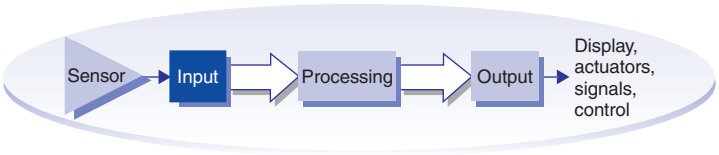

Analog-to-Digital Conversion

Part 5 in a series of tutorials in instrumentation and measurement

David A. Rauth and Vincent T. Randal

In Part 4 of the series [1], the signal conditioning stage for a variety of transducer types and sensors was discussed. The output of this stage in the measurement chain is typically an analog signal that can drive an actuator or display or be converted into digital form. Since the advent of monolithic Analog/digital converters (ADCs) in the 1970s, digital measurement systems have become the standard. Today, almost all signal processing takes place in the digital domain where signals can be flexibly manipulated in complex ways when compared to their analog counterparts. Before we can enter the Utopian realm of digital signal processing, analog signals need to be converted into digital form, which is the subject of this tutorial.

Conversion Fundamentals

Understanding how A/D conversion works requires that the fundamental elements of the process be defined and examined. Generally speaking, an ADC performs amplitude quantization. But the overall conversion process also is concerned about antialias filtering and a sample/hold, as shown in Figure 1.

Commercial ADCs commonly include most of these stages in one package, but the conversion process is easier to understand if they are treated as independent subcomponents before discussing them as a single unit. The subcomponents can be modeled mathematically, allowing the reader to gain an understanding of what real converter implementations attempt to achieve.

Signal Conditioning

Signal conditioning matches the raw signal from the sensor to the ADC. More specifically, it matches the raw signal to the requirements of the sample-and-hold (S/H) circuit within the ADC. It also serves to isolate the sensor from the ADC. Part 4 of this tutorial series, which published in the June 2005 issue of *IEEE Instrumentemt Magazine*, covers signal conditioning

Sampling—Some Theory

The output of the signal conditioning (SC) stage is a continuous analog signal $x_f(t)$ that is a function of time. Because SC often includes an antialiasing filter, we'll skip over it temporarily and get right to sampling.

Also called quantization, sampling is performed by a circuit that can be thought of as a switch that periodically opens and closes according to a clock signal. The sampler stays open for a given sampling period T_s and then

1094-6969/05/$20.00©2005IEEE

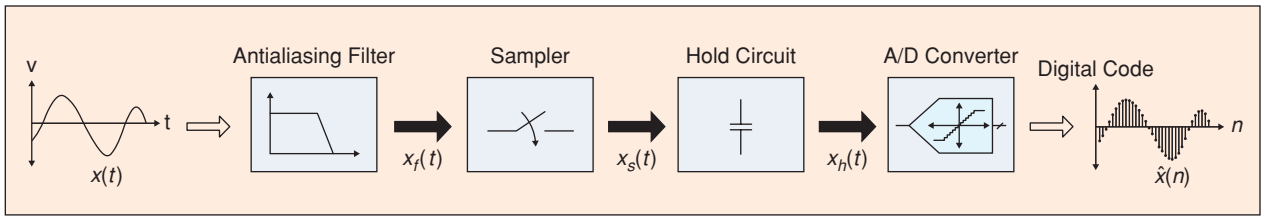

Fig. 1. A complete A/D system.

closes for an infinitesimally short time. This operation transforms the analog signal to a discrete-time signal $x_s(t)$ that is defined for every instant of time. Sampling can be modeled using the sifting property of the Dirac delta function (or unit impulse function), which is defined as

$$\delta(t) \triangleq \begin{cases} \infty, & t = 0 \\ 0, & t \neq 0 \end{cases} \text{ where } \int_{-\infty}^{\infty} \delta(t)\,dt = 1. \qquad (1)$$

The output of the sampler is the product of the analog signal $x_f(t)$ and a train of unit impulses occurring at integer multiples n of the sampling period T_s and can be expressed as

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_f(nT_s)\,\delta(t - nT_s). \qquad (2)$$

The resulting signal $x_s(t)$ has the same amplitude as the continuous-time signal at integer multiples of T_s and is zero elsewhere (i.e., the sample is sifted out of the continuous-time signal).

Notice that the sampling period T_s governs the resolution of the discrete-time signal with respect to the analog signal. For the discrete-time signal to perfectly match the analog signal, the sampling period T_s would have to be infinitesimally small. Since an ADC takes a finite amount of time to perform a conversion, the digital signal doesn't contain any information about the analog signal that occurred during conversion. Thus, the digital version will have finite bandwidth. The sampling process also replicates the frequency spectrum of the analog signal at integer multiples of the sampling frequency f_s. Figure 2(a) illustrates the spectrum and the spectral replicas.

Carrying the mathematical theory a little further shows the spectral replication of the original signal. Taking the Fourier transform of (2) converts the Dirac delta functions into a linear combination of sinusoidal carriers, each with its own frequency shift. Using the modulation property of the Fourier transform [2], which states that if $X(f)$ is the transform of $x(t)$, then $X(f - f_c)$ is the transform of $x(t)e^{2\pi jf_c t}$, yields

$$X_s(f) = \sum_{n=-\infty}^{\infty} x_f(nT_s)e^{-2\pi jfT_s n} = \frac{1}{T}\sum_{n=-\infty}^{\infty} X_f(f - nf_s). \qquad (3)$$

Aliasing—Some More Theory

If the frequency spectrum of the analog signal is large (i.e., has wide bandwidth) and the sampling frequency f_s is low, then these spectral replicas will overlap, as seen in Figure 2(b). This phenomenon is called aliasing, and it distorts the digital signal.

Aliasing can be avoided entirely if the sampling frequency is more than two times higher than the highest spectral frequency of the signal BW; this means you should choose $f_s \geq 2\,BW$. This is called the Nyquist criterion after Harry Nyquist, who did pioneering work at Bell Labs in the late 1920s. When the Nyquist criterion is satisfied, the digital signal will have

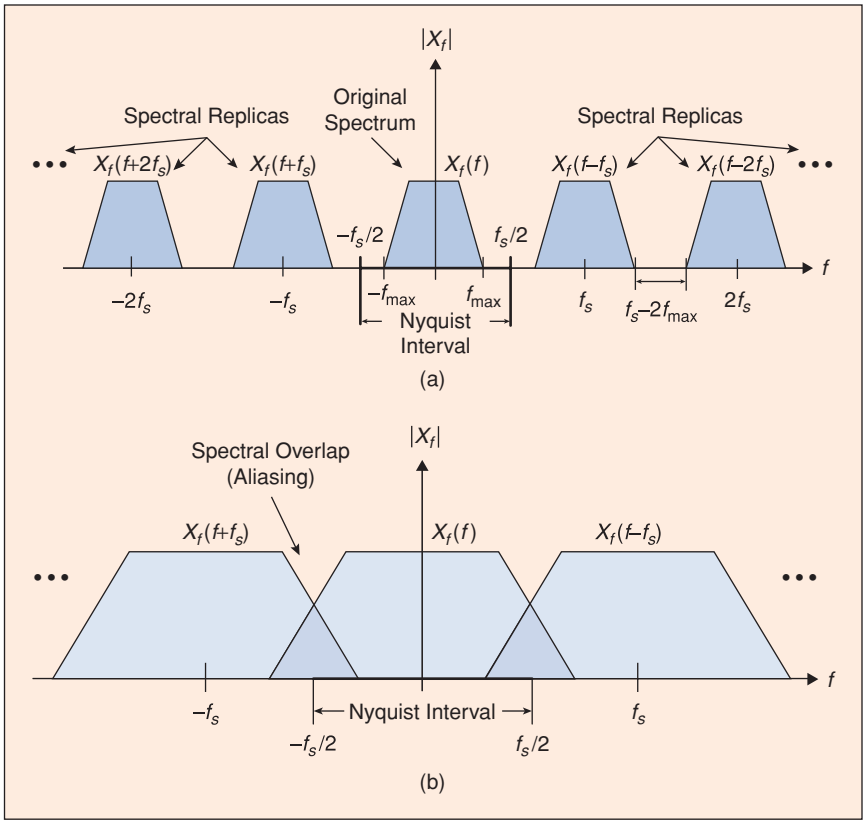

Fig. 2. (a) Spectrum replication caused by the sampling process and (b) aliasing.

no distortion, and the original (band-limited) analog signal can be perfectly reconstructed. Even if signal frequencies are apparently bounded, the presence of noise means that it is unlikely that $f_s \geq 2\,BW$ can be guaranteed. This is where the antialiasing filter comes into play.

The final step in the A/D conversion process is the amplitude-quantization of the discrete-time signal.

Antialiasing Filter

The antialiasing filter prevents aliasing by limiting the bandwidth of the raw analog signal $x_f(t)$ and noise so that its highest frequency component is less than half of the sampling frequency. (See $\pm f_{\max}$ in Figure 2(a). The magnitudes of $f_{\max}$ are less than $|f_s/2|$.) A real implementation of an antialiasing filter can't cut off exactly at half the sampling frequency, and it can't have infinite attenuation above the cut-off. Instead, a practical design trades off filter complexity for a higher sampling frequency. For example, it is common to pick a sampling frequency three to five times higher than the signal bandwidth BW while the filter cut-off frequency is set at BW.

One simple and popular implementation of an antialiasing filter is the RC filter [see Figure 3(a)]. It is a single-pole filter that has a rolloff attenuation of 20 dB per decade. (This means that the noise and signal amplitude decreases by ten times for every ten times increase in frequency above the corner cutoff frequency, f_c, of the filter.)

To implement an RC filter, the filter's corner cutoff frequency, f_c, should be significantly less than $f_{\max}$ of the signal to achieve the necessary attenuation. The values of the resistor, R, and the capacitor, C, determine the corner cutoff frequency, f_c. Choose the values of R and C to satisfy the following equation for f_c:

$$f_c = \frac{1}{2\pi RC}. \tag{4}$$

If you need sharper rolloff attenuations of 40, 60, or more decibels per decade, then standard "cookbook" filter designs exist. Figure 3(b) shows just one example of an active filter. One important consideration is the unity-gain buffer on the output of the filter; it prevents loading of the filter by the ADC circuitry.

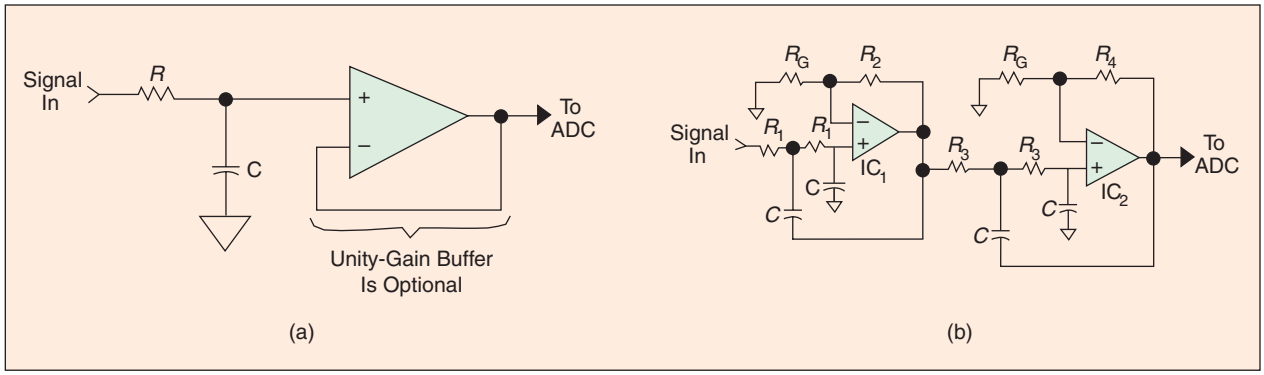

Fig. 3. (a) A simple RC filter for antialiasing. (b) Example of an active fourth-order filter.

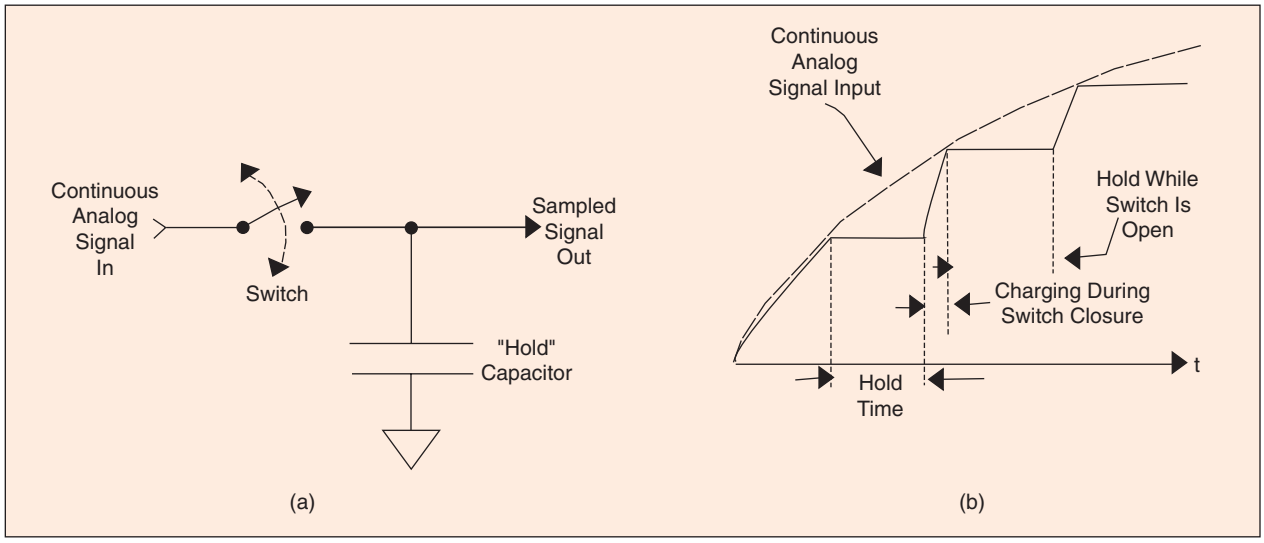

Fig. 4. (a) An example of an S/H circuit. (b) Operation of the S/H.

Hold That Sample

Quantization cannot be performed by an ADC architecture in an infinitesimal amount of time. Thus, the amplitude of the discrete-time signal must be held constant for a sufficient amount of time after being sampled. This is accomplished by a hold circuit, the simplest of which is a zero-order hold (ZOH). The ideal ZOH can be thought of as a capacitance that is instantaneously charged to the amplitude of the discrete-time signal at each sampling instant and then holds that value until the quantization is complete.

The majority of conversion elements are effectively combined into a single device.

Figure 4(a) outlines a S/H circuit and its operation. The switch described in the "Sampling" section is actually part of this S/H circuit. The switch closes for a short, but finite, amount of time to charge the "hold" capacitor. When the switch opens, the capacitor "holds" the voltage at a constant value. Figure 4(b) illustrates the operation of the S/H.

Amplitude Quantization

The final step in the A/D conversion process is the amplitude-quantization of the discrete-time signal. Amplitude quantization codes a signal into a binary representation using a fixed number of bits B. Since the discrete-time signal amplitude is analog (infinite valued), perfect representation requires an infinite number of bits. However, this level of precision is neither practical nor needed. Instead, a fixed-number of bits is used to represent the signal based on the precision required by the application.

Just as sampling limited the frequency resolution of the digital signal, amplitude quantization limits the amplitude resolution of the digital signal. More bits provide higher resolution and smaller quantization steps. The quantization step is the minimum change in amplitude needed to change the least significant bit (LSB) of the binary representation.

Another factor to consider is the full-scale range R of the converter. If the analog input amplitude is not constrained to be within R, the output will be clipped whenever the maximum A/D input range is exceeded. In a unipolar ADC, the full-scale range extends from 0 to R. In a bipolar converter, it typically extends from $-R/2$ to $R/2$. The range could be centered about any amplitude, depending on the needs of the application.

The quantization step size Q of a converter can be expressed as

$$Q = \frac{R}{2^B}. \tag{5}$$

Note that (5) indicates that the amplitude quantization is uniform, meaning that each quantization step is exactly the same size throughout the full-scale range. Practically, there will be errors, and while most ADCs are intended to quantize uniformly, some specialized applications use nonuniform quantization (e.g., voice channels).

Because only a finite number of bits is available to an ADC, there will be an imperfect mapping of input amplitudes to their corresponding code values. The amount of error is also dependent on whether truncation or rounding schemes are used. In truncation-based quantization, the LSB changes when the discrete-time signal amplitude exceeds the quantization step value; this biases the error around half the LSB. Rounding-based quantization is preferred over truncation because it removes the error bias by changing the LSB when the discrete-time signal amplitude exceeds half the quantization step value. The quantization error e_q, using the rounding scheme, is $-Q/2 \le e_q \le Q/2$.

Signal-to-Noise Ratio

Quantization error sets the theoretical noise floor for an ideal ADC, which sets the limit for the theoretical signal-to-noise ratio (SNR). If you assume a sine wave input, then the following derivation gives the theoretical SNR of an ideal ADC:

$$\begin{aligned} \text{signal}_{\text{rms}} &= \text{signal}_{\text{amplitude}} \big/ \sqrt{2} \\ &= (\text{fullscale}/2) \big/ \sqrt{2} = 2^{n-1} q/\sqrt{2}, \end{aligned}$$

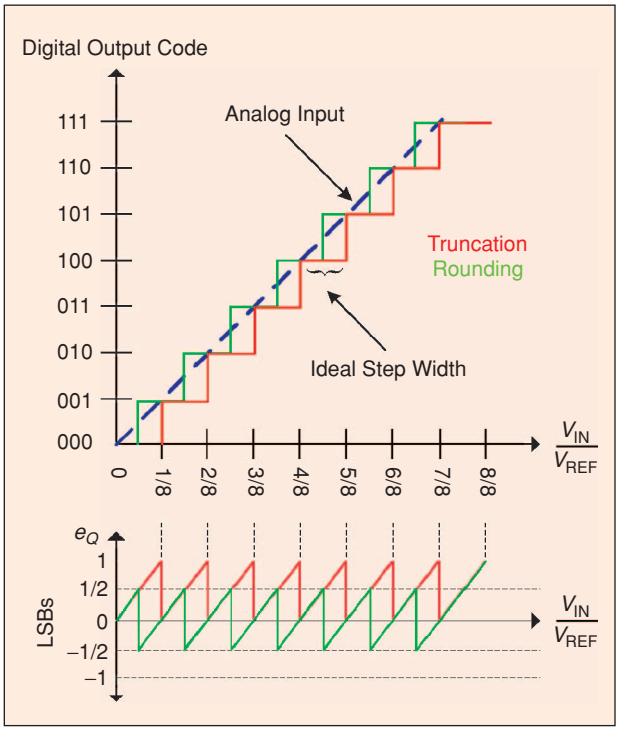

Fig. 5. Transfer curves for an ideal A/D converter using truncation and rounding based quantization and corresponding quantization error.

Table 1. Example codes for a three-bit bipolar quantizer with a 10 V (±5 V) full-scale range.

Unsigned	Signed	Quant. Step	Offset Binary	2's Complement
7	3	3.75	111	011
6	2	2.50	110	010
5	1	1.25	101	001
4	0	0	100	000
3	−1	−1.25	011	111
2	−2	−2.50	010	110
1	−3	−3.75	001	101
0	−4	−5.00	000	100

where q is the LSB value and $\text{noise}_{\text{rms}} = q\sqrt{12}$, which is the rms value of the quantization error from Figure 5. SNR is the ratio of the signal rms to the noise rms

$$\text{SNR} = \frac{2^{n-1}q/\sqrt{2}}{q/\sqrt{2}} = 2^{n-1}\sqrt{6}.$$

Therefore, the SNR of an ideal ADC, with n bits of resolution, is

$$\text{SNR(dB)} = 20\log_{10}\left(2^{n-1}\sqrt{6}\right) = 6.02n + 1.76. \qquad (6)$$

Figure 5 illustrates a unipolar 3-bit quantizer transfer function and error plot for both the truncation and rounding schemes. A unipolar transfer function is shown because it conveys the same idea as the bipolar version while reducing the complexity of the figure.

Other Considerations

Some thought must be given to the choice of binary coding scheme employed. This is primarily of importance to bipolar converters that have to deal with representing negative values (unipolar converters typically use natural binary coding). The two schemes commonly used to represent negative outputs in a bipolar converter are offset binary and two's complement coding. In offset binary, the most significant bit (MSB) of the output represents the sign; an MSB of 1 is positive and an MSB of 0 is negative. For example, a 3-bit offset binary output of 111 represents +3; an output of 000 is a −4. Notice that the positive and negative ranges cannot be symmetrical about zero. One unappealing feature of offset binary is the unnatural representation of zero, e.g., 100 for a 3-bit quantizer (see Table 1). In contrast, two's complement coding corrects for this by swapping the meaning of the MSB sign bit; an MSB of 1 is a negative sign and an MSB of 0 is a positive sign. The two's complement representation of zero for the same 3-bit quantizer is 000. Two's complement coding also makes sense arithmetically. For example, 100 (−4) + 0111(+3) = 111(−1).

Realities of Conversion

So far, our discussion has emphasized a theoretical view of A/D conversion. But it's the fine print—the deviations from ideal behavior—that often make the difference between success and failure in an ADC application. As noted earlier, the majority of conversion elements are effectively combined into a single device. In fact, some ADCs, like the AD7794 mentioned previously, [1] even contain the analog signal conditioning front-end on the same chip.

Table 2. Common S/H circuit parameters.

Parameter	Description
Acquisition Time	Time between when a sampling command is issued and the analog signal is tracked to within a certain tolerance, e.g., 0.01%.
Aperture Error	Time required to disconnect the hold capacitor from the input (a function of input signal noise and amplitude).
Sampling Error	Slight variations in the hold capacitor voltage occurring when a periodic signal is sampled repeatedly at the same points.
Droop	Hold capacitor voltage decrease between samples due to current leakage to ground and through sampler switch.
Pedestal Error	Hold capacitor voltage decreases due to charge injection and clock feed-through occurring during release of sampler switch in a CMOS implementation.
Slew Rate	The rate at which the output of an S/H amplifier can change for a given time period (affects acquisition time).
Overshoot	Amplifier nonlinearity caused by dynamic, large-signal transitions.
Settling Time	Time required for amplifier overshoot transient to subside to a certain tolerance (affects acquisition time).
Offset	DC amplifier output bias.
Linearity and Gain Error	S/H amplifier characteristics that cause distortion of the output; a S/H amplifier should have unity gain.

Sample and Hold Circuitry

A real S/H circuit does not process the input signal instantaneously nor does it track the input signal amplitude (voltage) perfectly [3]. This results in a number of errors that limit the performance of the S/H and therefore the overall performance of the A/D system. Table 2 describes some of the parameters associated with a real S/H circuit; these are illustrated in Figure 6.

Each of the converter architectures has its own unique advantages and disadvantages.

Quantizer

Unlike the ideal model described earlier, a real quantizer's transfer function behaves nonuniformly due to its nonideal subcomponents [3]. Each quantizer step may not be equal in width causing what is known as differential nonlinearity (DNL). If the DNL is too great for any quantization step, the quantizer may have a missing code, meaning that a quantization step is skipped and the signal will never be assigned the corresponding code. A quantizer is said to have integral nonlinearity (INL) if some codes do not transition on a best-fit straight line drawn through the endpoints of the first and last code transitions. Other quantizer errors are associated with deviations from a straight line drawn through the center of the transfer characteristic with a slope of one (representing an infinite-bit quantizer transfer function). An offset error exists when there is a difference between the value of the first code transition and the ideal value (1/2 LSB for a rounding quantizer). If the slope of the transfer characteristic differs from the ideal value of one, then a gain error exists. Each converter architecture should be examined to understand the unique errors that should be considered. Figure 7 depicts examples of the general quantizer errors; see [4] for a more complete explanation of DNL and INL.

SNR Revisited

One way to express how all of the S/H, quantizer, and similar errors affect the overall performance of the ADC is by computing an effective number of bits (ENOB) measure. The ENOB is calculated using SINAD, which is signal-to-noise-and-distortion. SINAD is also measured experimentally for a real converter because noise sources other than quantization normally exist. The ENOB is

$$\text{ENOB} = \frac{\text{SINAD} - \log_2(A/R) - 1.76}{6.02}, \tag{7}$$

where A is the peak-to-peak amplitude of the input signal and R is the full scale range of the ADC. Note that if the signal amplitude is the maximum that will not be clipped by the ADC, the log term is zero. The ENOB is a measure of accuracy that includes the total noise and distortion; the subject ADC has the same noise and distortion as that of an ideal ADC with the same full scale range and ENOB bits.

Converter Architectures

Five converter architectures are widely used to perform amplitude quantization: flash, pipeline, integrating, successive approximation, and oversampling converters. Each of these architectures has its own unique advantages and disadvantages, which should be investigated when determining which A/D best meets an application requirement.

Flash and Pipeline

Flash or parallel converters have the highest conversion speed of any of the architectures and can be typically found in communications equipment, digital oscilloscopes, video capture, and other high-speed instruments. A flash converter, like the one diagramed in Figure 8, uses one comparator for every quantization level. One input of each comparator is connected to the input signal and the other is connected to a reference voltage through a resistor network that divides the reference by the number of quantization levels. The input

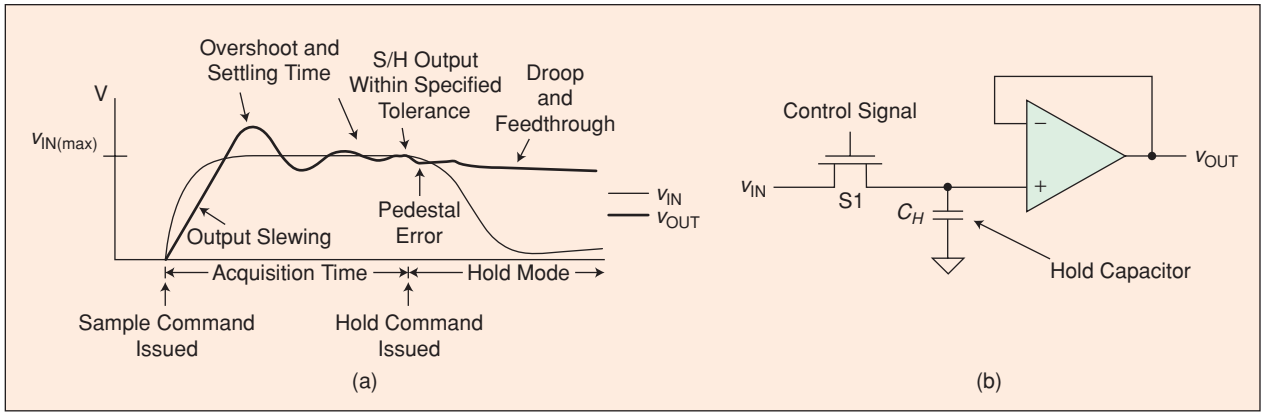

Fig. 6. (a) Typical S/H errors and (b) a typical circuit.

voltage is simultaneously compared with each reference value; the result of each comparator output is a 1 if the input is higher than its reference and a 0 if lower. The combined result of the comparator output is known as a thermometer code (the height of the 1's grows and shrinks with the input voltage—analogous to the column of alcohol in the thermometer capillary tube) and must be encoded to produce the final converter output. Since the conversion is performed simultaneously for all quantization levels (comparators), the flash converter can operate at speeds of several billion samples per second (GS/s). The big disadvantages of the flash converter architecture are size and power requirements: a B-bit flash converter requires $2^B - 1$ comparators; for example, a 16-bit flash converter would need 65,535 comparators. A clever way to reduce the number of comparators and still maintain high throughput is to use the comparators sequentially instead of in parallel. This type of converter is known as a pipeline or subranging converter. In a pipeline converter, a B-bit converter requires only B comparators, and the output of each comparator is a bit in the final converter output (i.e., no thermometer code). The disadvantages of the pipeline architecture are that the comparators must be very

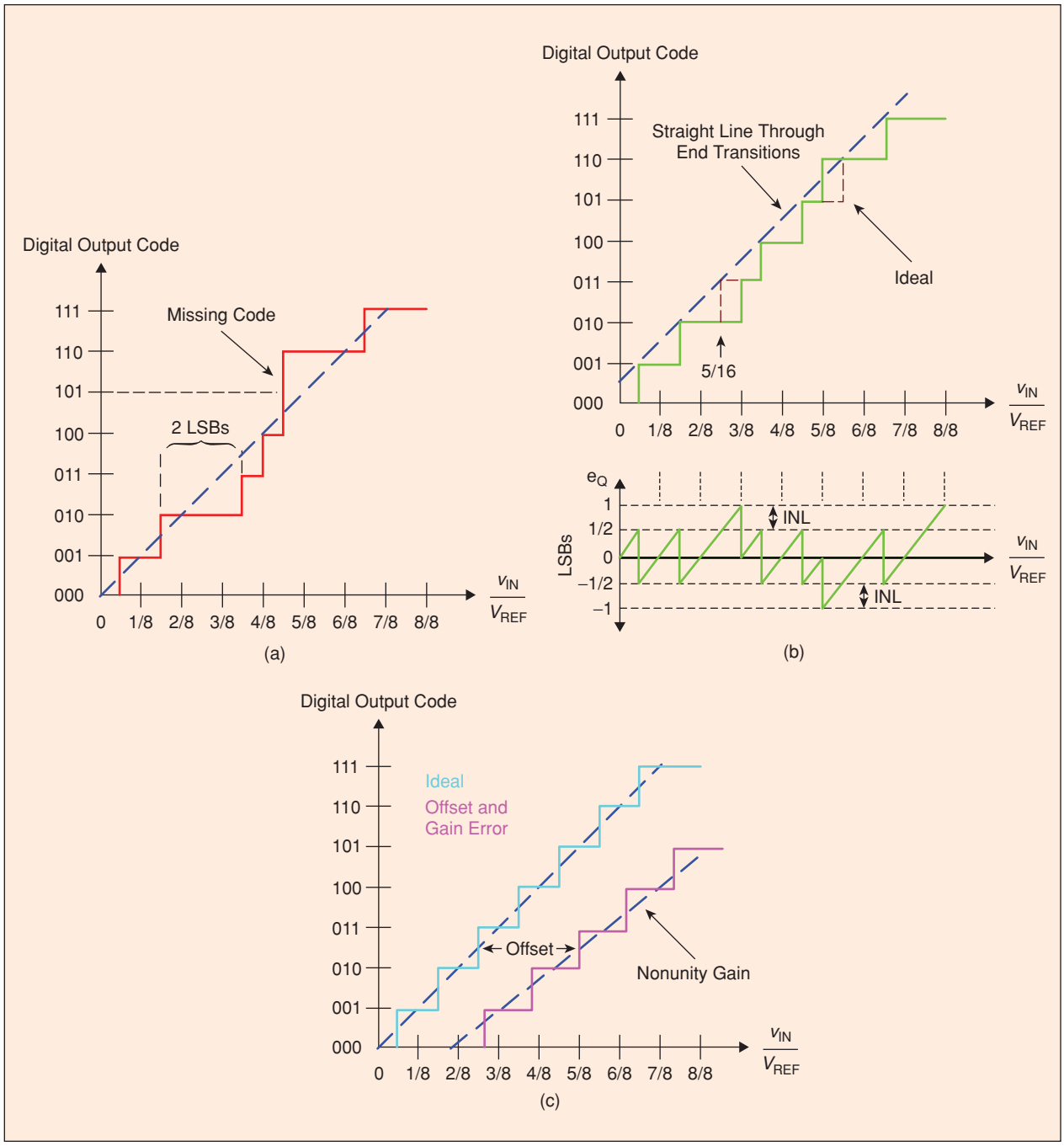

Fig. 7. Quantizer error examples.

precise to prevent error compounding down the pipeline, and there is a B clock latency (for a B-bit converter) as the pipeline is filled. However, after the initial latency, a new output is produced for each subsequent clock cycle.

A/D architectures that support higher numbers of bits imply longer conversion times and/or higher clock rates.

Integrating

The integrating architecture is low speed, low cost, and high resolution, making it ideal for devices like digital multimeters and temperature sensors. An integrating converter integrates the input signal and correlates the integration time with a digital counter. The output of the counter is proportional to the amplitude of the sample. A single-slope converter samples the input and then simultaneously resets the counter and begins integrating a reference voltage. Since the reference voltage is dc, the output of the integrator starts at zero and linearly increases with a slope that depends on the gain of the integrator. When the integrator output exceeds the sample voltage, a comparator switches state causing the counter output to be latched. The single-slope converter has many disadvantages that make it unappealing when compared to other architectures. Speed is a major concern; the single-slope converter requires 2^B clock pulses to perform a full-scale conversion. To accomplish a conversion in a timely manner, the clock driving the digital counter must operate at a frequency much greater than the bandwidth of the input signal. Another major disadvantage is accuracy; the accuracy of the converter is dependent on the precision of the passive components making up the integrator, the frequency jitter of the clock, and nonlinearities of the integrator's amplifier. One method of eliminating most of this error is to perform a dual integration of both the reference voltage and the input signal. In a dual-slope converter, the input sample (which is assumed to be negative) is integrated for a fixed time dictated by the digital counter. When the counter overflows, a switch is thrown, a positive reference voltage is connected to the integrator, and the counter is simultaneously reset. When the integrator output reaches zero (fully discharged) a comparator switches state, thereby latching the counter output. The dual-slope converter is (worst case) twice as slow as the single-slope converter for the same number of bits due to the extra integration. However, the advantage to the dual-slope architecture is that the precision and nonlinearity issues that reduced the accuracy of the single-slope version are cancelled out because the same circuitry is used for both the reference voltage and the sample voltage integration. Figure 9 shows a block diagram of a dual-slope converter.

Successive Approximation

Successive approximation is one of the most popular converter architectures for a wide variety of measurement products

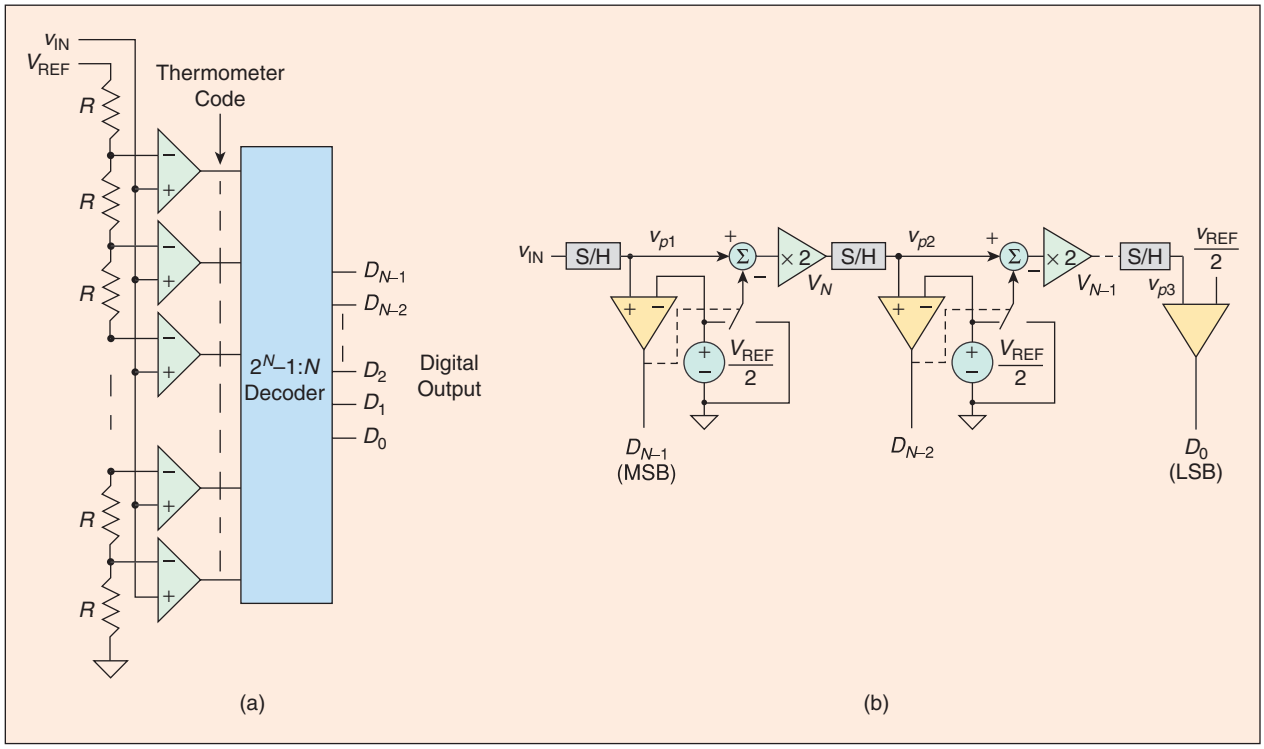

Fig. 8. Typical (a) flash and (b) pipeline converter block diagram.

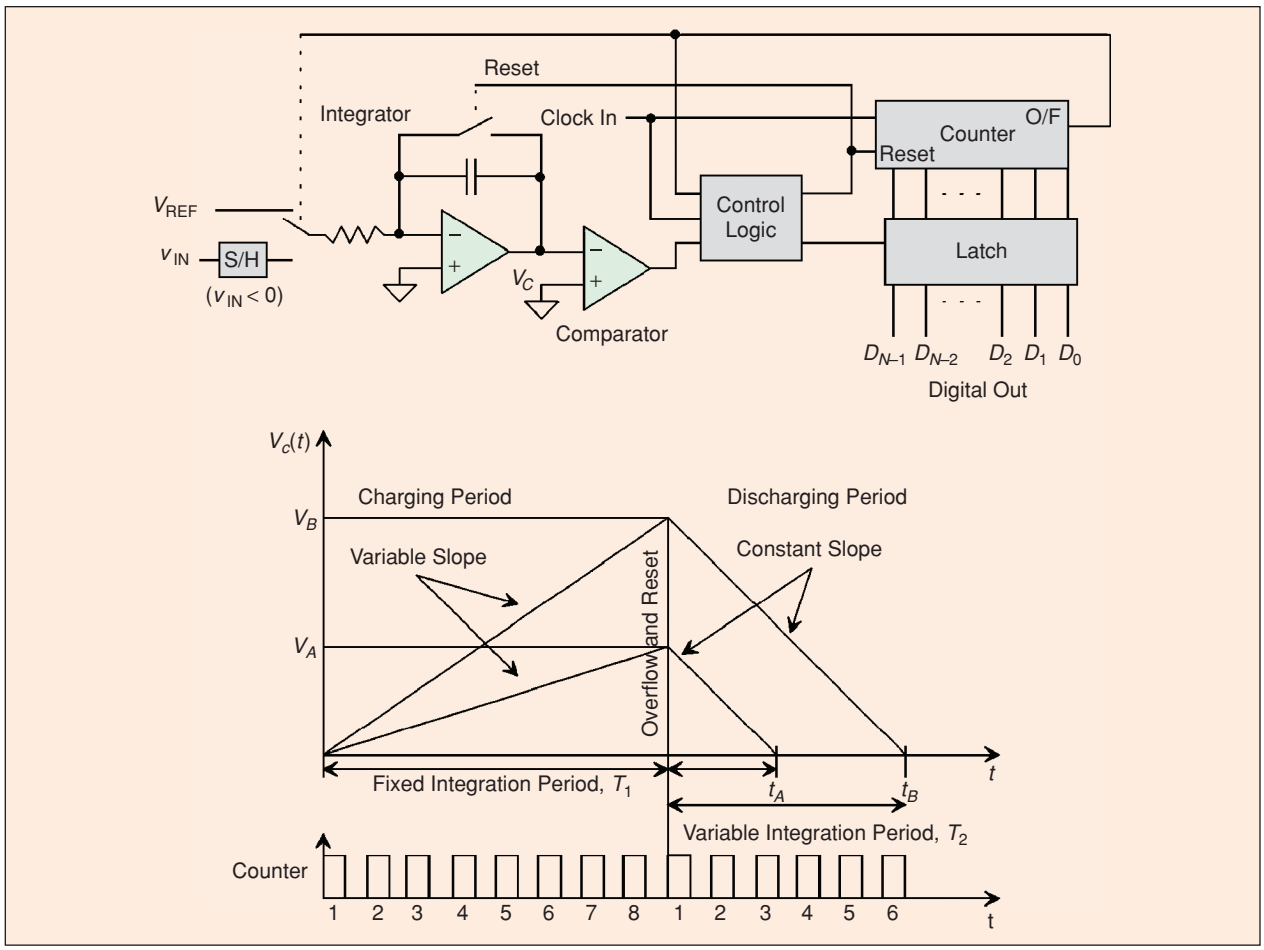

Fig. 9. Typical dual-slope (integrating) converter block diagram.

because of low cost, simplicity, relatively high speed, good resolution, and small footprint. A binary search is conducted through all the possible quantization levels before converging on and outputting the one that is closest to the input sample. The successive approximation converter, illustrated in Figure 10, does this search by using a shift register, S/H circuit, comparator, output register (successive approximation register or SAR), and a D/A converter (DAC). The DAC, as the name implies, is the counterpart to the ADC. A DAC takes a B-bit binary input signal and converts it into an analog output signal. The easiest way of understanding the successive approximation converter is algorithmically [3].

1) A 1 is applied to the shift register input. For each bit converted, the 1 is shifted to the right 1-bit position. $B_{N-1} = 1$ and B_{N-2} through $B_0 = 0$.
2) The MSB of the SAR, D_{N-1}, is initially set to 1, while the remaining bits D_{N-2} through D_0 are set to 0.
3) Since the SAR output controls the DAC and the SAR output is 100 . . . 0, the DAC output will be set to $V_{REF}/2$.
4) Next, v_{IN} is compared to $V_{REF}/2$. If $V_{REF}/2$ is greater than v_{IN}, then the comparator output is a 1 and the comparator resets D_{N-1} to 0. If $V_{REF}/2$ is less than v_{IN}, the comparator output is a 0 and the D_{N-1} remains a 1. D_{N-1} is the actual MSB final digital output code.
5) The 1 applied to the shift register is then shifted by one position so that $B_{N-2} = 1$, while the remaining bits are all 0.
6) D_{N-2} is set to a 1, D_{N-3} through D_0 remain 0, while D_{N-1} remains the value from the MSB conversion.

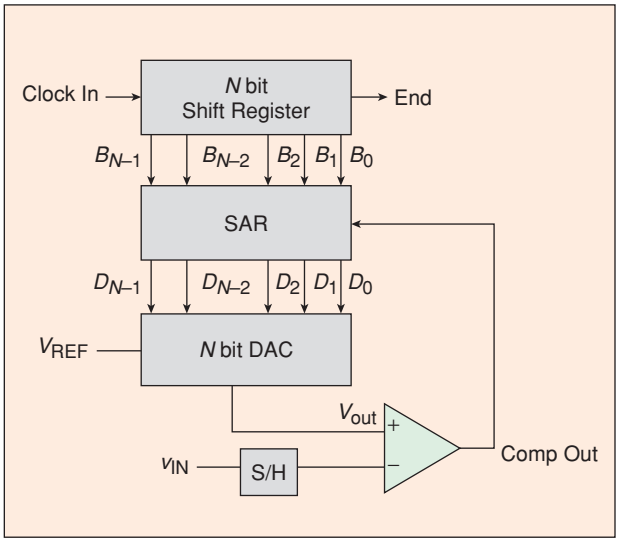

Fig. 10. Typical successive approximation converter block diagram.

The output of the DAC will now either equal $V_{REF}/4$ (if $D_{N-1} = 0$) or $3V_{REF}/4$ (if $D_{N-1} = 1$) and the comparison repeats.

This method is identical to an ideal game of "high or low" in which the most efficient strategy is employed to guess a random number between 1 and 100. You would start by guessing 50. If the other player said, "high'' you would know that the unknown number must be in the upper half of the range, and for "low'' you would know that the number must be in the lower half. You would continue subdividing each new range evenly until you arrived at the exact number (working your way through a binary tree). The number of steps involved is proportional to the resolution of the converter, and the accuracy is dependent on the accuracy of the DAC. If the DAC fails to produce an accurate voltage early in the conversion, a large error will result as the converter attempts to look for the correct quantization level in the wrong half of the binary tree.

What's Next

ADCs are so important to instrumentation that they deserve a closer look. In the next installment of the tutorials, the fundamentals of testing converters will be presented. In the meantime, let us know what you think. Are the current topics useful? What else would you like to see as a tutorial?

Oversampling

The final architecture is the newest and by far the most complex and is typically found in relatively low-speed, high-resolution applications like digital audio. All of the architectures we've previously discussed are called sampling converters because they sample near the Nyquist frequency (twice the maximum frequency component of the input signal). Oversampling converters, as the name implies, sample at a rate much higher than the input signal bandwidth and are used when high resolutions are required. Oversampling converters bypass the problems of the other converter architectures by using digital processing techniques in place of complex analog components. Also, because of the higher sampling rates, only basic antialiasing circuitry needs to be used and no S/H is required. The input signal bandwidth is so much lower than the sampling frequency that a simple single-pole low-pass filter will prevent aliasing in most applications. Oversampling converters were once slower than most sampling converters, but in recent years, their speed has improved with the advent of better mixed-signal manufacturing processes.

Two components make up an oversampling converter: a digital filter and a delta-sigma ($\Delta - \Sigma$) modulator [also called a sigma-delta ($\Sigma - \Delta$) modulator]. The $\Delta - \Sigma$ modulator provides quantization in the form of a pulse-density modulated signal. The density of the pulses represents the average value of the signal over a specified period. The more pulses per period (samples), the higher the resolution of the converter. The inner workings of a $\Delta - \Sigma$ modulator are very complex; only the fundamentals of operation depicted in Figure 11 are described.

The oversampled analog signal goes through an integrator whose output drives a comparator (a 1-bit ADC) that, in turn, drives a 1-bit DAC in a feedback loop. Through a series of iterations, the integrator, comparator, DAC, and summing junction produce a serial bitstream that represents the oversampled input voltage. Once digitized, the oversampled signal goes through a digital filter to remove frequency components at or above the Nyquist frequency. A decimator then removes the oversampled data (e.g., if the input is sampled at 64 times the Nyquist frequency, 1 bit will be retained for every 64 received). The final output is a B-bit serial stream (for a B-bit converter). One big advantage of the oversampling architecture is that the quantization noise is pushed out of the signal bandwidth. This process of noise shaping allows the converter to obtain very high resolutions. If the order of the modulator is increased, the noise is pushed farther away from the input signal spectrum, allowing even higher resolutions but at the expense of exponentially increasing circuit complexity. The accuracy achieved by oversampling converters and their throughput makes them popular choices.

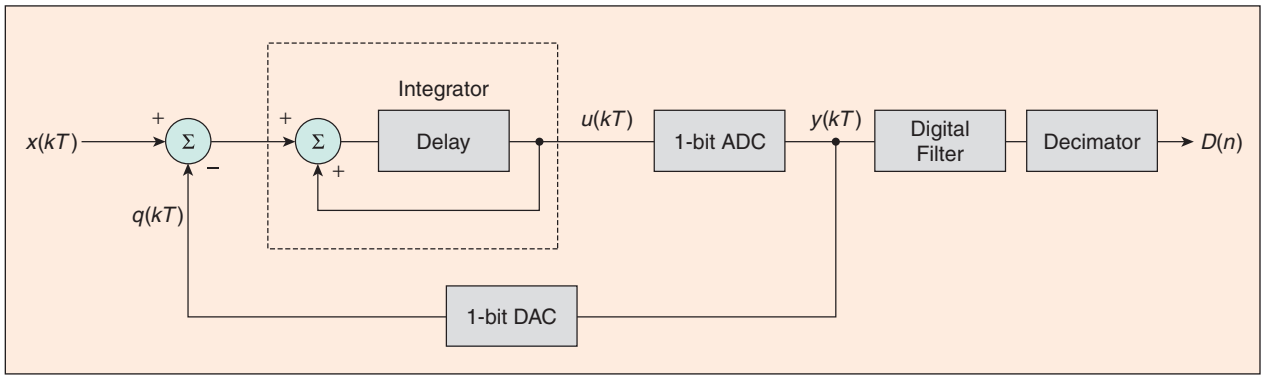

Fig. 11. First-order $\Delta - \Sigma$ converter.

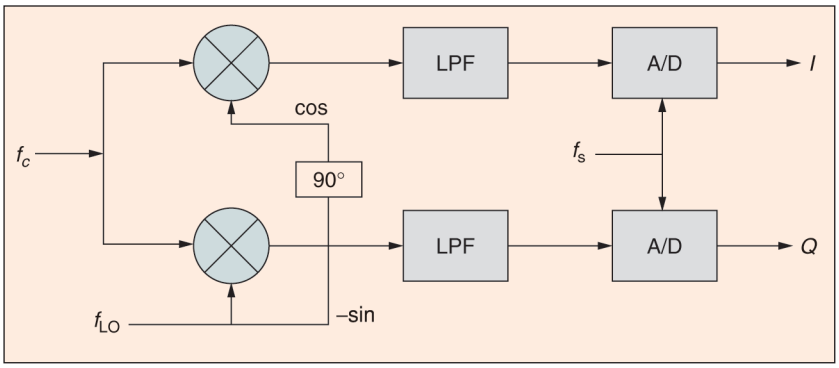

Fig. 12. A typical hybrid receiver.

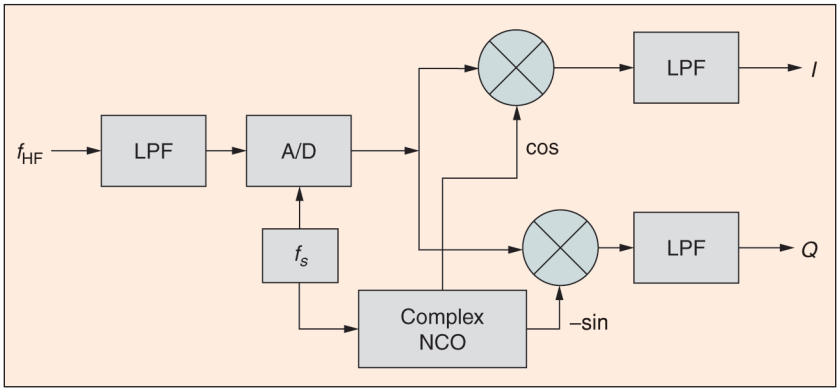

Fig. 13. A pure digital receiver.

Example Application: Bits Versus Bandwidth

A digital receiver design highlights two of the tradeoffs that often need to be made in ADC applications. From the preceding discussions, it is apparent that the availability of more bits means that better resolution is available along with improved dynamic range. Similarly, A/D architectures that support higher numbers of bits imply longer conversion times or higher clock rates. The designer of software radio front ends must strike a balance between these two competing parameters. If the A/D has fewer bits, then the sampling speed may be faster and the resulting bandwidth can be greater. These two fundamental ADC parameters directly affect the tradeoffs between receiver sensitivity and receiver selectivity.

Bits

Figure 12 shows a portion of a digital receiver with a quadrature sampling mixer [6]. This type of receiver uses two ADCs to sample a quadrature signal. Because the narrowband signal f_c, is sampled downstream from the analog mixers and low pass filters, the ADCs can effectively utilize more bits per sample. The result of having a higher dynamic range is better receiver sensitivity.

Bandwidth

Figure 13 shows a portion of a digital receiver that uses digital mixing [7]. This type of receiver architecture uses a single ADC as close to the antenna as possible in the signal processing chain to sample the whole HF band (30 Mhz and below). From the Nyquist criteria, the ADC sampling frequency f_s, has to be at least 60 Mhz. Improving the bandwidth of the digital receiver requires faster ADCs; the designer must contend with fewer bits in order to achieve the desired sampling rate and the associated receiver selectivity.

References

[1] J. Schmalzel and D. Rauth, "Tutorials on instrumentation and measurement: Part 4," *IEEE Instrum. Meas. Mag.*, vol. 8, no. 2, pp. 48–53, Mar. 2005.

[2] S. Orfanidis, *Introduction to Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[3] R. Baker, *CMOS: Circuit Design, Layout, and Simulation*. Piscataway, NJ: IEEE Press, 2005.

[4] K. Fowler, "Part 7: Analog-to-digital conversion in real-time systems," *IEEE Instrum. Meas. Mag.*, vol. 6, no. 3, pp. 58–64, Sept. 2003.

[5] S. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters*. Piscataway, NJ: IEEE Press, 1997.

[6] Flex Radio SDR-1000 datasheet [Online]. Available: http://www.flex-radio.com

[7] RFSpace SDR-14 datasheet [Online]. Available: http://www.rfspace.com

David A. Rauth (rauth@ieee.org) is an electrical and computer engineering graduate student working with Dr. John Schmalzel at Rowan University, Glassboro, New Jersey. His research interests include smart sensors, health management for integrated systems, and embedded vision systems.

Vincent T. Randal is a staff firmware engineer with Maxtor Corp. in Longmont, Colorado. He received his B.S.E.E. in 1985 and his M.S.E.E. in 1995, both from the University of Texas at San Antonio. His interests include embedded systems and software receiver development.