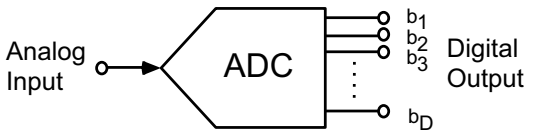
$$v_{OUT} = G_{DAC} D_{IN} + \text{offset} \tag{6.7}$$

Analog-to-digital converters, denoted as ADCs, can be considered as encoding devices that map an input analog level into a digital word of fixed bit length, as captured by the symbol shown in Figure 6.3. Mathematically, we can represent the input–output encoding process in general terms with the equation

$$D_{OUT} = G_{ADC} v_{IN} + \text{offset} \tag{6.8}$$

Returning to our number line analogy, we recognize the ADC process is one that maps an input analog value that is represented on a real number line to a value that lies on an integer number line.

Figure 6.3. A D-bit analog-to-digital converter.



However, not all numbers on a real number line map directly to a value on the integer number line. Herein lies the challenge with the encoding process. One solution to this problem is to divide the analog input full-scale range (V_{FSR}) into 2^D equal-sized intervals according to

$$V_{LSB} = \frac{V_{FSR}}{2^D} \tag{6.9}$$

and assign each interval a code number. Mathematically, we can write this in the form of a set of inequalities as follows:

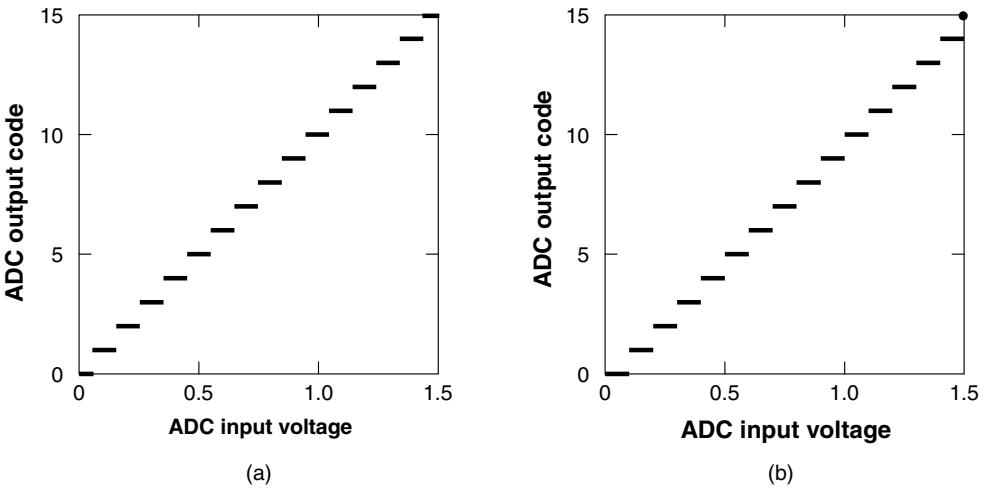
$$D_{OUT} = \begin{cases} 0, & V_{FS-} \le v_{IN} < V_{LSB} \\ 1, & V_{LSB} \le v_{IN} < 2V_{LSB} \\ \vdots & \vdots \\ 2^D - 2, & (2^D - 2)V_{LSB} \le v_{IN} < (2^D - 1)V_{LSB} \\ 2^D - 1, & (2^D - 1)V_{LSB} \le v_{IN} \le V_{FS+} \end{cases} \tag{6.10}$$

where V_{FS-} and V_{FS+} defined the ADC full-scale range of operation, that is, $V_{FSR} = V_{FS+} - V_{FS-}$. The transfer characteristic for a 4-bit ADC is shown in Figure 6.2b for a full-scale input range between 0 and 1.5 V and a LSB step size V_{LSB} of 0.09375 V.

The transfer characteristic of an ADC is not the same across all ADCs, unlike the situation that one finds for DACs. The reason for this comes back to the many-to-one mapping issue described above. Two common approaches used by ADC designers to define ADC operation are based on the mathematical principle of rounding or truncating fractional real numbers. The transfer characteristics of these two types of ADCs are shown in Figure 6.4 for a 4-bit example with $V_{FS-} = 0$ V and $V_{FS+} = 1.5$ V. If the ADC is based on the rounding principle, then the ADC transfer characteristics can be described in general terms as

$$D_{OUT} = \begin{cases} 0, & V_{FS-} \le v_{IN} < \frac{1}{2}V_{LSB} \\ 1, & \frac{1}{2}V_{LSB} \le v_{IN} < \frac{3}{2}V_{LSB} \\ \vdots & \vdots \\ 2^D - 2, & \left[(2^D - 2) - \frac{1}{2}\right]V_{LSB} \le v_{IN} < \left[(2^D - 1) - \frac{1}{2}\right]V_{LSI} \\ 2^D - 1, & \left[(2^D - 1) - \frac{1}{2}\right]V_{LSB} \le v_{IN} \le V_{FS+} \end{cases} \tag{6.11}$$

Figure 6.4. Alternative definitions of the ADC transfer characteristic. (a) ADC operation based on the rounding operation; (b) ADC operation based on the truncation operation.



or for the truncating principle as follows

$$D_{OUT} = \begin{cases} 0, & V_{FS-} \le v_{IN} < V_{LSB} \\ 1, & V_{LSB} \le v_{IN} < 2V_{LSB} \\ \vdots & \vdots \\ 2^D - 2, & \left(2^D - 2\right)V_{LSB} \le v_{IN} < \left(2^D - 1\right)V_{LSB} \\ 2^D - 1, & \left(2^D - 1\right)V_{LSB} = v_{IN} = V_{FS+} \end{cases} \tag{6.12}$$

In both of these two cases, the full-scale range is no longer divided into equal segments. These new definitions lead to a different value for the LSB step size as that described earlier. For these two cases, it is given by

$$V_{LSB} = \frac{V_{FSR}}{2^D - 1} \tag{6.13}$$

Finally, to complete our discussion on ideal ADCs, we like to point out that the proportionality constant G_{ADC} in Eq. (6.8) can be expressed in terms of bits per volt as

$$G_{ADC} = \frac{D_{OUT,\max} - D_{OUT,\min}}{v_{IN,\max} - v_{IN,\min}} = \frac{2^D - 1}{V_{FSR}} \tag{6.14}$$

For both the truncating- and rounding-based ADC, its gain is equal to the reciprocal of the LSB step size as is evident when Eqs. (6.13) and (6.14) are compared.

Exercises

6.1.	A 4-bit DAC has a full-scale voltage range of 0 to 5 V. What is the gain of this DAC?	ANS. $G_{DAC} = 0.33$ V/bit.
6.2.	A 10-bit DAC has a full-scale voltage range of −5.0 to 5.0 V. What is the LSB step size?	ANS. $V_{LSB} =$ 9.77 mV.
6.3.	A 3-bit ADC has a full-scale voltage range of 0 to 5.0 V. What is the gain of this ADC if its internal operation is based on rounding?	ANS. $G_{DAC} = 1.4$ *bits/V*.
6.4.	A 10-bit ADC has a full-scale voltage range of −5.0 to 5.0 V. What is the LSB step size if its internal operation is based on truncation?	ANS. $V_{LSB} =$ 9.77 mV.

6.1.2 Data Formats

There are several different encoding formats for ADCs and DACs including unsigned binary, sign/magnitude, two's complement, one's complement, mu-law, and A-law. One common omission in device data sheets is DAC or ADC data format. The test engineer should always make sure the data format has been clearly defined in the data sheet before writing test code.

The most straightforward data format is unsigned binary written as $b_{D-1}b_{D-2}...b_2b_1b_0$ whose equivalent base-10 integer value is found from

$$D_{IN} = b_{N-1}2^{N-1} + b_{N-2}2^{N-2} + \quad + b_1 2^1 + b_0 2^0 \tag{6.15}$$

Unsigned binary format places the lowest voltage at code 0 and the highest voltage at the code with all 1's. For example, an 8-bit DAC with a full-scale voltage range of 1.0 to 3.0 V would have the code-to-voltage relationship shown in Table 6.1.

One LSB step size is equal to the full-scale voltage range, $V_{FS+} - V_{FS-}$, divided by the number of DAC codes (e.g., 2D) minus one

$$V_{LSB} = \frac{V_{FS+} - V_{FS-}}{\#\text{ DAC codes } - 1} \tag{6.16}$$

Table 6.1. Unsigned Binary Format for an 8-Bit DAC

Code	Voltage
00000000 (integer 0)	1.0 V
00000001 (integer 1)	1.0 V + 1 VLSB = 1.007843 V
...	
01111111 (integer 127)	1.0 V + 127 VLSBs = 1.996078 V
10000000 (integer 128)	1.0 V + 128 VLSBs = 2.003922 V
...	
11111111 (integer 255)	1.0 V + 255 VLSBs = 3.0 V

In this example, the voltage corresponding to one LSB step size is equal to (3.0 V – 1.0 V)/255 = 7.843 mV. Sometimes the full-scale voltage is defined with one an additional imaginary code above the maximum code (i.e., code 256 in our 8-bit example). If so, then the LSB size would be (3.0 V – 1.0 V)/256 = 7.8125 mV. This source of ambiguity should be clarified in the data sheet.

Another common data format is two's complement; written exactly the same as an unsigned binary number, for example, $b_{D-1}b_{D-2}\cdots b_2b_1b_0$. A two's complement binary representation is converted to its equivalent base-10 integer value using the equation

$$D_{IN} = -b_{D-1}2^{D-1} + b_{D-2}2^{D-2} + b_{D-3}2^{D-3} + \cdots + b_1 2^1 + b_0 2^0 \tag{6.17}$$

A two's complement binary formatted number can be used to express both positive and negative integer values. Positive numbers are encoded the same as an unsigned binary in two's complement, except that the most significant bit must always be zero. When the most significant bit is one, the number is negative. To multiply a two's complement number by –1, all bits are inverted and one is added to the result. The two's complement encoding scheme for an 8-bit DAC is shown in Table 6.2. As is evident from the table, all outputs are made relative to the DAC's midscale value of 2.0 V. This level corresponds to input digital code 0. Also evident from this table is the LSB is equal to 5 mV. The midscale (MS) value is computed from either of the following two expressions using knowledge of the lower and upper limits of the DAC's full-scale range, denoted V_{FS-} and V_{FS+}, respectively, together with the LSB step size obtained from Eq. (6.16), as follows:

$$V_{MS} = V_{FS+} - \left(\frac{\#\text{ DAC codes}}{2} - 1\right)V_{LSB} \tag{6.19}$$

Note that the two's complement encoding scheme is slightly asymmetrical since there are more negative codes than positive ones.

One's complement format is similar to two's complement, except that it eliminates the asymmetry by defining 11111111 as minus zero instead of minus one, thereby making 11111111 a redundant code. One's complement format is not commonly used in data converters because it is not quite as compatible with mathematical computations as two's complement or unsigned binary formats.

Table 6.2. Two's Complement Format for an 8-Bit DAC

Code	Voltage
10000000 (integer –128)	2.0 V – 128 VLSBs = 1.360 V
10000001 (integer –127)	2.0 V – 127 VLSBs = 1.365 V
...	
11111111 (integer –1)	2.0 V – 1 VLSB = 1.995 V
00000000 (integer 0)	2.0 V (midscale voltage)
00000001 (integer –1)	2.0 V + 1 VLSB = 2.005 V
...	
01111110 (integer 126)	2.0 V + 126 VLSBs = 2.630 V
01111111 (integer 127)	2.0 V + 127 VLSBs = 2.635 V

Sign/magnitude format is occasionally used in data converters. In sign/magnitude format, the most significant bit is zero for positive values and one for negative values. A sign/magnitude formatted binary number expressed in the form $b_{D-1}b_{D-2}\cdots b_2b_1b_0$ has the following base-10 equivalent integer value:

$$D_{IN} = (-1)^{b_{N-1}} \times \left(b_{N-2}2^{N-2} + b_{N-3}2^{N-3} + \cdots + b_1 2^1 + b_0 2^0\right) \tag{6.20}$$

Like one's complement, sign/magnitude format also has a redundant negative zero value. Table 6.3 shows sign/magnitude format for the 8-bit DAC example. The midscale level corresponding to input code 0 for this type of converter is

$$V_{MS} = V_{FS-} + \left(\frac{\#\text{ DAC codes}}{2} - 1\right)V_{LSB} = V_{FS+} - \left(\frac{\#\text{ DAC codes}}{2} - 1\right)V_{LSB} \tag{6.21}$$

where V_{LSB} is given by

$$V_{LSB} = \frac{V_{FS+} - V_{FS-}}{\#\text{ DAC codes} - 2} \tag{6.22}$$

Two other data formats, mu-law and A-law, were developed in the early days of digital telephone equipment. Mu-law is used in North American and related telephone systems, while A-law is used in European telephone systems. Today the mu-law and A-law data formats are sometimes found not only in telecommunications equipment but also in digital audio applications, such as PC sound cards. These two data formats are examples of companded encoding schemes.

Companding is the process of compressing and expanding a signal as it is digitized and reconstructed. The idea behind companding is to digitize or reconstruct large amplitude signals with coarse converter resolution while digitizing or reconstructing small amplitude signals with finer resolution. The companding process results in a signal with a fairly constant signal to quantization noise ratio, regardless of the signal strength.

Compared with a traditional linear converter having the same number of bits, a companding converter has worse signal-to-noise ratio when signal levels are near full scale, but better signal-to-noise ratios when signal levels are small. This tradeoff is desirable for telephone conversations, since it limits the number of bits required for transmission of digitized voice. Companding is therefore a simple form of lossy data compression.

Table 6.3. Sign/Magnitude Format for an 8-Bit DAC

Code	Voltage
11111111 (integer –127)	2.0 V–127 VLSBs = 1.365 V
11111110 (integer –126)	2.0 V–126 VLSBs = 1.370 V
...	
10000001 (integer –1)	2.0 V–1 VLSB = 1.995 V
10000000 (integer –0)	2.0 V
00000000 (integer 0)	2.0 V
00000001 (integer 1)	2.0 V + 1 VLSB = 2.005 V
...	
01111110 (integer 126)	2.0 V + 126 VLSBs = 2.630 V
01111111 (integer 127)	2.0 V + 127 VLSBs = 2.635 V

Figure 6.5. Comparison of linear and companding 4-bit ADC-to-DAC transfer curves.

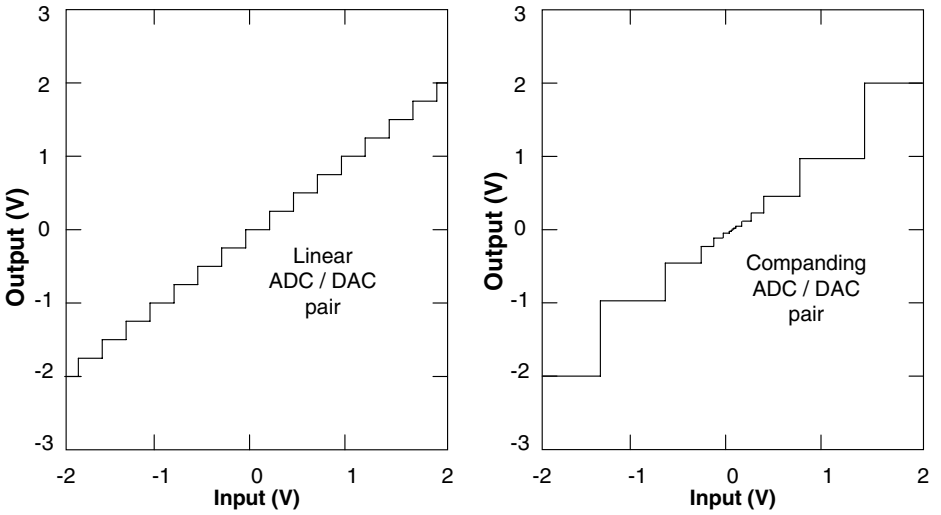


Figure 6.5 shows the transfer curve of a simple 4-bit companded ADC followed by a 4-bit DAC. In a true logarithmic companding process such as the one in Figure 6.5, the analog signal is passed through a linear-to-logarithmic conversion before it is digitized. The logarithmic process compresses the signal so that small signals and large signals appear closer in magnitude. Then the compressed signal may be digitized and reconstructed using an ADC and DAC. The reconstructed signal is then passed through a logarithmic-to-linear conversion to recover a companded version of the original signal.

Exercises

6.5. A 4-bit DAC has a full-scale voltage range of 0 to 5 V. The input is formatted using an unsigned binary number representation. List all possible ideal output levels. What output level corresponds to the DAC input code 0? What is the VLSB?

ANS. Code 0 to 15: 0, 0.333, 0.666, 0.999, 1.33, 1.66, 2.00, 2.33, 2.66, 3.00, 3.33, 3.66, 4.00, 4.33, 4.66, 5.00 V; Code 0 = 0 V; VLSB: = 0.333 V.

6.6. A 4-bit ADC has a full-scale voltage range of –5 to 5 V. The internal operation of the ADC is based on truncation and the input digital signal is formatted using a two's complement binary number representation. List all possible ideal output levels. What is the LSB step size?

ANS. Code –8 to 7: –5, –4.33, –3.67, –3.00, –2.33, –1.66, –1.00, –0.33, .34, 1.00, 1.67, 2.34, 3.00, 3.67, 4.34, 5.0 V; Code 0 = 0.34 V; VLSB: = 0.666 V.

6.7. What is the ideal gain of a 10-bit DAC with a full-scale voltage range of 0 to 5 V?

ANS. 4.888 mV/bit.

6.8. A 7-bit DAC has a full-scale voltage range of 0.5 to 2.5 V. The input is formatted using a 2's complement number representation. What is the midscale voltage level? What is the expected output voltage level if the input digital code is 1001101?

ANS. 1.5079 V, 0.7047 V.

The mu-law and A-law encoding and decoding rules are a sign/magnitude format with a piecewise linear approximation of a true logarithmic encoding scheme. They define a varying LSB size that is small near 0 and larger as the voltage approaches plus or minus full scale. Each of the piecewise linear sections is called a *chord*. The steps in each chord are of a constant size. The piecewise approximation was much easier to implement in the early days of digital telecommunications than a true logarithmic companding scheme, since the piecewise linear sections could be implemented with traditional binary weighted ADCs and DACs. Today, the A-law and mu-law encoding and decoding process is often performed using lookup tables combined with linear sigma-delta ADCs and DACs having at least 13 bits of resolution. A more complete discussion of A-law and mu-law codec testing can be found in Matthew Mahoney's book, *DSP-based Testing of Analog and Mixed-Signal Circuits*.[1]

6.1.3 Comparison of DACs and ADCs

Although this chapter is devoted to DAC testing, many of the concepts presented are closely tied to ADC testing. For instance, the code-to-voltage transfer characteristics for a DAC are similar to the voltage-to-code characteristics of an ADC. However, it is very important to note that a DAC represents a one-to-one mapping function whereas an ADC represents a many-to-one mapping. This distinction is illustrated in Figure 6.2(a) and (b). For each digital input code, a DAC produces only one output voltage.

An ADC, by contrast, produces the same output code for many different input voltages. In fact, because an ADC's circuits generate random noise and because any input signal will include a certain amount of noise, the ADC decision levels represent *probable* locations of transitions from one code to the next. We will discuss the probabilistic nature of ADC decision levels and their effect on ADC testing in Chapter 7. While DACs also generate random noise, this noise can be removed through averaging to produce a single, unambiguous voltage level for each DAC code. Therefore, the DAC transfer characteristic is truly a one-to-one mapping of codes to voltages.

The difference between DAC and ADC transfer characteristics prevents us from using complementary testing techniques on DACs and ADCs. For example, a DAC is often tested by measuring the output voltage corresponding to each digital input code. The test engineer might be tempted to test an ADC using the complementary approach, applying the ideal voltage levels for each code and then comparing the actual output code against the expected code. Unfortunately, this approach is completely inappropriate in most ADC testing cases, since it does not characterize the location of each ADC decision level. Furthermore, this crude testing approach will often fail perfectly good ADCs simply because of gain and offset variations that are within acceptable limits.

Although there are many differences in the testing of DACs and ADCs, there are enough similarities that we have to treat the two topics as one. In Chapter 7 we will see how ADC testing is similar to DAC testing and also how it differs. In this chapter, however, we will concentrate mainly on DAC testing. Also, we will concentrate mostly on voltage output DACs. Current output DACs are tested using the same techniques, using either a current mode DC voltmeter or a calibrated current-to-voltage translation circuit on the device interface board (DIB).

6.1.4 DAC Failure Mechanisms

The novice test engineer may be inclined to think that all *N*-bit DACs are created equal and are therefore tested using the same techniques. As we will see, this is not the case. There are many different types of DACs, including binary-weighted architectures, resistive divider architectures, pulse-width-modulated (PWM) architectures, and pulse-density-modulated (PDM) architectures (commonly known as *sigma-delta* DACs). Furthermore, there are hybrids of these architectures, such as the multibit sigma-delta DAC and segmented resistive divider DACs. Each of these DAC architectures has a unique set of strengths and weaknesses. Each architecture's weaknesses determines its likely failure mechanisms, and these in turn drive the testing methodology. As previously noted, the requirements of the DAC's system-level application also determine the testing methodology.

Before we discuss testing methodologies for each type of DAC, we first need to outline the DC and dynamic tests commonly performed on DACs. The DC tests include the usual specifications like gain, offset, power supply sensitivity, and so on. They also include converter-specific tests such as absolute error, monotonicity, integral nonlinearity (INL), and differential nonlinearity (DNL), which measure the overall quality of the DAC's code-to-voltage transfer curve. The dynamic tests are not always performed on DACs, especially those whose purpose is to provide DC or low-frequency signals. However, dynamic tests are common in applications such as video DACs, where fast settling times and other high-frequency characteristics are key specifications.

6.2 BASIC DC TESTS

6.2.1 Code-Specific Parameters

DAC specifications sometimes call for specific voltage levels corresponding to specific digital codes. For instance, an 8-bit two's complement DAC may specify a voltage level of 1.360 V ± 10 mV at digital code –128 and a voltage level of 2.635 V ± 10 mV at digital code +127. (See Section 6.1.2 for a description of converter data formats such as unsigned binary and two's complement.) Alternatively, DAC code errors can be specified as a percentage of the DAC's full-scale range rather than an absolute error. In this case, the DAC's full-scale range must first be measured to determine the appropriate test limits. Common code-specific parameters include the maximum full-scale (V_{FS+}) voltage, minimum full-scale (V_{FS-}) voltage, and midscale (V_{MS}) voltage. The midscale voltage typically corresponds to 0 V in bipolar DACs or a center voltage such as $V_{DD}/2$ in unipolar (single power supply) DACs. It is important to note that although the minimum full-scale voltage is often designated with the V_{FS-} notation, it is not necessarily a negative voltage.

6.2.2 Full-Scale Range

Full-scale range (V_{FSR}) is defined as the voltage difference between the maximum voltage and minimum voltage that can be produced by a DAC. This is typically measured by simply measuring the DAC's positive full-scale voltage, V_{FS+}, and then measuring the DAC's negative full-scale voltage, V_{FS-}, and subtracting

$$V_{FSR} = V_{FS+} - V_{FS-} \tag{6.23}$$

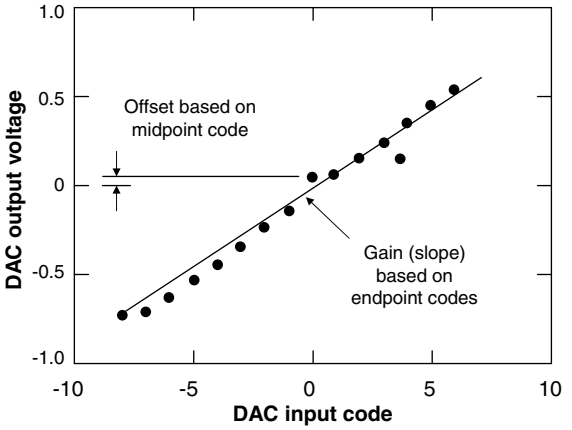
6.2.3 DC Gain, Gain Error, Offset, and Offset Error

It is tempting to say that the DAC's offset is equal to the measured midscale voltage, V_{MS}. It is also tempting to define the gain of a DAC as the full-scale range divided by the number of spaces, or steps, between codes. These definitions of offset and gain are approximately correct, and in fact they are sometimes found in data sheets specified exactly this way. They are quite valid in a perfectly linear DAC. However, in an imperfect DAC, these definitions are inferior because they are very sensitive to variations in the V_{FS-}, V_{MS}, and V_{FS+} voltage outputs while being completely insensitive to variations in all other voltage outputs.

Figure 6.6 shows a simulated DAC transfer curve for a rather bad 4-bit DAC. Notice that code 0 does not produce 0 V, as it should. However, the overall curve has an offset near 0 V. Also, notice that the gain, if defined as the full-scale range divided by the number of spaces between codes, does not match the general slope of the curve. The problem is that the V_{FS+}, V_{FS-}, and V_{MS} voltages are not in line with the general shape of the transfer curve.

A less ambiguous definition of gain and offset can be found by computing the best-fit line for these points and then computing the gain and offset of this line. For high-resolution DACs with

Figure 6.6. Endpoint/midpoint-referenced gain and offset for a 4-bit DAC.



reasonable linearity, the errors between these two techniques become very small. Nevertheless, the best-fit line approach is independent of DAC resolution; thus it is the preferred technique.

A best-fit line is commonly defined as the line having the minimum squared errors between its ideal, evenly spaced samples and the actual DAC output samples. For a sample set $S(i)$, where i ranges from 0 to $N-1$ and N is the number of samples in the sample set, the best-fit line is defined by its slope (DAC gain) and offset using a standard linear equation having the form

$$Best\_fit\_line = \text{gain} \times i + \text{offset} \quad \text{for } i = 0,1,\ldots,N-1 \tag{6.24}$$

The equations for slope and offset can be derived using various techniques. One technique minimizes the partial derivatives with respect to slope and offset of the squared errors between the sample set S and the best-fit line. Another technique is based on linear regression.[2] The equations derived from the partial derivative technique are

$$\text{gain} = \frac{N\,K_4 - K_1\,K_2}{N\,K_3 - K_1^2}, \qquad \text{offset} = \frac{K_2}{N} - \text{gain}\,\frac{K_1}{N} \tag{6.25}$$

where

$$K_1 = \sum_{i=0}^{N-1} i, \quad K_2 = \sum_{i=0}^{N-1} S(i), \quad K_3 = \sum_{i=0}^{N-1} i^2, \quad K_4 = \sum_{i=0}^{N-1} i\,S(i)$$

The derivation details are left as an exercise in the problem set found at the end of this chapter. These equations translate very easily into a computer program.

The values in the array *Best_fit_line* represent samples falling on the least-squared-error line. The program variable *Gain* represents the gain of the DAC, in volts per bit. This gain value is the average gain across all DAC samples. Unlike the gain calculated from the full-scale range divided by the number of code transitions, the slope of the best-fit line represents the true gain of the DAC. It is based on all samples in the DAC transfer curve and therefore is not especially sensitive to any one code's location. Gain error, ΔG, expressed as a percent, is defined as

$$\Delta G = \left(\frac{G_{ACTUAL}}{G_{IDEAL}} - 1 \right) \times 100\% \tag{6.26}$$

Likewise, the best-fit line's calculated offset is not dependent on a single code as it is in the midscale code method. Instead, the best-fit line offset represents the offset of the total sample set. The DAC's offset is defined as the voltage at which the best-fit line crosses the *y* axis. The DAC's offset error is equal to its offset minus the ideal voltage at this point in the DAC transfer curve. The *y* axis corresponds to DAC code 0.

In unsigned binary DACs, this voltage corresponds to *Best_fit_line*(*1*) in the MATLAB routine. However, in two's complement DACs, the value of *Best_fit_line*(*1*) corresponds to the DAC's V_{FS-} voltage, and therefore does not correspond to DAC code 0. In an 8-bit two's complement DAC, for example, the 0 code point is located at $i = 128$. Therefore, the value of the program variable *Offset* does not correspond to the DAC's offset. This discrepancy arises simply because we cannot use negative index values in MATLAB code arrays such as *Best_fit_line*(*–128*). Therefore, to find the DAC's offset, one must determine which sample in vector *Best_fit_line* corresponds to the DAC's 0 code. The value at this array location is equal to the DAC's offset. The ideal voltage at the DAC 0 code can be subtracted from this value to calculate the DAC's offset error.

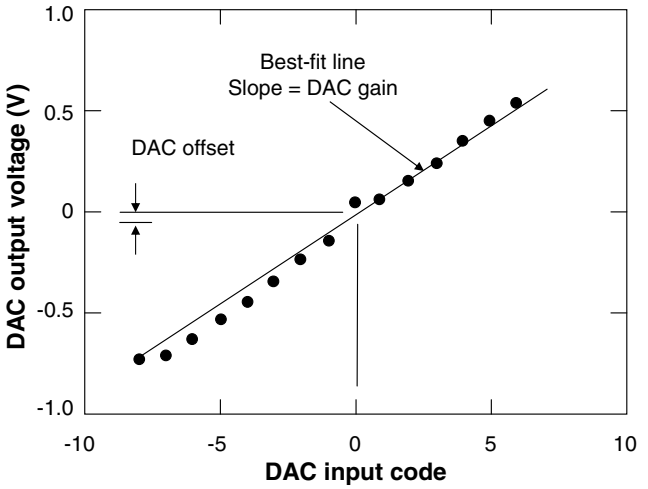
EXAMPLE 6.1

A 4-bit two's complement DAC produces the following set of voltage levels, starting from code –8 and progressing through code +7:

−780 mV, −705 mV, −530 mV, −455 mV, −400 mV, −325 mV, −150 mV, −75 mV,
120 mV, 195 mV, 370 mV, 445 mV, 500 mV, 575 mV, 750 mV, 825 mV

These code levels are shown in Figure 6.7. The ideal DAC output at code 0 is 0 V. The ideal gain is equal to 100 mV/bit. Calculate the DAC's gain (volts per bit), gain error, offset, and offset error using a best-fit line as reference.

Figure 6.7. A 4-bit DAC transfer curve and best-fit line.



Solution:

We calculate gain and offset using the previous MATLAB routine, resulting in a gain value of 109.35 mV/bit and an offset value of –797.64 mV. The gain error is found from Eq. (6.26) to be

$$\Delta G = \left(\frac{109.35\ \text{mV}}{100\ \text{mV}} - 1\right) \times 100\% = 9.35\%$$

Because this DAC uses a two's complement encoding scheme, this offset value is the offset of the best-fit line, not the offset of the DAC at code –8.

The DAC's offset is found by calculating the best-fit line's value at DAC code 0, which corresponds to i = 8

$$\begin{aligned}\text{DAC offset} &= \text{gain} \times 8 + \text{offset}\\ &= 109.35\ \text{mV} \times 8 - 797.64\ \text{mV}\\ &= 77.16\ \text{mV}\end{aligned}$$

$$\begin{aligned}\text{DAC offset error} &= \text{DAC offset} - \text{ideal offset}\\ &= 77.16\ \text{mV} - 0\ \text{V} = 77.16\ \text{mV}\end{aligned}$$

Clearly, when the ideal offset is 0 V, the DAC offset and offset error are identical. Many DACs have an ideal offset of $V_{DD}/2$ or some other nonzero value. These DACs are commonly used in applications requiring a single power supply. In such a case, the offset should be nonzero, but the offset error should always be zero.

6.2.4 LSB Step Size

The least significant bit (LSB) step size is defined as the average step size of the DAC transfer curve. It is equal to the gain of the DAC, in volts per bit. Although it is possible to measure the approximate LSB size by simply dividing the full-scale range by the number of code transitions, it is more accurate to measure the gain of the best-fit line to calculate the average LSB size. Using the results from the previous example, the 4-bit DAC's LSB step size is equal to 109.35 mV.

6.2.5 DC PSS

DAC DC power supply sensitivity (PSS) is easily measured by applying a fixed code to the DAC's input and measuring the DC gain from one of its power supply pins to its output. PSS for a DAC is therefore identical to the measurement of PSS in any other circuit, as described in Section 3.8.1. The only difference is that a DAC may have different PSS performance depending on the applied digital code. Usually, a DAC will exhibit the worst PSS performance at its full-scale and/or minus full-scale settings because these settings tie the DAC output directly to a voltage derived from the power supply. Worst-case conditions should be used once they have been determined through characterization of the DAC.

Exercises

6.9. A 4-bit unsigned binary DAC produces the following set of voltage levels, starting from code 0 and progressing through to code 15:

1.0091, 1.2030, 1.3363, 1.5617, 1.6925, 1.9453, 2.0871, 2.3206, 2.4522, 2.6529, 2.8491, 2.9965, 3.1453, 3.3357, 3.4834, 3.6218

The ideal DAC output at code 0 is 1 V and the ideal gain is equal to 200 mV/bit. The data sheet for this DAC specifies offset and offset using a best-fit line, evaluated at code 0. Gain is also specified using a best-fit line. Calculate the DAC's gain (volts per bit), gain error, offset, and offset error.

ANS. G = 177.3 mV/bit; ΔG = −11.3%; offset = 1.026 V; offset error = 26.1 mV.

6.10. Estimate the LSB step size of the DAC described in Exercise 6.7 using its measured full-scale range (i.e. using the endpoint method). What are the gain error and offset error?

ANS. LSB = 174.2 mV; ΔG = −12.9%; offset error = 9.1 mV.

6.3 TRANSFER CURVE TESTS

6.3.1 Absolute Error

The ideal DAC transfer characteristic or transfer curve is one in which the step size between each output voltage and the next is exactly equal to the desired LSB step size. Also, the offset error of the transfer curve should be zero. Of course, physical DACs do not behave in an ideal manner; so we have to define figures of merit for their actual transfer curves.

One of the simplest, least ambiguous figures of merit is the DAC's maximum and minimum absolute error. An absolute error curve is calculated by subtracting the ideal DAC output curve from the actual measured DAC curve. The values on the absolute error curve can be converted to LSBs by dividing each voltage by the ideal LSB size, V_{LSB}. The conversion from volts to LSBs is a process called *normalization*.

Mathematically, if we denote the ith value on the ideal and actual transfer curves as $S_{IDEAL}(i)$ and $S(i)$, respectively, then we can write the normalized absolute error transfer curve $\Delta S(i)$ as

$$\Delta S(i) = \frac{S(i) - S_{IDEAL}(i)}{V_{LSB}} \tag{6.27}$$

EXAMPLE 6.2

Assuming an ideal gain of 100 mV per LSB and an ideal offset of 0 V at code 0, calculate the absolute error curve for the 4-bit DAC of the previous example. Express the results in terms of LSBs.

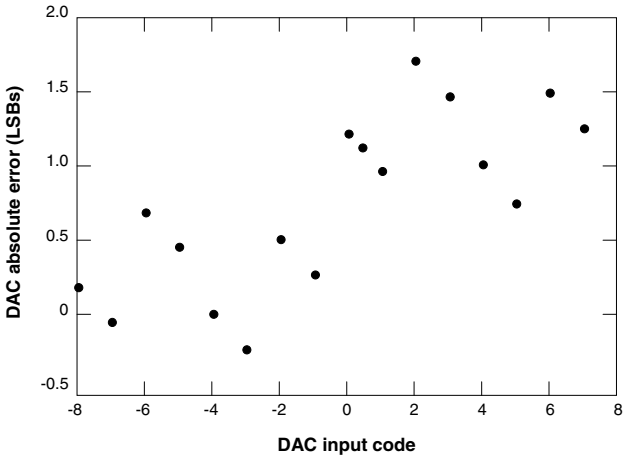
Solution:

The ideal DAC levels are –800, –700, . . . , +700 mV. Subtracting these ideal values from the actual values, we can calculate the absolute voltage errors $\Delta S(i)$ as:

+20 mV, 25 mV, +70 mV, +45 mV, 0 mV, 225 mV, +50 mV, +25 mV, +120 mV,
+95 mV, +170 mV, +145 mV, +100 mV, +75 mV, +150 mV, +125 mV

The maximum absolute error is +170 mV and the minimum absolute error is –25 mV. Dividing each value by the ideal LSB size (100 mV), we get the normalized error curve shown in Figure 6.8. This curve shows that this DAC's maximum and minimum absolute errors are +1.7 and –0.25 LSBs, respectively. In a simple 4-bit DAC, this would be considered very bad performance, but this is an imaginary DAC designed for instructional purposes. In high-resolution DACs, on the other hand, absolute errors of several LSBs are common. The larger normalized absolute error in high-resolution DACs is a result of the smaller LSB size. Therefore, absolute error testing is often replaced by gain, offset, and linearity testing in high-resolution DACs.

Figure 6.8. Normalized DAC absolute error curve.



6.3.2 Monotonicity

A monotonic DAC is one in which each voltage in the transfer curve is larger than the previous voltage, assuming a rising voltage ramp for increasing codes. (If the voltage ramp is expected to decrease with increasing code values, we simply have to make sure that each voltage is less than the previous one.) While the 4-bit DAC in the previous examples has a terrible set of absolute errors, it is nevertheless monotonic. Monotonicity testing requires that we take the discrete first derivative of the transfer curve, denoted here as $S'(i)$, according to

$$S'(i) = S(i+1) - S(i) \tag{6.28}$$

If the derivatives are all positive for a rising ramp input or negative for a falling ramp input, then the DAC is said to be monotonic.

EXAMPLE 6.3

Verify monotonicity in the previous DAC example.

Solution:

The first derivative of the DAC transfer curve is calculated, yielding the following values

75 mV, 175 mV, 75 mV, 55 mV, 75 mV, 175 mV, 75 mV, 195 mV, 75 mV,
175mV, 75 mV, 55 mV, 75 mV, 175 mV, 75 mV

Notice that there are only 15 first derivative values, even though there are 16 codes in a 4-bit DAC. This is the nature of the discrete derivative, since there are one fewer *changes* in voltage than there are voltages. Since each value in this example has the same sign (positive), the DAC is monotonic.

6.3.3 Differential Nonlinearity

Notice that in the monotonicity example the step sizes are not uniform. In a perfect DAC, each step would be exactly 100 mV corresponding to the ideal LSB step size. Differential nonlinearity (DNL) is a figure of merit that describes the uniformity of the LSB step sizes between DAC codes. DNL is also known as *differential linearity error* or DLE for short. The DNL curve represents the error in each step size, expressed in fractions of an LSB. DNL is computed by calculating the discrete first derivative of the DACs transfer curve, subtracting one LSB (i.e., V_{LSB}) from the derivative result, and then normalizing the result to one LSB

$$\mathrm{DNL}(i) = \frac{S(i+1) - S(i) - V_{LSB}}{V_{LSB}} \ \mathrm{LSB} \tag{6.29}$$

As previously mentioned, we can define the average LSB size in one of three ways. We can define it as the actual full-scale range divided by the number of code transitions (number of codes minus 1) or we can define the LSB as the slope of the best-fit line. Alternatively, we can define the LSB size as the ideal DAC step size.

Exercises

6.11. Assuming an ideal gain of 200 mV/bit and an ideal offset of 1 V at code 0, calculate the absolute error transfer curve for the 4-bit DAC of Exercise 6.7. Normalize the result to a single LSB step size.

ANS. 0.0455, 0.0150, -0.3185, −0.1915, −0.5375, −0.2735, −0.5645, −0.3970, −0.7390 −0.7355 −0.7545 −1.0175 −1.2735 −1.3215 −1.5830 −1.8910

6.12. Compute the discrete first derivative of the DAC transfer curve given in Exercise 6.7. Is the DAC output monotonic?

ANS. 0.1939, 0.1333, 0.2254, 0.1308, 0.2528, 0.1418, 0.2335, 0.1316, 0.2007, 0.1962, 0.1474, 0.1488, 0.1904, 0.1477, 0.1384
The DAC is monotonic since there are no negative values in the discrete derivative.

The choice of LSB calculations depends on what type of DNL calculation we want to perform. There are four basic types of DNL calculation method: best-fit, endpoint, absolute, and best-straight-line. Best-fit DNL uses the best-fit line's slope to calculate the average LSB size. This is probably the best technique, since it accommodates gain errors in the DAC without relying on the values of a few individual voltages. Endpoint DNL is calculated by dividing the full-scale range by the number of transitions. This technique depends on the actual values for the maximum full-scale (V_{FS+}) and minimum full-scale (V_{FS-}) levels. As such it is highly sensitive to errors in these two values and is therefore less ideal than the best-fit technique. The absolute DNL technique uses the ideal LSB size derived from the ideal maximum and minimum full-scale values. This technique is less commonly used, since it assumes the DAC's gain is ideal.

The best-straight-line method is similar to the best-fit line method. The difference is that the best-straight-line method is based on the line that gives the best answer for integral nonlinearity (INL) rather than the line that gives the least squared errors. Integral nonlinearity will be discussed later in this chapter. Since the best-straight-line method is designed to yield the best possible answer, it is the most relaxed specification method of the four. It is used only in cases where the DAC or ADC linearity performance is not critical. Thus the order of methods from most relaxed to most demanding is best-straight line, best-fit, endpoint, and absolute.

The choice of technique is not terribly important in DNL calculations. Any of the three techniques will result in nearly identical results, as long as the DAC does not exhibit grotesque gain or linearity errors. DNL values of ±1/2 LSB are usually specified, with typical DAC performance of ±1/4 LSB for reasonably good DAC designs. A 1% error in the measurement of the LSB size would result in only a 0.01 LSB error in the DNL results, which is tolerable in most cases. The choice of technique is actually more important in the integral nonlinearity calculation, which we will discuss in the next section.

EXAMPLE 6.4

Calculate the DNL curve for the 4-bit DAC of the previous examples. Use the best-fit line to define the average LSB size. Does this DAC pass a ±1/2 LSB specification for DNL? Use the endpoint method to calculate the average LSB size. Is this result significantly different from the best-fit calculation?

Solution:

The first derivative of the transfer curve was calculated in the previous monotonicity example. The first derivative values are

75 mV, 175 mV, 75 mV, 55 mV, 75 mV, 175 mV, 75 mV, 195 mV, 75 mV, 175 mV, 75 mV, 55 mV, 75 mV, 175 mV, 75 mV

The average LSB size, 109.35 mV, was calculated in Example 6.1 using the best-fit line calculation. Dividing each step size by the average LSB size yields the following normalized derivative values (in LSBs)

0.686, 1.6, 0.686, 0.503, 0.686, 1.6, 0.686, 1.783, 0.686, 1.6, 0.686, 0.503, 0.686, 1.6, 0.686

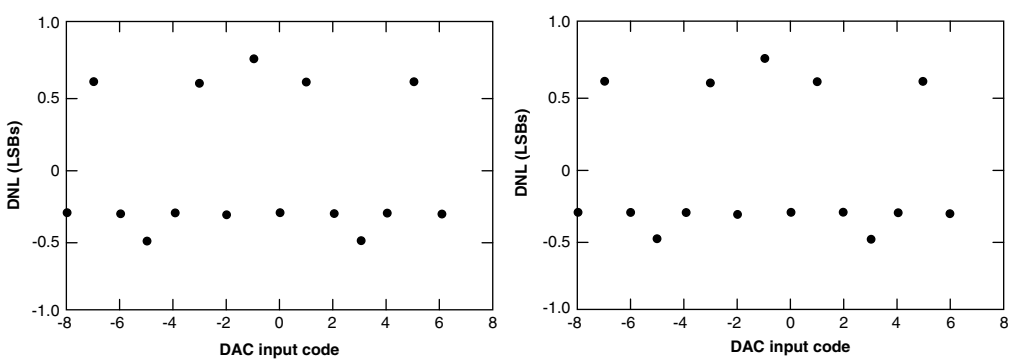
Subtracting one LSB from each of these values gives us the DNL values for each code transition of this DAC expressed as a fraction of an LSB

−0.314, 0.6, −0.314, −0.497, −0.314, 0.6, −0.314, 0.783, −0.314, 0.6, −0.314, −0.497, −0.314, 0.6, −0.314

Note that there is one fewer DNL value than there are DAC codes.

Figure 6.9a shows the DNL curve for this DAC. The maximum DNL value is +0.783 LSB, while the minimum DNL value is –0.497. The minimum value is within the –1/2 LSB test limit, but the

Figure 6.9. 4-bit DAC DNL curve (a) best-fit method (b) endpoint method.



maximum DNL value exceeds the +1/2 LSB limit. Therefore, this DAC fails the DNL specification of ±1/2 LSB.

The average LSB step size calculated using the endpoint method is given by

$$V_{LSB} = \frac{V_{FS+} - V_{FS-}}{\text{number of codes} - 1} = \frac{825\text{ mV} - (-780\text{ mV})}{16 - 1} = 107\text{ mV}$$

The DNL curve calculated using the endpoint method gives the following values, which have been normalized to an LSB size of 107 mV:

−0.299, 0.636, −0.299, −0.486, −0.299, 0.636, −0.299, 0.822, −0.299,
0.636, −0.299, −0.486, −0.299, 0.636, −0.299

The corresponding DNL curve is shown in Figure 6.9b. Using the endpoint calculation, we get slightly different results. Instead of a maximum DNL result of +0.783 LSB and a minimum DNL of –0.497 LSB, we get +0.822 and –0.486 LSB, respectively. This might be enough of a difference compared to the best-fit technique to warrant concern. Unless the endpoint method is explicitly called for in the data sheet, the best-fit method should be used since it is the least sensitive to abnormalities in any one DAC voltage.

Exercises

6.13. Calculate the DNL curve for the 4-bit DAC of Exercise 6.7. Use the best-fit line to define the average LSB size. Does this DAC pass a ±1/2 LSB specification for DNL?

ANS. 0.0937, −0.2481, 0.2714, −0.2622, 0.4259, −0.2002, 0.3170, −0.2577, 0.1320, 0.1067 −0.1686, −0.1607, 0.0739, −0.1669, −0.2194; pass

6.14. Calculate the DNL curve for the 4-bit DAC of Exercise 6.7. Use the endpoint method to calculate the average LSB size. Does this DAC pass a ±1/2 LSB specification for DNL?

ANS. 0.1132 −0.2347, 0.2941 −0.2491, 0.4514 −0.1859, 0.3406 −0.2445, 0.1523, 0.1264 −0.1537 −0.1457, 0.0931 −0.1520 −0.2054; pass

6.3.4 Integral Nonlinearity

The integral nonlinearity curve is a comparison between the actual DAC curve and one of three lines: the best-fit line, the endpoint line, or the ideal DAC line. The INL curve, like the DNL curve, is normalized to the LSB step size. As in the DNL case, the best-fit line is the preferred reference line, since it eliminates sensitivity to individual DAC values. The INL curve can be calculated by subtracting the reference DAC line (best-fit, endpoint, or ideal) from the actual DAC curve, dividing the results by the average LSB step size according to

$$INL(i) = \frac{S(i) - S_{REF}(i)}{V_{LSB}} \tag{6.30}$$

Note that using the ideal DAC line is equivalent to calculating the absolute error curve. Since a separate absolute error test is often specified, the ideal line is seldom used in INL testing. Instead, the endpoint or best-fit line is generally used. As in DNL testing, we are interested in the maximum and minimum value in the INL curve, which we compare against a test limit such as ±1/2 LSB.

EXAMPLE 6.5

Calculate the INL curve for the 4-bit DAC in the previous examples. First use an endpoint calculation, then use a best-fit calculation. Does either result pass a specification of ±1/2 LSB? Do the two methods produce a significant difference in results?

Solution:

Using an endpoint calculation method, the INL curve for the 4-bit DAC of the previous examples is calculated by subtracting a straight line between the V_{FS-} voltage and the V_{FS+} voltage from the DAC output curve. The difference at each point in the DAC curve is divided by the average LSB size, which in this case is calculated using an endpoint method. As in the endpoint DNL example, the average LSB size is equal to 107 mV. The results of the INL calculations are (again, these values are expressed in LSBs)

$$0.0, -0.299, 0.336, 0.037, -0.449, -0.748, -0.112, -0.411,$$
$$0.411, 0.112, 0.748, 0.449, -0.037, -0.336, 0.299, 0.0$$

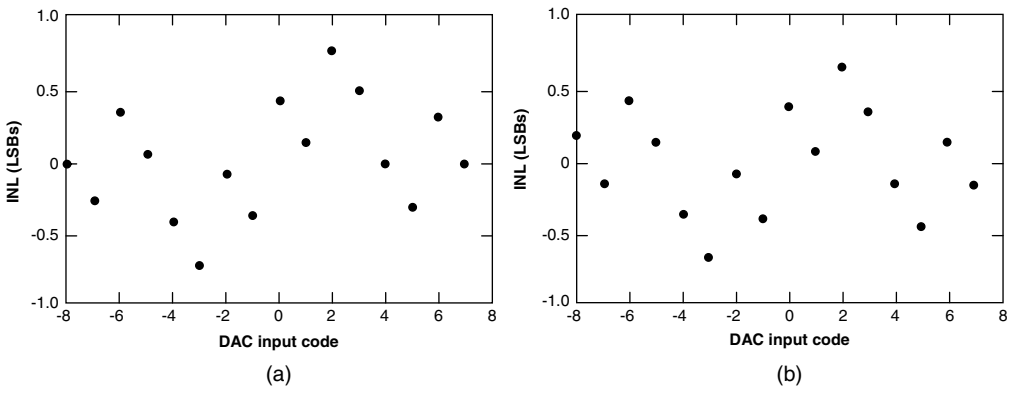
Figure 6.10a shows this endpoint INL curve. The maximum INL value is +0.748 LSB, and the minimum INL value is –0.748. This DAC does not pass an INL specification of ±1/2 LSB.

Using a best-fit calculation method, the INL curve for the 4-bit DAC of the previous examples is calculated by subtracting the best-fit line from the DAC output curve. Each point in the difference curve is divided by the average LSB size, which in this case is calculated using the best-fit line method. As in the best-fit DNL example, the average LSB size is equal to 109.35 mV. The results of the INL calculations are

$$0.161, -0.153, 0.448, 0.133, -0.364, -0.678, -0.077, -0.392, 0.392,$$
$$0.077, 0.678, 0.364, -0.133, -0.448, 0.153, -0.161$$

The maximum value is +0.678, and the minimum value is –0.678. These INL results are better than the endpoint INL values, but still do not pass a ±1/2 LSB test limit. The best-fit INL curve is shown in Figure 6.10b for comparison with the endpoint INL curve. The two INL curves are somewhat similar in shape, but the individual INL values are quite different. Remember that the DNL curves for endpoint and best-fit calculations were nearly identical. So, as previously stated,

Figure 6.10. 4-bit DAC INL curve (a) endpoint method (b) best-fit method.



the choice of calculation technique is much more important for INL curves than for DNL curves. Notice also that while an endpoint INL curve always begins and ends at zero, the best-fit curve does not necessarily behave this way. A best-fit curve will usually give better INL results than an endpoint INL calculation. This is especially true if the DAC curve exhibits a bowed shape in either the upward or downward direction. The improvement in the INL measurement is another strong argument for using a best-fit approach rather than an absolute or endpoint method, since the best-fit approach tends to increase yield.

The INL curve is the integral of the DNL curve, thus the term "integral nonlinearity"; DNL is a measurement of how consistent the step sizes are from one code to the next. INL is therefore a measure of accumulated errors in the step sizes. Thus, if the DNL values are consistently larger than zero for many codes in a row (step sizes are larger than 1 LSB), the INL curve will exhibit an upward bias. Likewise, if the DNL is less than zero for many codes in a row (step sizes are less than 1 LSB), the INL curve will have a downward bias. Ideally, the positive error in one code's DNL will be balanced by negative errors in surrounding codes and vice versa. If this is true, then the INL curve will tend to remain near zero. If not, the INL curve may exhibit large upward or downward bends, causing INL failures.

The INL integration can be implemented using a running sum of the elements of the DNL. The *i*th element of the INL curve is equal to the sum of the first *i*–1 elements of the DNL curve plus a constant of integration. When using the best-fit method, the constant of integration is equal to the difference between the first DAC output voltage and the corresponding point on the best-fit curve, all normalized to one LSB. When using the endpoint method, the constant of integration is equal to zero. When using the absolute method, the constant is set to the normalized difference between the first DAC output and the ideal output. In any running sum calculation it is important to use high-precision mathematical operations to avoid accumulated math error in the running sum. Mathematically, we can express this process as

$$INL(i) = \sum_{k=0}^{i-1} DNL(k) + C \tag{6.31}$$

where

$$C = \begin{cases} \dfrac{S(0) - Best\_fit\_line(0)}{V_{LSB}} & \text{for best-fit linearity method} \\ 0 & \text{for endpoint linearity method} \\ \dfrac{S(0) - S_{IDEAL}(0)}{V_{LSB}} & \text{for absolute linearity method} \end{cases}$$

and $i = 0$ indicates the DAC level corresponding to V_{FS-}.

Conversely, DNL can be calculated by taking the first derivative of the INL curve

$$\mathrm{DNL}(i) = \mathrm{INL}'(i) = \mathrm{INL}(i+1) - \mathrm{INL}(i) \tag{6.32}$$

This is usually the easiest way to calculate DNL. The first derivative technique works well in DAC testing, but we will see in the next chapter that the DNL curve for an ADC is easier to capture than the INL curve. In ADC testing it is more common to calculate the DNL curve first and then integrate it to calculate the INL curve. In either case, whether we integrate DNL to get INL or differentiate INL to get DNL, the results are mathematically identical.

Integral nonlinearity and differential nonlinearity are sometimes referred to by the names integral linearity error (ILE) and differential linearity error (DLE). However, the terms INL and DNL seem to be more prevalent in data sheets and other literature. We will use the terms INL and DNL throughout this text.

Exercises

6.15. Calculate the INL curve for a 4-bit unsigned binary DAC whose DNL curve is described by the following values (in LSBs)
0.0937, −0.2481, 0.2714, −0.2622, 0.4259, −0.2002, 0.3170, −0.2577, 0.1320, 0.1067, −0.1686, −0.1607, 0.0739, −0.1669, −0.2194
The DAC output for code 0 is 1.0091 V. Assume that the best-fit line has a gain of 177.3 mV/bit and an offset of 1.026 V.

ANS. −0.0959, −0.0022, −0.2503, 0.0210, −0.2412, 0.1847, −0.0155, 0.3016, 0.0438, 0.1759, 0.2825, 0.1139, −0.0467, 0.0272, -0.1397, −0.3591

6.3.5 Partial Transfer Curves

A customer or systems engineer may specify that only a portion of a DAC or ADC transfer curve must meet certain specifications. For example, a DAC may be designed so that its V_{FS-} code corresponds to 0 V. However, due to analog circuit clipping as the DAC output signal approaches ground, the DAC may clip to a voltage of 100 mV. If the DAC is designed to perform a specific function that never requires voltages below 100 mV, then the customer may not care about this clipping. In such a case, the DAC codes below 100 mV are excluded from the offset, gain, INL, DNL, and so on. specifications. The test engineer may then treat these codes as if they do not exist. This type of partial DAC and ADC testing is becoming more common as more DACs and ADCs are designed into custom applications with very specific requirements. General-purpose DACs are unlikely to be specified using partial curves, since the customer's application needs are unknown.

6.3.6 Major Carrier Testing

The techniques discussed thus far for measuring INL and DNL are based on a testing approach called *all-codes testing*. In all-codes testing, all valid codes in the transfer curve are measured to determine the INL and DNL values. Unfortunately, all-codes testing can be a very time-consuming process.

Depending on the architecture of the DAC, it may be possible to determine the location of each voltage in the transfer curve without measuring each one explicitly. We will refer to this as selected-code testing. Selected-code testing can result in significant test-time savings, which of course represents substantial savings in test cost. There are several selected-code testing techniques, the simplest of which is called the *major carrier* method.

Many DACs are designed using an architecture in which a series of binary-weighted resistors or capacitors are used to convert the individual bits of the converter code into binary-weighted currents or voltages. These currents or voltages are summed together to produce the DAC output. For instance, a binary-weighted unsigned binary D-bit DAC's output can be described as a sum of binary-weighted voltage or current values, $W_0, W_1, \ldots, W_n$, multiplied by the individual bits of the DAC's input code, $b_{D-1}, b_{D-2}, \ldots, b_2, b_1$. The DAC's output value is therefore equal to

$$\text{DAC Output} = b_0W_0 + b_1W_1 + \cdots + b_{D-2}W_{D-2} + b_{D-1}W_{D-1} + \text{DC Base} \quad \textbf{(6.33)}$$

where

- DC base is the DAC output value with a V_{FS-} input code.
- DAC code bits $b_{D-1}, b_{D-2}, ..., b_2, b_1, b_0$; take on values of 1 or 0.

If this idealized model of the DAC is sufficiently accurate, then we only need to make *D+1* measurements of DAC behavior and solve for the unknown model parameters: W_0, W_1, …, W_{D-1} and the DC Base term. Subsequently, we can cycle through all binary values using Eq. (6.33) and compute the entire DAC transfer curve. This DAC testing method is called the major carrier technique. The major carrier approach can be used for ADCs as well as DACs. The assumption of sufficient DAC or ADC model accuracy is only valid if the actual superposition errors of the DAC or ADC are low. This may or may not be the case. The superposition assumption can only be determined through characterization, comparing the all-codes DAC output levels with the ones generated by the major carrier method.

The most straightforward way to obtain each model parameter W_n is to set code bit b_n to 1 and all other to zero. This is then repeated for each code bit for n from 0 to *D*—1. However, the resulting output levels are widely different in magnitude. This makes them difficult to measure accurately with a voltmeter, since the voltmeter's range must be adjusted for each measurement. A better approach that alleviates the accuracy problem is to measure the step size of the major carrier transitions in the DAC curve, which are all approximately 1 LSB in magnitude. A major carrier transition is defined as the voltage (or current) transition between the DAC codes 2^n–1 and 2^n. For example, the transition between binary 00111111 and 01000000 is a major carrier transition for $n = 6$. Major carrier transitions can be measured using a voltmeter's sample-and-difference mode, giving highly accurate measurements of the major carrier transition step sizes.

Once the step sizes are known, we can use a series of inductive calculations to find the values of W_0, W_1, …, W_{D-1}. We start by realizing that we have actually measured the following values:

$$\text{DC base} = \text{measured DAC output with minus full-scale code}$$
$$V_0 = W_0$$
$$V_1 = W_1 - W_0$$
$$V_2 = W_2 - (W_1 + W_0)$$
$$V_3 = W_3 - (W_2 + W_1 + W_0)$$
$$\cdots$$
$$V_n = W_n - (W_{n-1} + W_{n-2} + W_{n-3} + \cdots + W_0)$$

The value of the first major transition, V_0, is a direct measurement of the value of W_0 (the step size of the least significant bit). The value of W_1 can be calculated by rearranging the second equation: $W_1 = V_1 + W_0$. Once the values of W_0 and W_1 are known, the value of W_2 is calculated by rearranging the third equation: $W_2 = V_2 + W_1 + W_0$, and so forth. Once the values of W_0–W_n are known, the complete DAC curve can be reconstructed for each possible combination of input bits $b_0 - b_n$ using the original model of the DAC described by Eq. (6.33).

The major carrier technique can also be used on signed binary and two's complement converters, although the codes corresponding to the major carrier transitions must be chosen to match the converter encoding scheme. For example, the last major transition for our two's complement 4-bit DAC example happens between code 1111 (decimal –1) and 0000 (decimal 0). Aside from these minor modifications in code selection, the major carrier technique is the same as the simple unsigned binary approach.

EXAMPLE 6.6

Using the major carrier technique on the 4-bit DAC example, we measure a DC base of −780 mV setting the DAC to V_{FS-} (binary 1000, or −8). Then we measure the step size between 1000 (−8) and 1001 (−7). The step size is found to be 75 mV. Next we measure the step size between 1001 (−7) and 1010 (−6). This step size is 175 mV. The step size between 1011 (−5) and 1100 (−4) is 55 mV and the step size between 1111 (−1) and 0000 (0) is 195 mV. Determine the values of W_0, W_1, W_2, and W_3. Reconstruct the voltages on the ramp from DAC code −8 to DAC code +7.

Solution:
Rearranging the set of equations $V_n = W_n - (W_{n-1} + W_{n-2} + W_{n-3} + \ldots + W_0)$ to solve for W_n, we obtain

$$\text{DC baseline} = \text{measured DAC output with } V_{FS-} \text{ code} = -780 \text{ mV}$$
$$W_0 = V_0 = 75 \text{ mV}$$
$$W_1 = V_1 + W_0 = 175 \text{ mV} + 75 \text{ mV} = 250 \text{ mV}$$
$$W_2 = V_2 + W_1 + W_0 = 380 \text{ mV}$$
$$W_3 = V_3 + W_2 + W_1 + W_0 = 900 \text{ mV}$$

For a two's complement DAC, we have to realize that the most significant bit is inverted in polarity compared to an unsigned binary DAC. Therefore, the DAC model for our 4-bit DAC is given by

$$\text{DAC output} = b_0W_0 + b_1W_1 + b_2W_2 + \overline{b_3}W_3 + \text{DC base} \tag{6.34}$$

Using this two's complement version of the DAC model, the 16 voltage values of the DAC curve are reconstructed as shown in Table 6.4. Notice that these values are exactly equal to the all-codes results in Figure 6.8. The example DAC was created using a binary-weighted model with perfect superposition; so it is no surprise the major carrier technique works for this imaginary DAC. Real DACs and ADCs often have superposition errors that make the major carrier technique unusable.

Table 6.4. DAC Transfer Curve Calculated Using the Major Carrier Technique

DAC Code	Calculation	Output Voltage (mV)
1000	DC Base	−780
1001	W_0+DC Base	−705
1010	W_1+DC Base	−530
1011	W_1+W_0+DC Base	−455
1100	W_2+DC Base	−400
1101	W_2+W_0+DC Base	−325
1110	W_2+W_1+DC Base	−150
1111	W_2+W_1+W0+DC Base	−75
0000	W_3+DC Base	120
0001	W_3+W_0+DC Base	195
0010	W_3+W_1+DC Base	370
0011	W_3+W_1+W_0+DC Base	445
0100	W_3+W_2+DC Base	500
0101	W_3+W_2+W_0+DC Base	575
0110	W_3+W_2+W_1+DC Base	750
0111	W_3+W_2+W_1+W_0+DC Base	825

6.3.7 Other Selected-Code Techniques

Besides the major carrier method, other selected-code techniques have been developed to reduce the test time associated with all-codes testing. The simplest of these is the segmented method. This method only works for certain types of DAC and ADC architectures, such as the 12-bit segmented DAC shown in Figure 6.11. Although most segmented DACs are actually constructed using a different architecture than that in Figure 6.11, this simple architecture is representative of how segmented DACs can be tested.

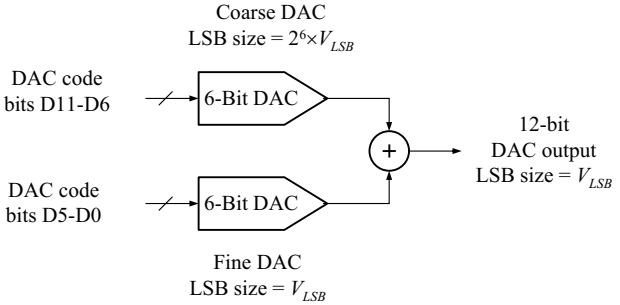
The example DAC uses a simple unsigned binary encoding scheme with 12 data bits, D11-D0. It consists of two portions, a 6-bit coarse resolution DAC and a 6-bit fine resolution DAC. The LSB step size of the coarse DAC is equal to the full-scale range of the fine DAC plus one fine DAC LSB. In other words, if the combined 12-bit DAC has an LSB size of V_{LSB}, then the fine DAC also has a step size of V_{LSB}, while the coarse DAC has a step size of $2^6 \times V_{LSB}$. The output of these two 6-bit DACs can therefore be summed together to produce a 12-bit DAC

$$\text{DAC output} = \text{coarse DAC contribution} + \text{fine DAC contribution} \qquad \textbf{(6.35)}$$

Both the fine DAC and the coarse DAC are designed using a resistive divider architecture rather than a binary-weighted architecture. Since major carrier testing can only be performed on binary-weighted architectures, an all-codes testing approach must be used to verify the performance of each of the two 6-bit resistive divider DACs. However, we would like to avoid testing each of the 212, or 4096 codes of the composite 12-bit DAC. Using superposition, we will test each of the two 6-bit DACs using an all-codes test. This requires only 2 × 26, or 128 measurements. We will then combine the results mathematically into a 4096-point all-codes curve using a linear model of the composite DAC.

Let us assume that through characterization, it has been determined that this example DAC has excellent superposition. In other words, the step sizes of each DAC are independent of the setting of the other DAC. Also, the summation circuit has been shown to be highly linear. In a case such as this, we can measure the all-codes output curve of the coarse DAC while the fine DAC is set to 0 (i.e., D5-D0 = 000000). We store these values into an array $V_{DAC\text{-}COARSE}(n)$, where n takes on the values 0 to 63, corresponding to data bits D11-D6. Then we can measure the all-codes output curve for the fine DAC while the coarse DAC is set to 0 (i.e., D11-D6 = 000000). These voltages are stored in the array $V_{DAC\text{-}FINE}(n)$, where n corresponds to data bits D5-D0.

Figure 6.11. Segmented DAC conceptual block diagram.



Although we have only measured a total of 128 levels, superposition allows us to recreate the full 4096-point DAC output curve by a simple summation. Each DAC output value $V_{DAC}(i)$ is equal to the contribution of the coarse DAC plus the contribution of the fine DAC

$$V_{DAC}(i) = V_{DAC-FINE}(i \text{ AND } 000000111111) + V_{DAC-COARSE}\left(\frac{i \text{ AND } 111111000000}{64}\right) \quad \textbf{(6.36)}$$

where i ranges from 0 to 4095.

Thus a full 4096-point DAC curve can be mathematically reconstructed from only 128 measurements by evaluating this equation at each value of i from 0 to 4095. Of course, this technique is totally dependent on the architecture of the DAC. It would be inappropriate to use this technique on a nonsegmented DAC or a segmented DAC with large superposition errors.

A more advanced selected-codes technique was developed at the National Institute of Standards and Technology (NIST). This technique is useful for all types of DACs and ADCs. It does not make any assumptions about superposition errors or converter architecture. Instead, it uses linear algebra and data collected from production lots to create an empirical model of the DAC or ADC. The empirical model only requires a few selected codes to recreate the entire DAC or ADC transfer curve. Although the details of this technique are beyond the scope of this book, the original NIST paper is listed in the references at the end of this chapter.[3]

Another similar technique uses wavelet transforms to predict the overall performance of converters based on a limited number of measurements.[4] Again, this topic is beyond the scope of this book.

Exercises

6.16. The step sizes between the major carries of a 4-bit unsigned binary DAC were measured to be as follows

code 0→1: 0.2010 V; code 1→2: 0.1987 V; code 3→4: 0.1877 V; code 7→8: 0.1998 V

Determine the values of W_0, W_1, W_2, and W_3.

ANS. $W_0 = 0.2010$, $W_1 = 0.3997$, $W_2 = 0.7884$, $W_3 = 1.5889$

6.4 DYNAMIC DAC TESTS

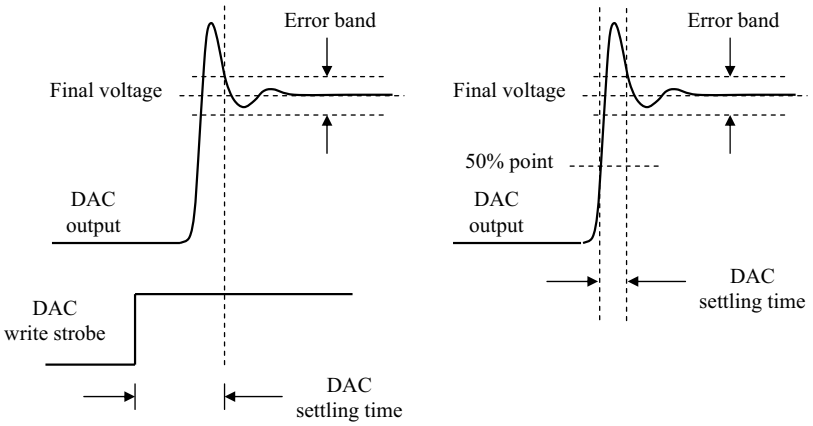
6.4.1 Conversion Time (Settling Time)

So far we have discussed only low-frequency DAC performance. The DAC DC tests and transfer curve tests measure the DAC's static characteristics, requiring the DAC to stabilize to a stable voltage or current level before each output level measurement is performed. If the DAC's output stabilizes in a few microseconds, then we might step through each output state at a high frequency, but we are still performing static measurements for all intents and purposes.

A DAC's performance is also determined by its dynamic characteristics. One of the most common dynamic tests is settling time, commonly referred to as *conversion time*. Conversion time is defined as the amount of time it takes for a DAC to stabilize to its final static level *within a specified error band* after a DAC code has been applied. For instance, a DAC's settling time may be defined as 1 μs to ±1/2 LSB. This means that the DAC output must stabilize to its final value plus or minus a 1/2 LSB error band no more than 1 μs after the DAC code has been applied.

This test definition has one ambiguity. Which DAC codes do we choose to produce the initial and final output levels? The answer is that the DAC must settle from any output level to any other level within the specified time. Of course, to test every possibility, we might have to measure

Figure 6.12. DAC settling time measurement (a) referenced to a digital signal; (b) referenced to the DAC output 50% point.



millions of transitions on a typical DAC. As with any other test, we have to determine what codes represent the worst-case transitions. Typically settling time will be measured as the DAC transitions from minus full-scale (V_{FS-}) to plus full-scale (V_{FS+}) and vice versa, since these two tests represent the largest voltage swing.

The 1/2 LSB example uses an error band specification that is referenced to the LSB size. Other commonly used definitions require the DAC output to settle within a certain percentage of the full-scale range, a percentage of the final voltage, or a fixed voltage range. So we might see any of the following specifications:

settling time = 1 μs to ± 1% of full-scale range
settling time = 1 μs to ± 1% of final value
settling time = 1 μs to ±1 mV

The test technique for all these error-band definitions is the same; we just have to convert the error-band limits to absolute voltage limits before calculating the settling time. The straightforward approach to testing settling time is to digitize the DAC's output as it transitions from one code to another and then use the known time period between digitizer samples to calculate the settling time. We measure the final settled voltage, calculate the settled voltage limits (i.e., ±1/2 LSB), and then calculate the time between the digital signal transition that initiates a DAC code change and the point at which the DAC first stays within the error band limits, as shown in Figure 6.12a.

In extremely high frequency DACs it is common to define the settling time not from the DAC code change signal's transition but from the time the DAC passes the 50% point to the time it settles to the specified limits as shown in Figure 6.12b. This is easier to calculate, since it only requires us to look at the DAC output, not at the DAC output relative to the digital code.

6.4.2 Overshoot and Undershoot

Overshoot and undershoot can also be calculated from the samples collected during the DAC settling time test. These are defined as a percentage of the voltage swing or as an absolute voltage. Figure 6.13 shows a DAC output with 10% overshoot and 2% undershoot on a V_{FS-} to V_{FS+} transition.

Figure 6.13. DAC overshoot and undershoot measurements.

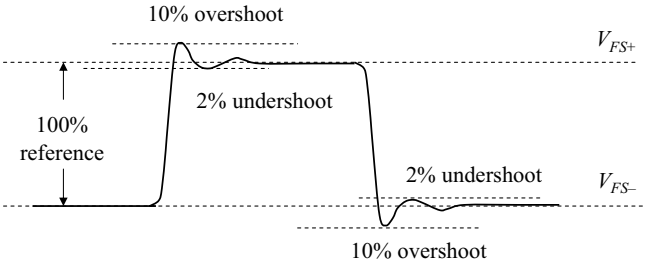
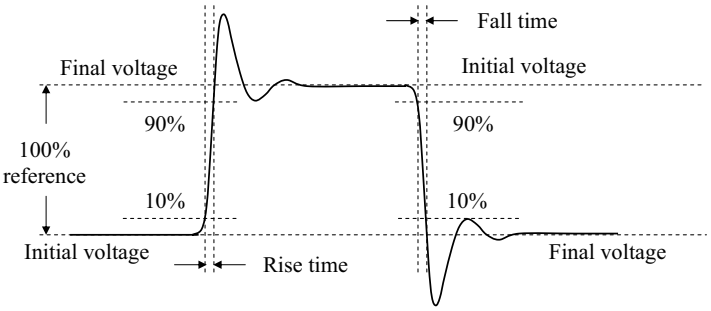


Figure 6.14. DAC rise and fall time measurements.



6.4.3 Rise Time and Fall Time

Rise and fall time can also be measured from the digitized waveform collected during a settling time test. Rise and fall times are typically defined as the time between two markers, one of which is 10% of the way between the initial value and the final value and the other of which is 90% of the way between these values as depicted in Figure 6.14. Other common marker definitions are 20% to 80% and 30% to 70%.

6.4.4 DAC-to-DAC Skew

Some types of DACs are designed for use in matched groups. For example, a color palette RAM DAC is a device that is used to produce colors on video monitors. A RAM DAC uses a random access memory (RAM) lookup table to turn a single color value into a set of three DAC output values, representing the red, green, and blue intensity of each pixel. These DAC outputs must change almost simultaneously to produce a clean change from one pixel color to the next. The degree of timing mismatch between the three DAC outputs is called *DAC-to-DAC skew*. It is measured by digitizing each DAC output and comparing the timing of the 50% point of each output to the 50% point of the other outputs. There are three skew values (R-G, G-B, and B-R), as illustrated in Figure 6.15. Skew is typically specified as an absolute time value, rather than a signed value.

6.4.5 Glitch Energy (Glitch Impulse)

Glitch energy, or glitch impulse, is another specification common to high-frequency DACs. It is defined as the total area under the voltage-time curve of the glitches in a DAC's output as it

Figure 6.15. DAC-to-DAC skew measurements.

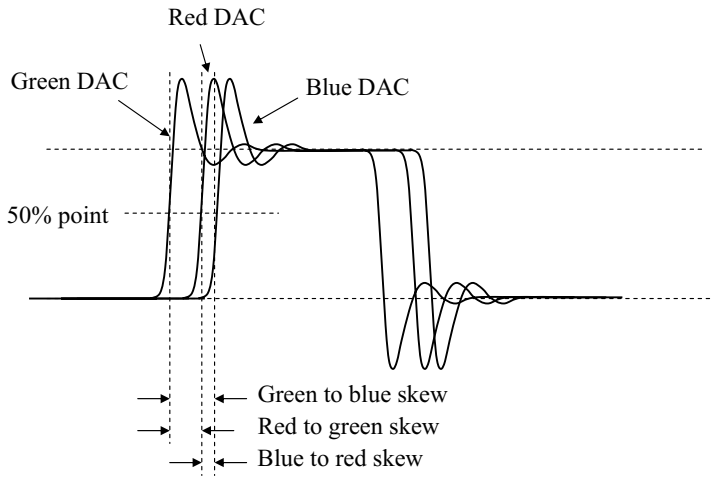
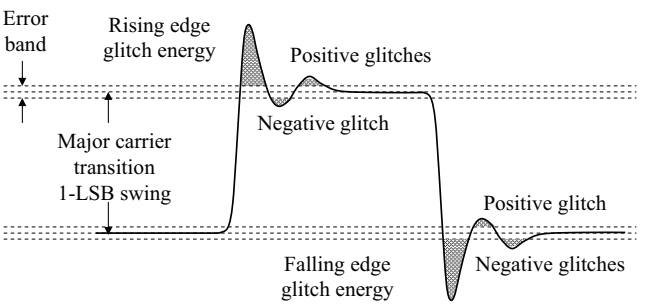


Figure 6.16. Glitch energy measurements.



switches across the largest major transition (i.e., 01111111 to 10000000 in an 8-bit DAC) and back again. As shown in Figure 6.16, the glitch area is defined as the area that falls *outside* the rated error band. These glitches are caused by a combination of capacitive/inductive ringing in the DAC output and skew between the timing of the digital bits feeding the binary-weighted DAC circuits. The parameter is commonly expressed in picosecond-volts (ps-V) or equivalently, picovolt-seconds (pV-s). (These are not actually units of energy, despite the term *glitch energy*.) The area under the negative glitches is considered positive area, and should be added to the area under the positive glitches. Both the rising-edge glitch energy and the falling-edge glitch energy should be tested.

6.4.6 Clock and Data Feedthrough

Clock and data feedthrough is another common dynamic DAC specification. It measures the cross-talk from the various clocks and data lines in a mixed-signal circuit that couple into a DAC output. There are many ways to define this parameter; so it is difficult to list a specific test technique.

However, clock and data feedthrough can be measured using a technique similar to all the other tests in this section. The output of the DAC is digitized with a high-bandwidth digitizer. Then the various types of digital signal feedthrough are analyzed to make sure they are below the defined test limits. The exact test conditions and definition of clock and data feedthrough should be provided in the data sheet. This measurement may require time-domain analysis, frequency-domain analysis, or both.

6.5 TESTS FOR COMMON DAC APPLICATIONS

6.5.1 DC References

As previously mentioned, the test list for a given DAC often depends on its intended functionality in the system-level application. Many DACs are used as simple DC references. An example of this type of DAC usage is the power level control in a cellular telephone. As the cellular telephone user moves closer or farther away from a cellular base station (the radio antenna tower), the transmitted signal level from the cellular telephone must be adjusted. The transmitted level may be adjusted using a transmit level DAC so that the signal is just strong enough to be received by the base station without draining the cellular telephone's battery unnecessarily.

If a DAC is only used as a DC (or slow-moving) voltage or current reference, then its AC transmission parameters are probably unimportant. It would probably be unnecessary to measure the 1-kHz signal to total harmonic distortion ratio of a DAC whose purpose is to set the level of a cellular telephone's transmitted signal. However, the INL and DNL of this DAC would be extremely important, as would its absolute errors, monotonicity, full-scale range, and output drive capabilities (output impedance).

DACs used as DC references are usually measured using the intrinsic parameters listed in this chapter, rather than the transmission parameters outlined in Chapter 11. Notable exceptions are signal-to-noise ratio and idle channel noise (ICN). These may be of importance if the DC level must exhibit low noise. For example, the cellular telephone's transmitted signal might be corrupted by noise on the output of the transmit level control DAC, and therefore we might need to measure the DAC's noise level.

Dynamic tests are not typically performed on DC reference DACs, with the exception of settling time. The settling time of typical DACs is often many times faster than that required in DC reference applications; so even this parameter is frequently guaranteed by design rather than being tested in production.

6.5.2 Audio Reconstruction

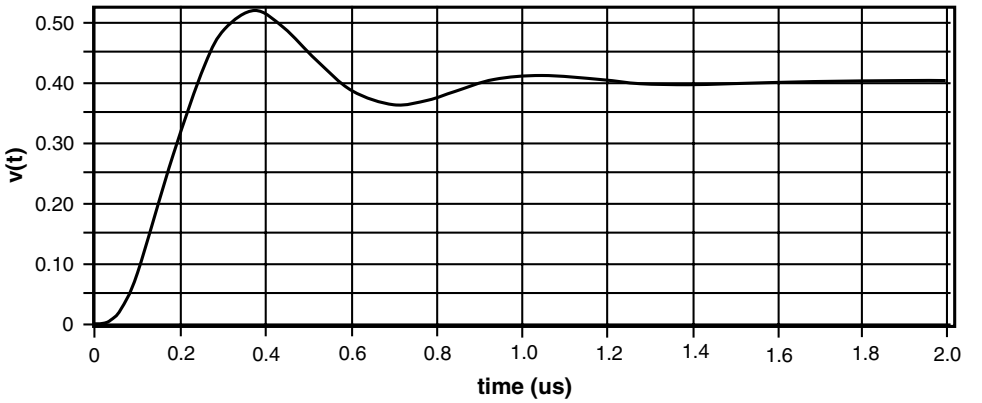
Audio reconstruction DACs are those used to reproduce digitized sound. Examples include the voice-band DAC in a cellular telephone and the audio DAC in a PC sound card. These DACs are more likely to be tested using the transmission parameters of Chapter 11, since their purpose is to reproduce arbitrary audio signals with minimum noise and distortion.

The intrinsic parameters (i.e., INL and DNL) of audio reconstruction DACs are typically measured only during device characterization. Linearity tests can help track down any transmission parameter failures caused by the DAC. It is often possible to eliminate the intrinsic parameter tests once the device is in production, keeping only the transmission tests.

Dynamic tests are not typically specified or performed on audio DACs. Any failures in settling time, glitch energy, and so on, will usually manifest themselves as failures in transmission parameters such as signal-to-noise, signal-to-distortion, and idle channel noise.

Exercises

6.17. The step response of a DAC obtained from an oscilloscope is as follows:



The data sheet states that the settling time is 1 μs (error band = ± 20 mV). Does this DAC settle fast enough to meet the settling time specification? Also, determine the overshoot of this signal and its rise time. Estimate the total glitch energy during the positive-going transition.

ANS. Actual settling time = 0.82 μs; yes; overshoot = 30%; rise time = 0.2 μs. Glitch energy = 0.5(0.3)(0.13) + 0.5(0.5)(-0.033) + 0.5(0.6)(0.01) = 14 ns-V (triangle approximation).

6.5.3 Data Modulation

Data modulation is another purpose to which DACs are often applied. The cellular telephone again provides an example of this type of DAC application. The IF section of a cellular telephone base-band modulator converts digital data into an analog signal suitable for transmission, similar to those used in modems (see Section 11.1.2). Like the audio reconstruction DACs, these DACs are typically tested using sine wave or multitone transmission parameter tests.

Again, the intrinsic tests like INL and DNL may be added to a characterization test program to help debug the design. However, the intrinsic tests are often removed after the transmission parameters have been verified. Dynamic tests such as settling time may or may not be necessary for data modulation applications.

Data modulation DACs also have very specific parameters such as error vector magnitude (EVM) or phase trajectory error (PTE). Parameters such as these are very application-specific. They are usually defined in standards documents published by the IEEE, NIST, or other government or industry organization. The data sheet should provide references to documents defining application-specific tests such as these. The test engineer is responsible for translating the measurement requirements into ATE-compatible tests that can be performed on a production tester. ATE vendors are often a good source of expertise and assistance in developing these application-specific tests.

6.5.4 Video Signal Generators

As discussed earlier, DACs can be used to control the intensity and color of pixels in video cathode ray tube (CRT) displays. However, the type of testing required for video DACs depends on the nature of their output. There are two basic types of video DAC application, RGB and NTSC. An RGB (red-green-blue) output is controlled by three separate DACs. Each DAC controls the intensity of an electron beam, which in turn controls the intensity of one of the three primary colors of each pixel as the beam is swept across the CRT. In this application, each DAC's output voltage or current directly control the intensity of the beam. RGB DACs are typically used in computer monitors.

The NTSC format is used in transmission of standard (i.e., non-HDTV) analog television signals. It requires only a single DAC, rather than a separate DAC for each color. The picture intensity, color, and saturation information is contained in the time-varying offset, amplitude, and phase of a 3.54-MHz sinusoidal waveform produced by the DAC. Clearly this is a totally different DAC application than the RGB DAC application. These two seemingly similar video applications require totally different testing approaches.

RGB DACs are tested using the standard intrinsic tests like INL and DNL, as well as the dynamic tests like settling time and DAC-to-DAC skew. These parameters are important because the DAC outputs directly control the rapidly changing beam intensities of the red, green, and blue electron beams as they sweep across the computer monitor. Any settling time, rise time, fall time, undershoot, or overshoot problems show up directly on the monitor as color or intensity distortions, vertical lines, ghost images, and so on.

The quality of the NTSC video DAC, by contrast, is determined by its ability to produce accurate amplitude and phase shifts in a 3.54-MHz sine wave while changing its offset. This type of DAC is tested with transmission parameters like gain, signal-to-noise, differential gain, and differential phase (topics of Chapters 10 and 11).

6.6 SUMMARY

DAC testing is far less straightforward than one might at first assume. Although DACs all perform the same basic function (digital-to-analog conversion), the architecture of the DAC and its intended application determine its testing requirements and methodologies. A large variety of standard tests have been defined for DACs, including transmission parameters, DC intrinsic parameters, and dynamic parameters. We have to select DAC test requirements carefully to guarantee the necessary quality of the DAC without wasting time with irrelevant or ineffective tests.

ADC testing is very closely related to DAC testing. Many of the DC and intrinsic tests defined in this chapter are very similar to those performed on ADCs. However, due to the many-to-one transfer characteristics of ADCs, the measurement of the ADC input level corresponding to each output code is much more difficult than the measurement of the DAC output level corresponding to each input code. Chapter 7, "ADC Testing," explains the various ways the ADC transfer curve can be measured, as well as the many types of ADC architectures and applications the test engineer will likely encounter.

PROBLEMS

6.1. Given a set of N points denoted by $S(i)$, derive the parameters of a straight line described by

$$\text{Best\_fit\_line}(i) = \text{gain} \times i + \text{offset} \quad \text{for } i = 0,1,\ldots,N-1$$

that minimizes the following mean-square error criteria

$$\overline{e^2} = \sum_{i=0}^{N-1}\left[S(i) - \text{Best\_fit\_line}(i)\right]^2 = \sum_{i=0}^{N-1}\left[S(i) - \text{gain} \times i + \text{offset}\right]^2$$

Hint: Find partial derivatives $\partial\overline{e^2}/\partial\, gain$ and $\partial\overline{e^2}/\partial\, offset$, set them both to zero, and solve for the two unknowns, gain and offset, from the system of two equations.

6.2. The output levels of a 4-bit DAC produces the following set of voltage levels, starting from code 0 and progressing through to code 15:

0.0465, 0.3255, 0.7166, 1.0422, 1.5298, 1.8236, 2.1693, 2.5637,

2.8727, 3.3443, 3.6416, 4.0480, 4.3929, 4.7059, 5.0968, 5.5050

What is the full-scale range of this DAC?

6.3. A 4-bit DAC has a full-scale voltage range of 0 to 1.0 V. The input is formatted using an unsigned binary number representation. List all possible ideal output levels. What output level corresponds to the DAC input code 0?

6.4. A 5-bit DAC has a full-scale voltage range of 2.0 to 4.0 V. The input is formatted using a 2's complement number representation. List all possible ideal output levels. What output level corresponds to the DAC input code 0?

6.5. A 4-bit ADC has a full-scale voltage range of −10 to 10 V. The internal operation of the ADC is based on truncation and the input digital signal is formatted using a two's complement binary number representation. List all possible ideal output levels. What is the LSB step size?

6.6. A 5-bit ADC has a full-scale voltage range of 0 to 5 V. The internal operation of the ADC is such that all code widths are equal. Also, the input digital signal is formatted using a unsigned binary number representation. List all possible ideal output levels. What is the LSB step size?

6.7. A 4-bit unsigned binary DAC produces the following set of voltage levels, starting from code 0 and progressing through to code 15

0.0465, 0.3255, 0.7166, 1.0422, 1.5298, 1.8236, 2.1693, 2.5637,
2.8727, 3.3443, 3.6416, 4.0480, 4.3929, 4.7059, 5.0968, 5.5050

The ideal DAC output at code 0 is 0 V and the ideal gain is equal to 400 mV/bit. Answer the following questions assuming a best-fit line is used as a reference.
(a) Calculate the DAC's gain (volts per bit), gain error, offset and offset error.
(b) What is the LSB step size?
(c) Calculate the absolute error transfer curve for this DAC. Normalize the result to one LSB.
(d) Is the DAC output monotonic?
(e) Compute the DNL curve for this DAC. Does this DAC pass a ±1/2 LSB specification for DNL?

6.8. A 4-bit unsigned binary DAC produces the following set of voltage levels, starting from code 0 and progressing through to code 15

0.0465, 0.3255, 0.7166, 1.0422, 1.5298, 1.8236, 2.1693, 2.5637,
2.8727, 3.3443, 3.6416, 4.0480, 4.3929, 4.7059, 5.0968, 5.5050

The ideal DAC output at code 0 is 0 V and the ideal gain is equal to 400 mV/bit. Answer the following questions assuming an endpoint-to-endpoint line is used as a reference.
(a) Calculate the DAC's gain (volts per bit), gain error, offset and offset error.
(b) What is the LSB step size?
(c) Calculate the absolute error transfer curve for this DAC. Normalize the result to one LSB.
(d) Is the DAC output monotonic?
(e) Compute the DNL curve for this DAC. Does this DAC pass a ±1/2 LSB specification for DNL?

6.9. A 4-bit two's complement DAC produces the following set of voltage levels, starting from code -8 and progressing through to code +7

−0.9738, −0.8806, −0.6878, −0.6515, −0.3942, −0.3914, −0.2497, −0.1208,
−0.0576, 0.1512, 0.2290, 0.4460, 0.4335, 0.5999, 0.6743, 0.8102

The ideal DAC output at code 0 is 0 V and the ideal gain is equal to 133.3 mV/bit. Answer the following questions assuming a best-fit line is used as a reference.
(a) Calculate the DAC's gain (volts per bit), gain error, offset and offset error.

(b) Estimate the LSB step size using its measured full-scale range. What is the gain error and offset error?
(c) Calculate the absolute error transfer curve for this DAC. Normalize the result to one LSB.
(d) Is the DAC output monotonic?
(e) Compute the DNL curve for this DAC. Does this DAC pass a ±1/2 LSB specification for DNL?

6.10. A 4-bit two's complement DAC produces the following set of voltage levels, starting from code −8 and progressing through to code +7

−0.9738, −0.8806, −0.6878, −0.6515, −0.3942, −0.3914, −0.2497, −0.1208, −0.0576, 0.1512, 0.2290, 0.4460, 0.4335, 0.5999, 0.6743, 0.8102

The ideal DAC output at code 0 is 0 V and the ideal gain is equal to 133.3 mV/bit. Answer the following questions assuming a endpoint-to-endpoint line is used as a reference.
(a) Calculate the DAC's gain (volts per bit), gain error, offset and offset error.
(b) Estimate the LSB step size using its measured full-scale range. What is the gain error and offset error?
(c) Calculate the absolute error transfer curve for this DAC. Normalize the result to one LSB.
(d) Is the DAC output monotonic?
(e) Compute the DNL curve for this DAC. Does this DAC pass a ±1/2 LSB specification for DNL?

6.11. Calculate the INL curve for a 4-bit unsigned binary DAC whose DNL curve is described by the following values

−0.0815, −0.1356, −0.1133, 0.0057, 0.0218, 0.1308, −0.0361, −0.0950, 0.1136, −0.1633, 0.2101, 0.0512, 0.0119, −0.0706, −0.0919

The DAC output for code 0 is −0.4919 V. Assume that the best-fit line has a gain of 63.1 mV/bit and an offset of −0.5045 V. Does this DAC pass a ±1/2 LSB specification for INL?

6.12. Calculate the DNL curve for a 4-bit DAC whose INL curve is described by the following values

0.1994, 0.1180, −0.0177, −0.1310, −0.1253, −0.1036, 0.0272, −0.0089, −0.1039, 0.0096, −0.1537, 0.0565, 0.1077, 0.1196,0.0490, −0.0429

Does this DAC pass a ±1/2 LSB specification for DNL?

6.13. The step sizes between the major carries of a 5-bit unsigned binary DAC were measured to be as follows

code 0 → 1: 0.1939 V, code 1 → 2: 0.1333 V, code 3 → 4: 0.1308 V, code 7 → 8: 0.1316 V, code 15 → 16: 0.1345 V

Determine the values of W_0, W_1, W_2, W_3, and W_4. Reconstruct the voltages on the ramp from DAC code 0 to DAC code 31 if the DC base value is 100 mV.

6.14. The step sizes between the major carries of a 4-bit two's complement DAC were measured to be as follows:

code −8 → −7: 0.1049 V, code −7 → −6: 0.1033 V, code −5 → −4: 0.0998 V, code −1 → 0: 0.1016 V

Determine the values of W_0, W_1, W_2, and W_3. Reconstruct the voltages on the ramp from DAC code −8 to DAC code +7 if the DC base value is 500 mV.

6.15. Can a major carrier test technique be used to describe a 4-bit unsigned DAC if the output levels beginning with code 0 were found to be the following

0.0064, 0.0616, 0.1271, 0.1812, 0.2467, 0.3206, 0.3856, 0.4406,

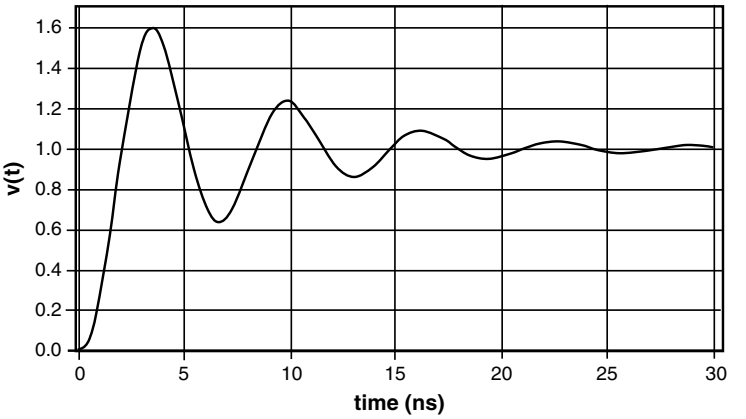
0.5021, 0.5716, 0.6364, 0.6880, 0.7662, 0.8262, 0.8871, 0.9480

What if the DAC output levels were described by the following

0.0064, 0.0616, 0.1271, 0.1823, 0.2478, 0.3030, 0.3684, 0.4236,
0.4851, 0.5403, 0.6058, 0.6610, 0.7264, 0.7816, 0.8471, 0.9023

Explain your reasoning.

6.16. The step response of a DAC obtained from an oscilloscope is as follows:



The data sheet states that the settling time is 10 ns (error band = ±50 mV). Does this DAC settle fast enough to meet this specification? Also, determine the overshoot of this signal and its rise time. Estimate the total glitch energy during the positive-going transition.

6.17. Using MATLAB or equivalent software, evaluate the following expression for the step response of a circuit using a time step of no larger than 20 ns

$$v(t) = 1 + \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin\left(\omega_n t\sqrt{1-\zeta^2} - \cos^{-1}\zeta\right)$$

where $\omega_n = 2\pi \times 100$ MHz and $\zeta = 0.3$. Determine the time for circuit to settle to within 1% of its final value. Determine the rise time.

REFERENCES

1. M. Mahoney, *Tutorial DSP-Based Testing of Analog and Mixed-Signal Circuits*, The Computer Society of the IEEE, 1730 Massachusetts Avenue N.W., Washington D.C. 200361903, 1987, ISBN: 0818607858.
2. G. W. Snedecor and W. G. Cochran, *Statistical Methods*, Eighth Edition, Iowa State University Press, 1989, ISBN: 0813815614, pp. 149–176.
3. G. N. Stenbakken, T. M. Souders, *Linear Error Modeling of Analog and Mixed-Signal Devices*, Proc. International Test Conference, 1991.
4. T. Yamaguchi, M. Soma, *Dynamic Testing of ADCs Using Wavelet Transforms*, Proc. International Test Conference, 1997, pp. 379–88.