DIGITAL MEETS ANALOG

CHAPTER 13

Here we meet the major subject of conversion between analog and digital signals – analog-to-digital (A/D) and digital-to-analog (D/A) converters (ADCs and DACs) – as well as the important "mixed-signal" phase-locked loop (PLL). And we cannot resist a look at the fascinating topic of pseudorandom noise generation.

We live in a largely analog (continuous) world – of sounds, images, distances, times, voltages and currents, and so on – which would seem to call for analog circuits (oscillators, amplifiers, filters, combiners, etc.). But we live also in a partly digital (discrete) world – of numbers and arithmetic, of text and symbols, and the like – which would seem to call for digital circuits (arithmetic logic and storage, etc.). And that's how it was, for many years: analog amplifiers and filters for audio and video; analog oscillators, tuned circuits, and mixers for radio and television; and even *analog computers*, for solving differential equations[1] or for real-time control of flight or weaponry. Meanwhile digital techniques (initially with mechanisms and relays, then with vacuum tubes, followed by discrete transistors, small-scale ICs, and finally the large and fast microprocessors with a billion+ transistors that we take for granted) were used for computational tasks like keeping track of money, and of words.

But the almost miraculous improvements in the speeds and densities of purely digital electronics have produced a major paradigm shift, namely, the use of digital conditioning and processing for nearly every "analog" quantity. For example, audio engineers now digitize the individual microphone signals at the time of recording, and perform all subsequent mixing and conditioning (e.g., the addition of reverberation) as arithmetic on those numbers; the same goes for digital video. And at the everyday level, digital techniques have invaded our lives: the authors' bathroom scales indicate to 0.1 pound (sometimes to our regret) – that's a resolution[2] of a part per thousand; our porch light is switched on and off by a digital wall switch that follows the seasonal variation of dusk and dawn; and our automobiles depend on a digital bus, to which are connected some 50 or more embedded digital controllers for functions like engine control and diagnostics, braking, air bags, entertainment, climate control, and so on.

The bottom line is that A/D and D/A conversion techniques have become central to every aspect of analog measurement and control. This is important stuff, and it is the major subject of this chapter. Let's go at it.

Our treatment of the various conversion techniques is not aimed at developing skill in converter design itself. Rather, we try to point out the advantages and disadvantages of each method, because in most cases the sensible thing is to buy commercially available chips or modules, rather than to build the converter from scratch. An understanding of conversion techniques and idiosyncrasies will guide you in choosing from among the thousands of available units.

13.1 Some preliminaries

13.1.1 The basic performance parameters

Before getting into lots of detail, we'd like to summarize the important performance parameters that you need to keep in mind when choosing ADCs and DACs. Knowing what you need makes it a lot easier to find what you want.

Digital-to-analog converters

Resolution: number of bits
Accuracy: monotonicity; linearity; dc stability
Reference: internal or external; multiplying DAC (MDAC)?
Output type: voltage output or current output
Output scaling: unipolarity or bipolarity; V_{out} ranges; I_{out} compliance
Speed: settling time; update rate

[1] There's a nice example of this in the section on Analog Function Circuits in Chapter *4x*: modeling the fascinating chaotic behavior of the system of nonlinear differential equations devised by Lorenz.

[2] To be distinguished from *accuracy* – recall the discussion in §5.1.1. Although the bathroom scale reads with a *resolution* of 0.1 pound, its accuracy is likely poorer (perhaps to ±1 pound), with some drift over time and temperature.

Quantity: single or multiple DACs/pkg

Digital input format: serial (I^2C, SPI, or a variant) or parallel

Package: module, through-hole, or various surface-mount packages

Other: glitch energy; power-on state; programmable internal digital scaling

Analog-to-digital converters

Resolution: number of bits

Accuracy: monotonicity; linearity; dc stability

Reference: internal or external

Input scaling: unipolarity or bipolarity; voltage range

Speed: conversion time and latency

Quantity: single or multiple ADCs/pkg

Digital-output format: serial (I^2C, SPI, or a variant) or parallel

Package: module, through-hole, or various surface-mount packages

Other: internal programmable gain amplifier (PGA); spur-free dynamic range (SFDR)

13.1.2 Codes

At this point you should review §10.1.3 on the various number codes used to represent signed numbers. Offset binary and 2s complement are commonly used in A/D conversion schemes, with sign-magnitude and Gray codes also popping up from time to time. Here is a reminder:

	Offset binary	*2s Complement*
+Full scale	11111111	01111111
+Full scale−1	11111110	01111110
↓	↓	↓
0+1 LSB	10000001	00000001
0	10000000	00000000
0−1 LSB	01111111	11111111
↓	↓	↓
−Full scale+1	00000001	10000001
−Full scale	00000000	10000000

13.1.3 Converter errors

The subject of ADC and DAC errors is a complicated one, about which whole volumes could be written. According to Bernie Gordon at Analogic, if you think a high-accuracy converter system lives up to its claimed specifications, you probably haven't looked closely enough. We won't go into the application scenarios necessary to support Bernie's claim, but it's worth a first look at the four most common types of converter errors: offset error, scale error, nonlinearity, and nonmonotonicity, nicely illustrated in the self-explanatory Figure 13.1. Rather than boring you with a long-winded discussion, though, we'll move directly to a description of D/A converter techniques and capabilities. Then we'll revisit the business of converter errors (§13.4), which will make a lot more sense in context.

13.1.4 Stand-alone versus integrated

Sometimes an ADC or DAC (or both) is integrated into a fancier IC. The most common example is the microcontroller (Chapter 15), where you frequently see both ADCs and DACs integrated on the same chip as the processor and its other I/O peripherals. As far as we can tell, the least-expensive stand-alone ADC costs significantly more than the least-expensive microcontroller-*with*-ADC.[3] Microcontrollers are fond of integrating lots of useful peripherals, along with program and data memory, so that you've got essentially a "system-on-a-chip." Be aware, though, that these converters that come bundled with inexpensive general-purpose microcontrollers do not attain the excellent performance of a good stand-alone converter: you can get 8-bit or even 10-bit performance; but you won't get 16 bits, and nothing approaching the 24-bit performance of a high-quality audio ADC.[4]

For some classes of IC, though, an integrated converter delivers excellent performance. One example is a direct digital synthesis (DDS) chip (§7.1.8), where on-chip phase counters and a sine lookup table create digital values of the synthesized sinewave output; these things can clock at speeds of 1 GHz or more, with an on-chip 14-bit (for example) DAC generating the analog output signal. Another example comes from the video world, where it's common to see digital video processing and conversion functions combined on a single high-performance IC. And in the audio business you see parts like the Cirrus CS470xx-series (their "All-In-One Audio IC" system-on-a-chip), which includes multiple 24-bit ADCs and DACs with 105 dB dynamic range, integrated onto a chip that has a 32-bit DSP (with 32 kB RAM), audio codecs and sample rate

[3] To wit: National's ADC0831 8-bit ADC costs $1.85, whereas Microchip's PIC10F with its 8-bit ADC and 2-input multiplexer costs $0.48 (both in quantity 25).

[4] A shining exception is provided by Analog Devices' series of "Analog Microcontrollers," with honest performance to 16 or 24 bits. You might think of these as consisting of a high-quality converter core, with a quiet microcontroller tacked on.

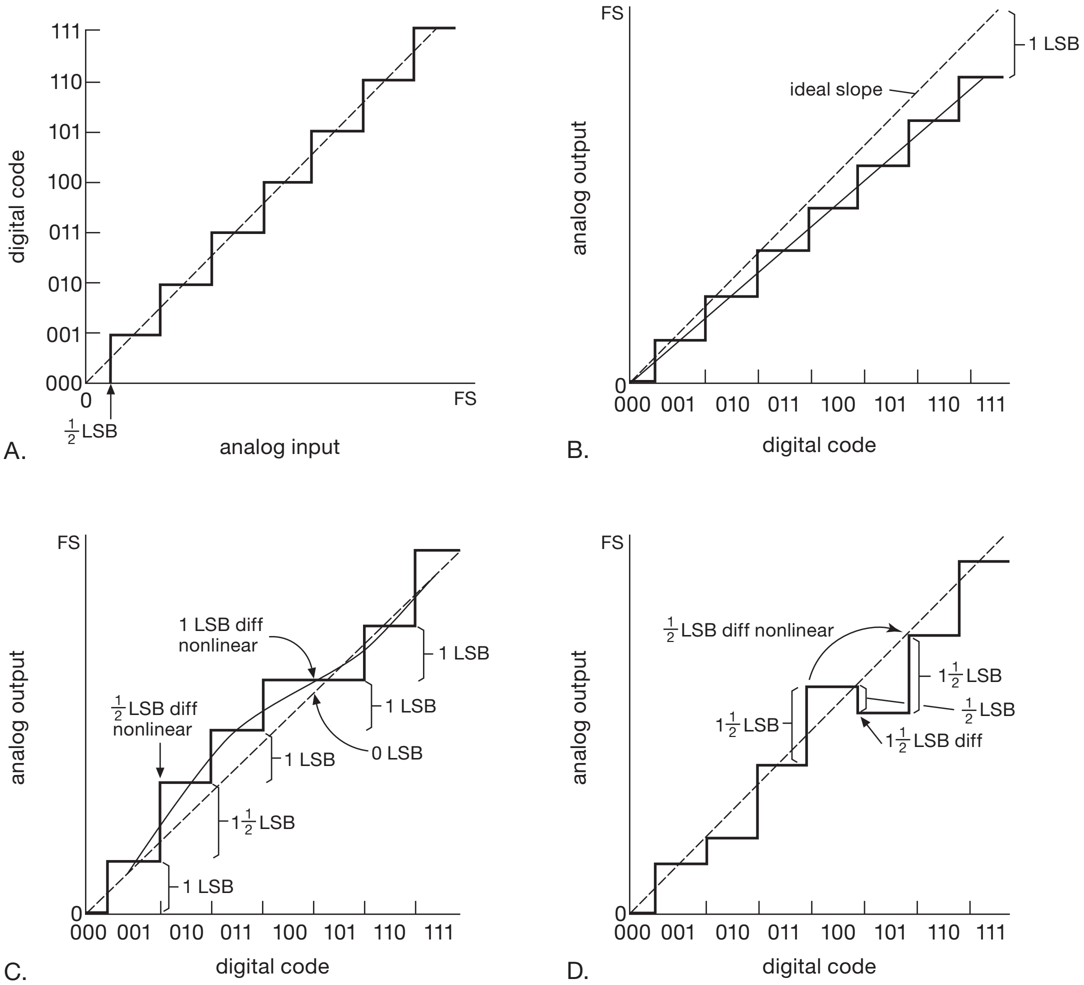


Figure 13.1. Graphs illustrating the definitions of four common digital conversion errors, for a 3-bit converter over its 8 levels from 0 to full scale (FS). A. ADC transfer curve, $\frac{1}{2}$LSB offset at zero. B. Linear, 1 LSB scale error. C. $\pm\frac{1}{2}$LSB nonlinearity (implies 1 LSB possible error); 1 LSB differential nonlinearity (implies monotonicity). D. Nonmonotonic (must be $>\pm\frac{1}{2}$LSB nonlinear). Used with permission of Texas Instruments Inc.

converters, digital audio ports (SPDIF), and an SPI/I^{2}C control port.

Stand-alone converters are dominant, though, in high-accuracy and high-linearity applications (voltmeters; quality audio gear). They also provide a tremendous selection range, in terms of the many parameters just listed, as compared with the rather limited selection of on-chip converters you find in microcontrollers.

13.2 Digital-to-analog converters

The goal is to convert a quantity specified as a binary (or multidigit BCD, see §10.1.3B) number to a voltage, or to a current, proportional to the value of the digital input. There are several popular methods: (a) resistor string with MOS switches; (b) *R*–2*R* ladder; (c) binary-scaled current sources; and (d) delta–sigma (and other pulse-averaging) converters. Let's take them in turns.

13.2.1 Resistor-string DACs

This method is about as straightforward as you can get. A string of 2^n equal-value resistors is connected between a stable voltage reference and ground, creating a very tall voltage divider; and a set of MOSFET analog switches is used to route the selected tap's voltage to an output voltage buffer (Figure 13.2). The figure shows the configuration of TI's impressive DAC8564, a quad 16-bit DAC (four independent DACs in one package), each section having a string of 2^{16} (65,536) resistors connected between a precision internal +2.5 V reference and ground. Quoting from the terse description in the datasheet, "The code loaded into the DAC register determines at which node on the string the voltage is tapped off to be fed into the output amplifier by closing one of the switches connecting the string to the amplifier."[5]

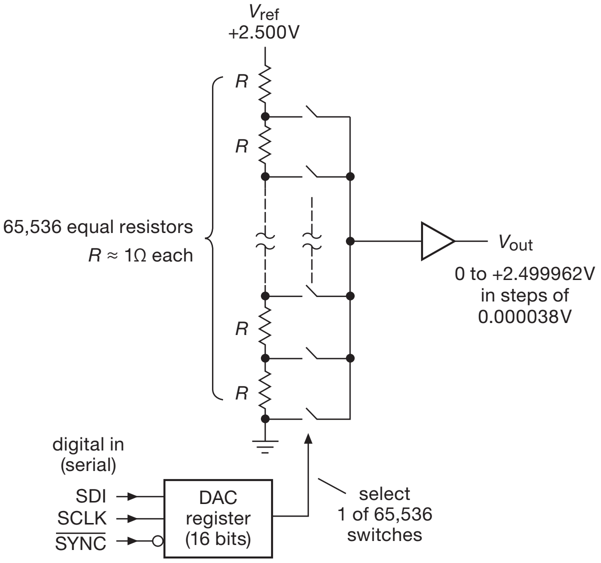


Figure 13.2. An easy-to-understand DAC: the digital input selects the corresponding MOSFET switch tap on a giant voltage divider. TI crams four of these into their DAC8564.

This method has the virtue of guaranteed monotonicity. As the datasheet puts it (even more tersely than before), "It is monotonic because it is a string of resistors." And this particular DAC exhibits other nice qualities, specifically low *glitch energy* (spikes that appear at the output during code transitions), excellent accuracy and stability (worst-case values of ±0.02% initial accuracy and 5 ppm/°C tempco), rail-to-rail output ("RRO") with a single positive supply (+2.7–5.5 V), and low power (1 mA, typ). This puppy costs about $12. The same method is used in DACs of more modest performance, for example National's DAC121: 12-bit, voltage output, micropower (150 μA) single-supply with rail-to-rail input and output (RRIO). The latter is a single DAC without internal reference (full scale is the positive supply), and of course it has "only" 4096 resistors in its string; it costs under $2. Both of these (and most other converters these days) use a serial digital input, which for these particular converters is the simple 3-wire SPI (see §14.7). See also the discussion of digital potentiometers in §3.4.3E.

13.2.2 *R*–2*R* ladder DACs

A string of 2^{16} matched resistors and switches is a pretty impressive piece of engineering. But the exponential number of components eventually overwhelms engineering finesse. An attractive alternative is the *R*–2*R* ladder, which collapses the requirement to an array of just 2*n* matched resistors (versus 2^n) for an *n*-bit DAC.

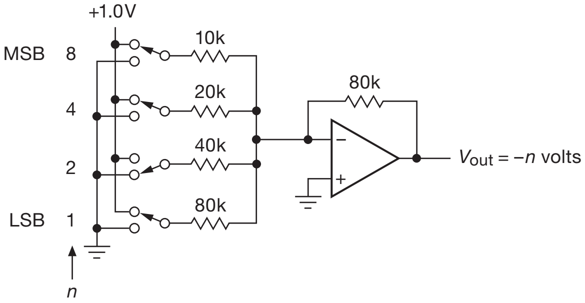


Figure 13.3. Summing scaled currents to create a simple DAC. Simple to understand, but never used in practice: an *R*–2*R* network is used instead.

To get started, consider the simple notional scheme shown in Figure 13.3: the resistor values are in a binary sequence, so their binary-weighted currents at the summing junction produce a binary-weighted voltage output. Simple, but not terribly practical with more than a few bits, because the resistor values must span a wide range, and with increasingly demanding accuracy for the lower resistance values; of greater worry still is the need for very low switch R_{on} corresponding to the low resistance values.[6]

Exercise 13.1. Design a 2-digit BCD DAC. Assume that the inputs are 0 or +1 volt; the output should go from 0 to 9.9 volts.

Instead, the scheme shown in Figure 13.4 is used. It's easy to convince yourself that this clever arrangement produces a binary-weighted current into the op-amp's summing junction, therefore a corresponding output voltage. And only two resistor values are needed (*R* and 2*R*, which, however, must both be accurately replicated and in a precise 2:1 ratio), regardless of the number of bits.

[5] These guys seem not to be fond of commas. We were tempted to add a few, to pace the breathless flow of the sentence. But, hey, a quote is a quote, right?

[6] This method does have the flexibility, however, of allowing arbitrary bit weights.

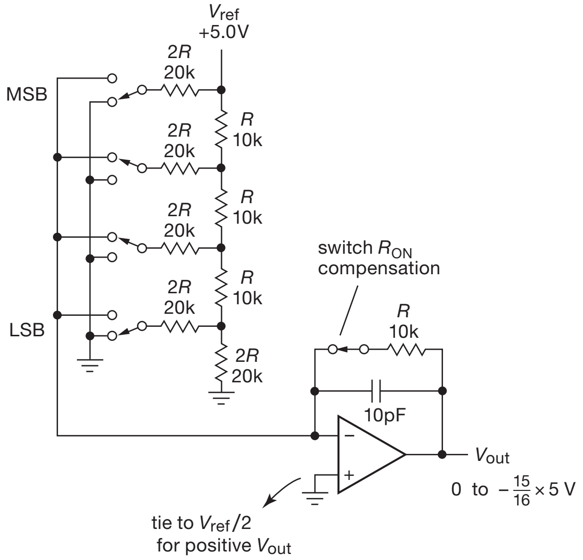


Figure 13.4. An *R*–2*R* ladder network generates a binary-scaled output current into the op-amp's summing junction, producing a voltage-output DAC.

There are many excellent *R*–2*R* DACs out there. For example, TI's DAC9881 is an 18-bit voltage-output DAC with an SPI serial input; it is guaranteed monotonic,[7] with integral linearity to ±2 LSB. It requires an external voltage reference (V_{ref}), which sets the full-scale voltage (the noninverting input is biased at $V_{ref}/2$, for positive output polarity). It excels in accuracy and low noise, and costs about $30.

Exercise 13.2. Prove that the foregoing *R*-2*R* ladder works as advertised.

In practice, most *R*–2*R* DACs use the alternative configuration of Figure 13.5, in which the output of the *R*–2*R* network is itself a voltage. For example, the TI DAC7611 is a 12-bit voltage-output DAC (+4.095 V full-scale output, popular for DACs that run from +5 V) with an SPI serial input and on-chip voltage reference; it's linear and monotonic to its full 12 bits, comes in an 8-pin package, and costs about $4.

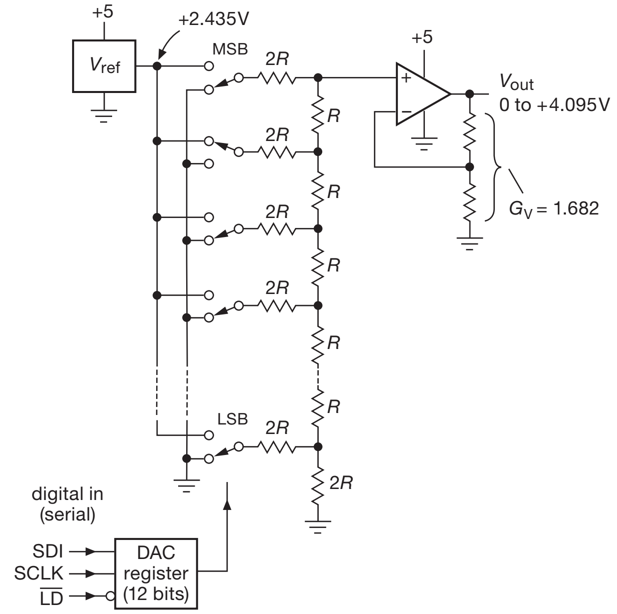


Figure 13.5. *R*–2*R* voltage-output DAC, in the more common voltage-combining configuration.

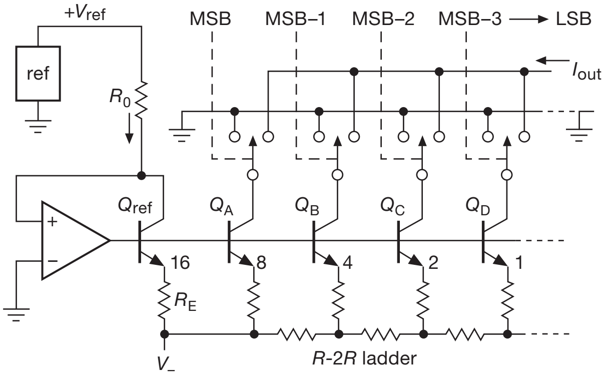


Figure 13.6. Classic current-switched DAC.

13.2.3 Current-steering DACs

The preceding converters generate *voltage* outputs. This is often most convenient, but the op-amp tends to be the slowest part of the converter circuit. In situations where you can use a converter with *current* output, you'll get better speeds, and usually at lower price. Some additional advantages of current-output DACs are: (a) flexible choice of current-to-voltage op-amp, for example to minimize noise, or to produce a larger output voltage swing; (b) the ability to combine several DAC outputs directly; and (c) the availability of *multiplying DACs* (see next subsection), in which

[7] Note that, unlike a resistor-string DAC, an *R*–2*R* DAC is not *guaranteed* monotonic when resistor tolerances are taken into account. The semiconductor industry does a good job, however, and most *R*–2*R* DACs are monotonic to 1 LSB.

the output current is the product of the digital input code and an analog signal applied to the V_{ref} input.

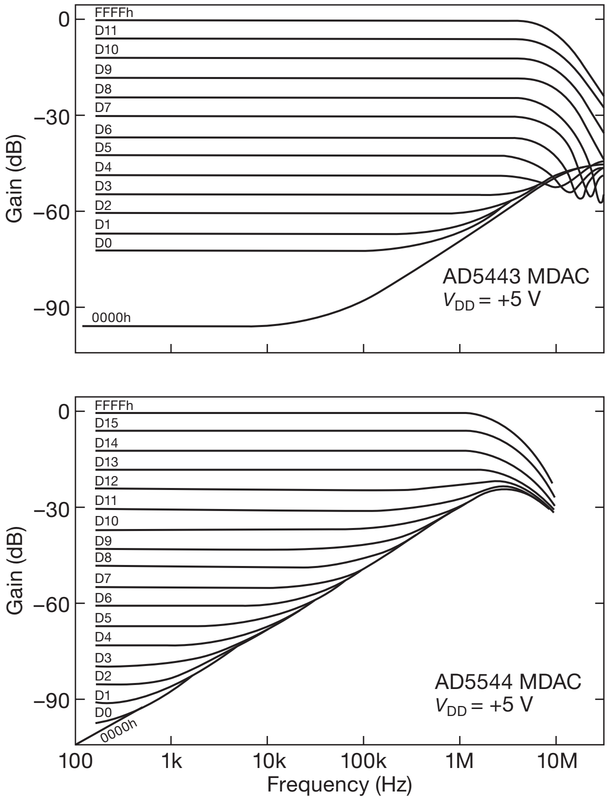


Figure 13.7. An MDAC's analog "reference multiplying bandwidth" listed in the datasheet is usually specified for the maximum digital input code only; for these Analog Devices MDACs those values are 2 MHz and 10 MHz. These graphs, found later in the datasheets, tell the whole story. Although the AD5544 and AD5443 are similar designs with similar part numbers, the latter has considerably better capability in the 0 dB to −40 dB region out to 10 MHz.

Figure 13.6 shows how these "current-steering" converters work. The currents can be generated by an array of transistor current sources with scaled emitter resistors, although IC designers usually use instead an *R*–2*R* ladder of emitter resistors. In most converters of this type, the current sources are ON all the time, and their output current is switched to the output terminal or to ground, under control of the digital input code. Watch out for limited output compliance in current-output DACs; it can be as little as 0.5 V, though values of a few volts are typical.

Some examples of current-steering DACs (with serial digital inputs and internal voltage references) are the LTC1668 (16 bits, 20 ns settling time into a 50 Ω load as "voltage out," output compliance to ±1 V, about $20) and the TI DAC5682 (dual, 16 bits, 10 ns settling time, output compliance to $V_+ \pm 0.5$ V, about $45). For *real* speed, there's the AD9739 (14 bits, 2500 Msps!). On a more modest scale there's the industry-standard DAC/LTC8043 (external voltage reference input, 12 bits, 0.25 μs settling time into 100 Ω, about $6), and the similar parallel-input AD/LTC7541.

13.2.4 Multiplying DACs

Note that these latter two converters require an external voltage reference, an apparent disadvantage that can be turned into an advantage: they accept a continuous range of V_{ref} input voltages, *of either polarity*. In other words, the (current) output is proportional to the product of the digital input and the analog reference voltage: it's a "multiplying DAC" (MDAC). Furthermore, the product can be positive or negative; so its full name is a "four-quadrant multiplying DAC." Examples of higher-resolution four-quadrant MDACs are the 16-bit DAC8814 (serial input) and closely similar DAC8820 (parallel input). Multiplying DACs specify both their conversion precision (linearity, monotonicity) and the bandwidth of the analog multiplying input (i.e., V_{ref}); for these two converters the "reference multiplying bandwidths" are 10 MHz and 8 MHz, respectively, with respective prices of $25 and $15.

Note that not all DACs are optimized for use in this way, so it is best to check the datasheets of the converters you're considering for details. A DAC with good multiplying properties (wide analog input range, high speed, etc.) will usually be called a "multiplying DAC" right at the top of the datasheet. Table 13.3 on page 894 lists selected Multiplying DACs.

A caution: the specified bandwidth can be seriously misleading, owing to the effects of capacitive feedthrough. Figure 13.7 shows this behavior, as illustrated in the respective datasheets of these two MDACs.

Multiplying DACs (and the ADC equivalent) open the possibility of *ratiometric* measurements and conversions. If a sensor of some sort (e.g., a variable-resistance transducer like a thermistor) is energized by the same voltage that also supplies the reference for the ADC, then variations in the reference voltage will not affect the measurement. This concept is very powerful, because it permits measurement and control with accuracy greater than the stability of voltage references or power supplies; conversely, it relaxes the requirements on supply stability and accuracy. The ratiometric principle is used in its simplest form in the classic *bridge* circuit, in which two ratios are adjusted to equality by nulling the differential signal taken

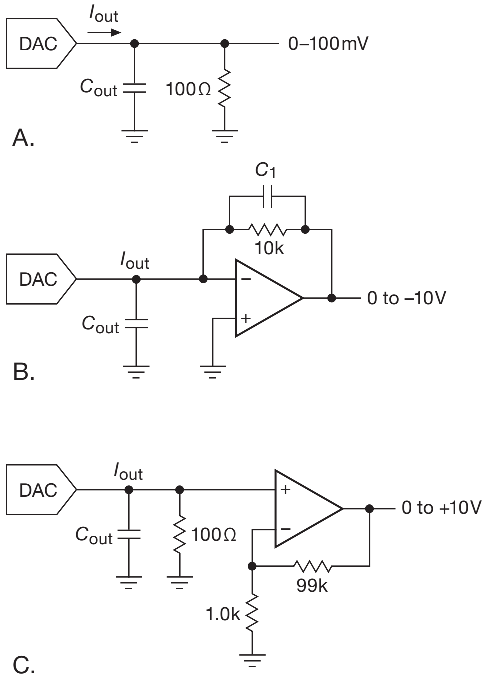


Figure 13.8. Generating voltages from current-output DACs.

between the two voltage-divider outputs. Devices like the 555 (see §7.1.3) achieve good stability of output frequency with large variations of supply voltage by using essentially a ratiometric scheme: the capacitor voltage, generated by an *RC* network from the supply, is compared with a fixed fraction of the supply voltage ($\frac{1}{3}V_{CC}$ and $\frac{2}{3}V_{CC}$), giving an output frequency that depends only on the *RC* time constant.[8] We will have more to say on this important subject in connection with ADCs later in this chapter.

13.2.5 Generating a voltage output

If you've chosen a voltage-output DAC, you're done! But with a current-output DAC you've got to use one of several schemes for generating a voltage output. Figure 13.8 shows some ideas. If the load capacitance is low and large voltage swings aren't needed, a simple resistor to ground will do nicely (but see warning, below). This is what's usually done with video DACs. For example, the THS8133 triple 10-bit video DAC generates full-scale output currents of 26.7 mA, which produce a standard 1.0 V analog video signal when driving doubly terminated 75 Ω coax. This method works fine also for general applications: with the usual 1 mA full-scale output current, a 50 Ω load resistor will give 50 mV full-scale output with 50 Ω output impedance. If the capacitance of the DAC's output combined with the load capacitance doesn't exceed 100 pF, you will get 50 ns settling time in the preceding example, assuming the DAC is that fast. When worrying about the effect of *RC* time constants on DAC output response, don't forget that it takes quite a few *RC* time constants for the output to settle to within $\frac{1}{2}$ LSB of the final voltage. It takes 7.6 *RC* time constants, for instance, to settle to within 1 part in 2048, which is what you would want for a 10-bit converter output.

To generate large swings, or to buffer into small load resistances or large load capacitances, an op-amp can be used in the transresistance configuration (current-to-voltage amplifier), as shown. The capacitor across the feedback resistor may be necessary for stability, because the DAC's output capacitance in combination with the feedback resistance introduces a lagging phase shift; unfortunately, that compromises the speed of the amplifier. It is an irony of this circuit that a relatively expensive high-speed (fast-settling) op-amp may be necessary to maintain the high speed of even an inexpensive DAC.[9] In practice, the last circuit may give better high-speed performance, since no compensation capacitor is needed. Watch out for offset voltage error, because the op-amp's input offset voltage is amplified by the voltage gain (here a factor of 100).

An Important Warning: when using current-output DACs, note that both the initial accuracy (e.g., full-scale I_{out}) and the stability of the current output may be grotesquely poor, relative to the DACs resolution. It is not uncommon to see a spread of as much as 2:1 (!) in the full-scale current. What to do? Most current-output DACs include a built-in feedback resistor, closely matched to the *R*–2*R* resistors, intended to be used with an external op-amp (Figure 13.9). If you don't use it, you may have gain errors of ±25% or more; and even if you trim away this gain error, there will be residual gain *drift* (not usually specified on datasheets) that typically will be 100 times larger than you get with the internal resistor.

To give an example, the LTC8043 is an improved version of the industry-standard DAC8043 12-bit multiplying

[8] We showed analogous ratiometric schemes in §§4.3.5, 4.6.4, 7.1.3D, and 10.4.5A.

[9] Some current-output DACs have surprisingly-high output capacitance, C_{out}, e.g., as much as 200 pF for the LTC7541 12-bit MDAC. So you need a stabilizing capacitor C_1, of value $C_1 > \sqrt{C_{out}/2\pi f_T R_f}$, as discussed before, see for example §§8.11.6 and *4x.3*. Choose an op-amp, then, with a high enough f_T so that C_1 is small enough for the desired speed. Some fast current-output DACs take pains to keep C_{out} small, as little as 5 pF.

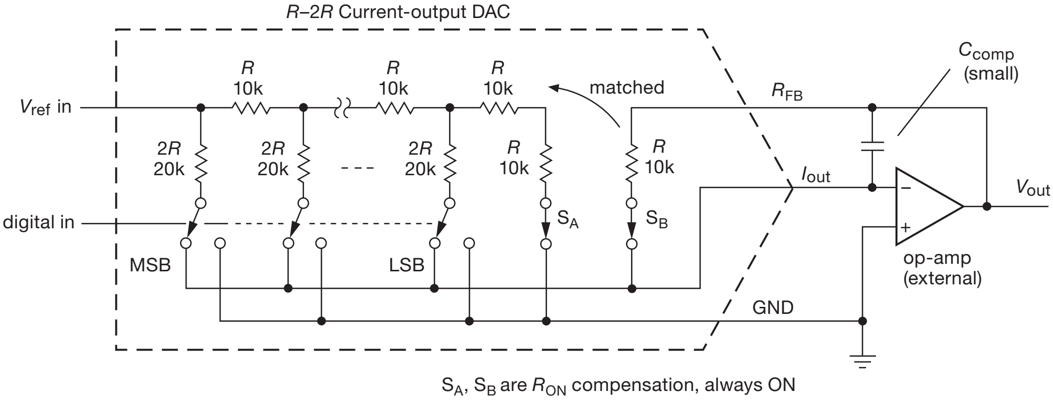


Figure 13.9. Current-output *R*–2*R* DAC with internal feedback resistor matched to the precision network resistors both in initial resistance and in temperature coefficient. Ignore R_{FB} at your peril!

DAC. In the best grade (-E suffix) it specifies a gain error of ±1% (max), and a gain tempco of 5 ppm/°C (max). (It also guarantees maximum integral and differential nonlinearities of ±0.5%; see the discussion in §13.4.) Impressive specs. Note, however, that the gain specifications state "Using internal feedback resistor." What if you don't? The datasheet doesn't say! But you can tease the answer from what it *does* say, namely that the input resistance of the *R*–2*R* network (seen at the V_{ref} input) is 11 kΩ (nominal), with limits of 7 kΩ (min) and 15 kΩ (max). That is, this DAC, which guarantees an admirable gain accuracy of ±0.024% when the matched internal feedback resistor is used in conjunction with an external op-amp, would deliver a shockingly poor[10] ±35% absolute gain error as a current-output device (or, equivalently, with an external feedback resistor and op-amp, for voltage output).[11]

Bottom line: if a current-output DAC gives you an internal feedback resistor, you need to think long and hard before deciding not to use it.

Another bottom line: if you want a bipolarity output-voltage range, you might be tempted to sink a reference current (derived from V_{ref}) from the summing junction in Figure 13.9. *Don't do it!* Instead, append a difference amplifier to V_{out}, using an offset of $V_{ref}/2$ for the other input.

13.2.6 Six DACs

To give a further sense of what's out there, let's look at a few examples of relatively simple DACs of modest performance. By "modest" we mean that these converters do not push close to the limits of speed or accuracy; rather, they're inexpensive, compact, and easy to use. You can drop them onto a circuit board, and off you go. Later, in §13.3, we'll look at some applications that demand DACs of greater performance; in those situations you'll need carefully designed surrounding circuitry to exploit fully the advanced capabilities of the converter.

Take a look at Figure 13.10, with reference to Table 13.1. These were chosen somewhat arbitrarily from the many thousands (no kidding!) of available DACs, though we did make an effort to select devices from multiple manufacturers. They range from 8 to 14 bits, with settling times of 4–10 μs. Except for E, they are all voltage-output devices.

Taking it from the top of Figure 13.10: A–C are single-supply low-voltage parts with serial interfaces, so they can fit into tiny SOT23 or (even tinier) SC70 packages. The pair in A use the supply voltage as reference, with the full-scale output voltage V_{FS} equal to the supply V_{DD}; one uses an SPI interface (with the usual chip select), whereas the other uses an I^{2}C interface (with the input pin A_0 selecting between factory preset bus addresses of decimal 76 or 77).[12] If you want a fixed and stable reference voltage, rather than using V_{DD}, a converter like the LTC2630 in B is a good choice. Its internal reference has adequate

[10] I'm shocked, shocked to find that gain inaccuracy and drift is going on in here!

[11] You might ask why an IC manufacturer whose chips deliver outstanding accuracy has a problem making resistors with even "pretty good" absolute accuracy. Good question. Turns out that the process is optimized for best *tracking*, with overall resistor scaling only of secondary importance.

[12] The DAC7512 from the same manufacturer substitutes SPI for I^{2}C; and the AD5601/11/21 are less-expensive 8- to 12-bit versions of the 14-bit AD5641.

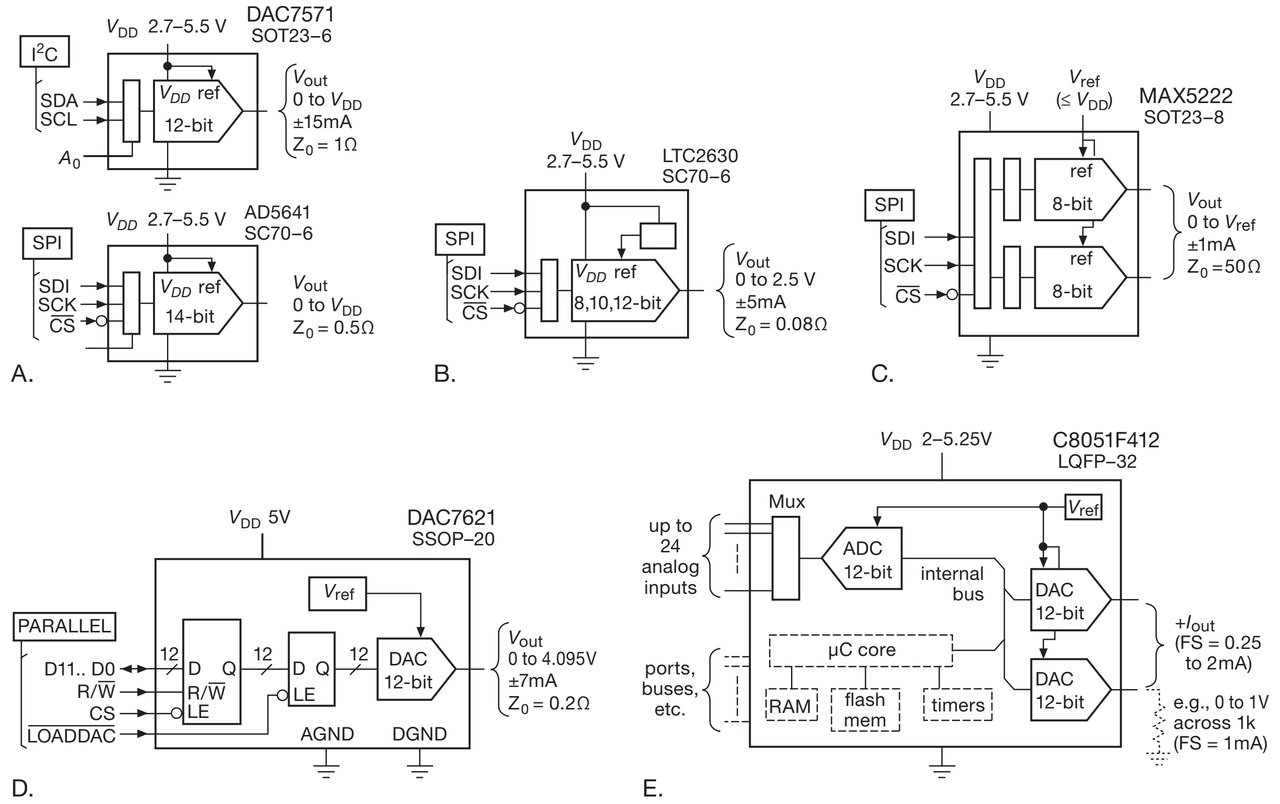


Figure 13.10. Six DACs, whose specifications are listed in Table 13.1. The DAC7571, LTC2630, and MAX5222 in the upper row are typical of inexpensive serial-input converters in small packages (see §14.7 for discussion of the SPI and I²C serial interfaces). The DAC7621 uses a traditional parallel data input; and the C8051 is a general-purpose microcontroller that includes a pair of current-output DACs among its many internal assets.

stability for a converter of this precision (±10 ppm/°C, typ), though its absolute accuracy is only modest (full-scale error of ±0.2% typ, ±0.8% worst case). The LTC2630 family includes a "high-voltage" variant with a 4.096 V internal reference (thus 1.0 mV/LSB), for which the V_{DD} range is narrowed to 4.5–5.5 V (5 V nominal); it also includes 8- and 10-bit variants.

Although you *could* imagine squeezing two converters into a 6-pin package (e.g., with an I²C 2-wire interface and no choice of address), the converter in C takes the rational choice of adding two pins, which both accommodates the SPI port (three wires, including CS′) and allows an external reference.[13] By reducing the lead pitch (from 0.95 mm to 0.65 mm) this 8-pin device fits in the same overall package size (1.6×2.9 mm) as the 6-pin SOT23-6.

The bottom row illustrates a converter with a parallel input port, an interface that has largely gone out of popularity except for converters that operate at the highest speeds (e.g., the 2500 Msps AD9739 in §13.2.3). But there are applications for which this is useful, even at modest speeds, for example if you want to use the *n*-bit output of a counter directly: no microcontroller, no programming… just wires.

Finally, the last "converter" E is actually just some carry-on baggage belonging to a full-function microcontroller, complete with on-chip program memory (flash ROM), SRAM, timers, ports (parallel, SPI, UART), accurate oscillator, and even a 24-channel (multiplexed) 12-bit 200 ksps ADC. The dual DACs use an internal reference, and produce a current-sourcing output (with programmable full-scale ranges of 0.25–2 mA, by factors of 2) with compliance to 1.2 V below the supply rail (which runs over a wide range of 2.0–5.25 V). Oh, and did we mention that this thing includes a computational core that runs at

[13] One small fly in the ointment here is the lack of double buffering along with an 8-bit data payload; thus the two channels cannot be updated to different values simultaneously.

50 million instructions per second? Or that it includes on-chip full-speed in-system debugging? Or that you can get it also in a diminutive 5 mm square QFN-28 package? Quite a performance!

13.2.7 Delta–sigma DACs

The final DAC technique is somewhat weird, and not so easily understood in all its richness. (See Table 13.11 on page 939.) We'll describe it in detail later in the chapter (§13.9). Very roughly speaking, the technique consists of generating a train of fixed-amplitude pulses, at a high clock rate, on a single output line. These pulses are all of the same width, and are either present or absent at each clocking interval, according to the digital input code. (You could imagine simply generating a regular train of pulses, with a duty cycle proportional to the input code; but the delta–sigma process is considerably more sophisticated, as explained in §13.9.) This pulse train is then lowpass filtered, with a cutoff well below the clock frequency, to generate the analog output.

This is sometimes called a "1-bit DAC." That's a seriously misleading name, though, because these things in fact deliver stunningly linear output signals of high resolution. They are widely used in professional audio. A nice example is the ADI AD1955 dual (stereo) ADC, which delivers 20-bit analog audio output (120 dB dynamic range) when clocked at 12 MHz.[14]

13.2.8 PWM as digital-to-analog converter

One last method of driving an analog system from a digital quantity is via *pulse-width modulation* (PWM). This is qualitatively different from the true DACs just described, because it does not generate an analog output directly; but it is widely used for power loads such as a heater. The idea is to run a repeating cycle of N clock cycles, during which the load is switched ON for some smaller number of clock cycles k, and OFF for the rest, with the fraction (duty cycle) k/N proportional to the digital input (Figure 13.11). This is easily done with a counter, magnitude comparator, and high-frequency clock (see Exercise 13.3). The slowly responding load does the averaging over the full cycle. This is more efficient than driving the load with a properly smoothed analog signal, because the driver is a switch, with very little dissipation; a switch is also simpler than a linear amplifier. This technique is popular in "class-D" audio power amplifiers (§2.4.1C) and in other power-control applications such as stepping motors and dc servomotors (§9.9). Many microcontrollers are configured with internal PWM timer modes; even lacking that, you can always program it in software.

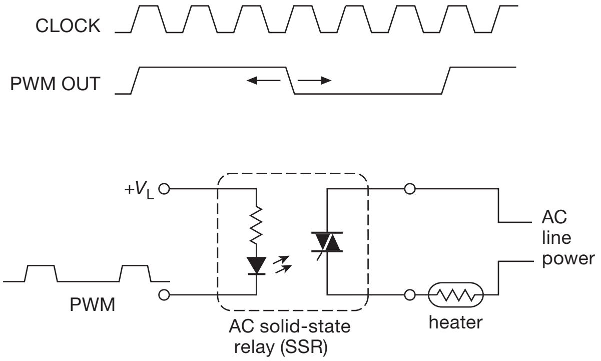


Figure 13.11. Pulse-width modulator (PWM) as time-averaged DAC, for slow power loads. For an ac-powered load (as shown) the clock should be synchronous with the powerline.

Although a simple lowpass filter could be used to generate an output voltage proportional to the average time spent in the HIGH state (i.e., proportional to the digital input code), pulse-width modulators are most often used when the load is itself a slowly responding system. The pulse-width modulator then generates precise parcels of energy, averaged by the system connected as a load. For example, the load might be capacitive (as in a switching regulator, see Chapter 9), thermal (a thermostated bath with heater), mechanical (a tape-speed servo, variable-speed motor, or stepping motor), or electromagnetic (a large electromagnet controller).

PWM outputs are attractive both for their simplicity and for their natural match to digital devices like counters and power-driving switches (MOSFETs); but there are some serious tradeoffs. For example, to get high PWM resolution for the k/N fraction, we need a large N. But the timer has a maximum clock rate f_{clk}, which sets a lower cycle rate $f_c = f_{\text{clk}}/N$. For a PWM that is within a feedback loop (as in the PWM discussion in §15.6) this implies reduced loop bandwidth and gain.

In practice you'll most likely encounter digital PWM hardware in a microcontroller. Sometimes you can choose a specific μC on account of its PWM capabilities, but more often not. For example, further along in §13.9.11B we choose TI's MSP430F2101 because it has an analog comparator. So, what kind of PWM does one get in the MSP430x2xx family? This information is not in the

[14] It does cheat, however, by generating several parallel 1-bit streams internally; it's a "multibit" delta–sigma ADC. More on this, and other delta–sigma amusements, in §13.9.

Table 13.1 Six Digital-to-Analog Converters

					Total V				Reference		Output				Cost
				t_{settle}	min	max	I_S				Z_{out}	I_{out}			qty 1
	Bits	Channels	Output	(μs)	(V)	(V)	(μA)	bus/Mbps	type	error[m]	(Ω)	(mA)	pkg	pins	$US
DAC7571	12	1	V	10	2.7	5.5	140	I^2C / 4.8	V_{DD}	0.2%	1	±15	SOT23	6	4.65
AD5641	14	1	V	6	2.7	5.5	75	SPI / 30	V_{DD}	0.04%	0.5	±5	SC70	6	5.40
LTC2630	12[a]	1	V	4.4	2.7	5.5	180	SPI / 50	2.5[c]	0.8%	0.08	±15	SC70	6	3.70
MAX5222	8	2	V	10	2.7	5.5	380	SPI / 25	ext	10mV	50	±1	SOT23	8	3.00
DAC7621	12	1	V	7	4.7	5.3	500	parallel[d]	4.096	0.4%	0.2	±7	SSOP	20	7.00
C8051F412	12	2	I	10	2	5.3	I_{out}	μC internal	2mA[e]	2%	CS	NA	LQFP	32	7.80

Notes: (a) 10 bit and 8 bits avail, $2.63. (b) 5V logic OK. (c) choice of V_{DD} or V_{ref}, 2.5V or 4.096V avail. (d) double buffered. (e) software select 0.25mA to 2mA, by factors of 2. (f) compliance to V_{CC}–1.2V. (m) max.

52- or 88-page datasheets; we had to consult the 693-page "MSP430x2xx Family Users Guide," where 40 pages were devoted to Timers A and B.

Figure 13.12 shows the MSP430F2002 member of this microcontroller family (which has a 10-bit ADC instead of a comparator) driving a dc torque motor. We've arranged four MOSFETs with an H-bridge driver and the controller's PWM signal to set the current. Naively, you'd expect that a 50% duty cycle would correspond to zero motor current; but that's true only when the motor is stopped, because the motor's "back EMF" disturbs this simple concept. Here we've used two sense resistors to measure the forward or reverse currents, with a pair of G=80 instrumentation amplifiers (§5.15) to inform the μC's ADC so it can servo the PWM to set the desired motor current and torque.[15] The amplifiers can operate from a single supply if their output reference pin is at least +0.8 V (see Table 5.8 on page 363); here we used a 1.25 V zener-type IC reference.

The microcontroller has a pair of 16-bit timers with programmable input selectors and dividers. They can run as fast as 16 MHz, which results in a cycle frequency of f_c=244 Hz if we use the full 16-bit resolution. You can program the length N to less than 2^{16}, but remember that the timer's other users have to live with your choice. Timer A has two capture/compare registers (CCR1 and CCR2), which can be used in compare mode to generate two PWM outputs (CCR0 is already used to set N). For example, let's say we need a speedier cycle rate of f_c=10 kHz, and we stick with the maximum 16 MHz clock. We set the counter's modulus N=f_{clk}/f_c=1600 counts, and our PWM resolution will be limited to… about 10 bits. My goodness, only 10 bits?[16]

This example illustrates that a favorable alternative to PWM may involve attaching a few external DACs to a microcontroller. Another possibility is to drive the MOSFET switch with the internal bitstream (if provided on an external pin) of a delta–sigma DAC, with its improved resolution–bandwidth tradeoff (compared with $f_c = f_{clk}/N$ for the simple PWM).[17]

Exercise 13.3. Design a circuit to generate a 10 kHz train of pulses of width proportional to an 8-bit binary input code. Use counters and magnitude comparators (suitably expanded).

A. An unusual PWM DAC

We close the PWM discussion by mentioning an unusual DAC from Linear Technology. Their LTC2644 (dual) and 2645 (quad) are PWM-to-DAC converters (a DAC with a digital PWM input). Each channel measures the duty cycle (fraction of time HIGH) during each incoming PWM cycle, and immediately presents *and holds* the corresponding correct output voltage. These particular devices are available in 8-, 10-, or 12-bit resolution versions, with internal 10 ppm/°C voltage reference and monotonic rail-to-rail voltage output. They are a great improvement over the classic technique of lowpass filtering a PWM input. The possibilities are mind-boggling!

[15] You could instead use a differential amplifier between the two sense resistors, with a single ADC channel digitizing the resulting bipolarity signal.

[16] In motor-drive applications it's often desirable to keep the drive frequency above audibility, so you don't drive people nuts. In this example we'd have to use a modulus N=800 or less, throwing out another bit of control resolution.

[17] There are families of ICs, intended for speaker-driving audio power output stages, that accept a digital input stream in a standard audio format (such as I^2S), and create as output a ΔΣ-like switching waveform to drive a MOSFET H-bridge. Some of these include the MOSFETs on-chip – a single-chip PCM-to-speaker solution.

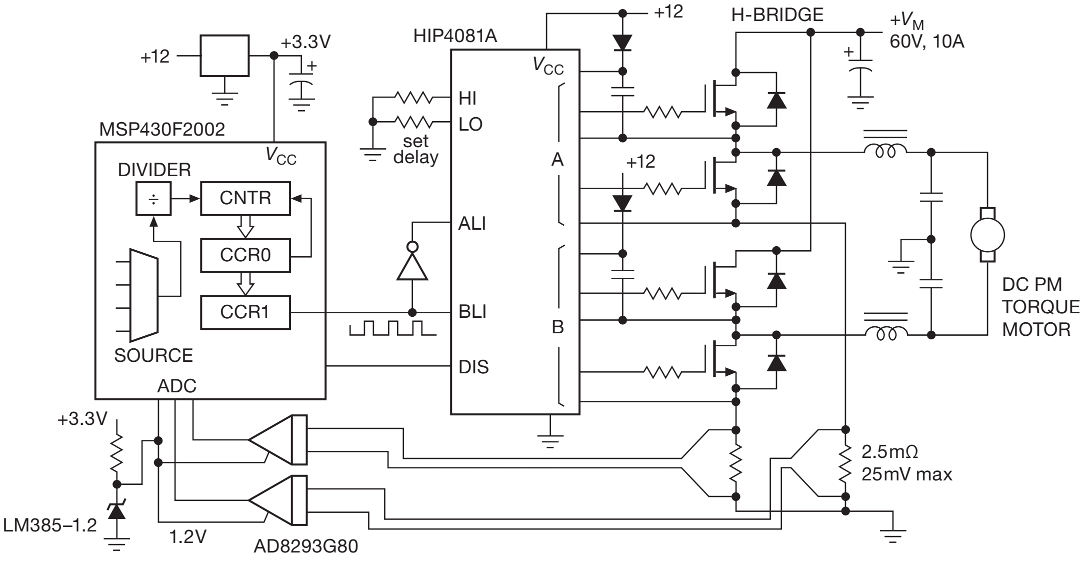


Figure 13.12. Controlling a torque motor with pulse-width modulation.

13.2.9 Frequency-to-voltage converters

In conversion applications the "digital" input may be a train of pulses or other waveform of some frequency; in that case, direct conversion to a voltage is sometimes more convenient than the alternative of counting for predetermined time, and then converting the binary count as in the preceding methods. In direct F/V conversion, a standard pulse is generated for each input cycle; it may be a voltage pulse or a pulse of current (i.e., a fixed amount of charge).

An *RC* lowpass filter or integrator then averages the pulse train, giving an output voltage proportional to the average input frequency. Of course, some output ripple results, and the lowpass filter necessary to keep this ripple less than the D/A precision (e.g., $\frac{1}{2}$ LSB) will, in general, cause a slow output response. To ensure less than $\frac{1}{2}$ LSB output ripple, the time constant τ of a simple *RC* lowpass filter must be at least $\tau \geq 0.69(n+1)T_0$, where T_0 is the output period of the *n*-bit F/V converter corresponding to maximum input frequency. The output of this *RC* network will settle to $\frac{1}{2}$ LSB, following a full scale change at the input, in $0.69\,(n+1)$ filter time constants. In other words, the output settling time to $\frac{1}{2}$ LSB will be approximately $t = 0.5\,(n+1)^2\,T_0$. A 10-bit F/V converter with 100 kHz maximum input frequency, smoothed with an *RC* filter, will have an output-voltage settling time of 0.6 ms. With more complicated lowpass filters (sharp cutoff) you can get improved performance. Before you get carried away with fancy filter design, however, you should remember that F/V techniques are most often used when a voltage output is not what's needed. For some perspective, see the previous discussion about intrinsically slow loads in connection with pulse-width modulation (§13.2.8).

13.2.10 Rate multiplier

This is a somewhat rarified method, of occasional (*very* occasional!) utility. A "rate multiplier" is a bit of clocked synchronous logic that accepts a multibit digital input quantity (binary or BCD), and that passes (or blocks) clock pulses to its single output line with an *average* rate proportional to that digital quantity. You can get these as standard logic (CD4089, CD4527, or SN7497), or you can make your own. Then simple averaging, as in the preceding F/V converter, can be used to generate a dc output proportional to the digital input code, although in this case the resulting output time constant may be intolerably long, because the rate-multiplier output will have to be averaged for a time equal to the longest output period it can generate (i.e., $2^n/f_{\text{clk}}$ for a rate multiplier with an *n*-bit rate-setting input). As with PWM, rate multipliers are most useful when the output is intrinsically averaged by the slowly responding characteristics of the load itself.

An application for which this is suited is in digital temperature control, where complete cycles of ac power are switched across the heater for each rate-multiplier output pulse. In this application the rate multiplier is arranged so that its lowest output frequency is an integral submultiple of 120 Hz, and a solid-state relay (or triac) is used to switch the ac power (at zero crossings of its waveform) from logic signals.

Note that the last four conversion techniques involve some time averaging, whereas the resistor-ladder and current-source methods are "instantaneous," a distinction that also exists in the various methods of analog-to-digital conversion. Whether a converter averages the input signal or converts an instantaneous sample of it can make an important difference, as we will see shortly in some examples.

13.2.11 Choosing a DAC

To guide you in choosing a DAC for a particular application, we've assembled in Tables 13.2 and 13.3 a representative selection of DACs of various precisions and speeds. This listing is by no means exhaustive, but it does include many of the more popular converters and some more recent entries that were intended as improved replacements.

When looking for a DAC for some application, here are some issues to keep in mind:

1. resolution;
2. speed (settling time, update rate);
3. accuracy (linearity, monotonicity; external trimming required?);
4. input structure (parallel or serial? latched? CMOS/TTL/ECL-compatible?);
5. reference (internal, or externally supplied? if external, MDAC?);
6. output structure (current output? compliance? voltage output? range?);
7. required supply voltages and power dissipation;
8. single or multiple DACs per package;
9. package style;
10. price.

13.3 Some DAC application examples

It's always helpful to get a guided tour of a real-world application example, to get a sense of the details wherein the devil resides. It's remarkably easy to put together a circuit that delivers nowhere near the performance that its converter is capable of. The four examples in this section illustrate some of the things you've got to worry about when using a DAC.

13.3.1 General-purpose laboratory source

In our research laboratories it's common to control experimental parameters with low-noise analog voltages, which need to be highly stable over temperature and time. For example, electromagnetic traps for ions and molecules require precise voltages applied to pairs of electrostatic plates and precise currents through coils. Given the diversity of applications, the output range should be selectable in both polarity and span.

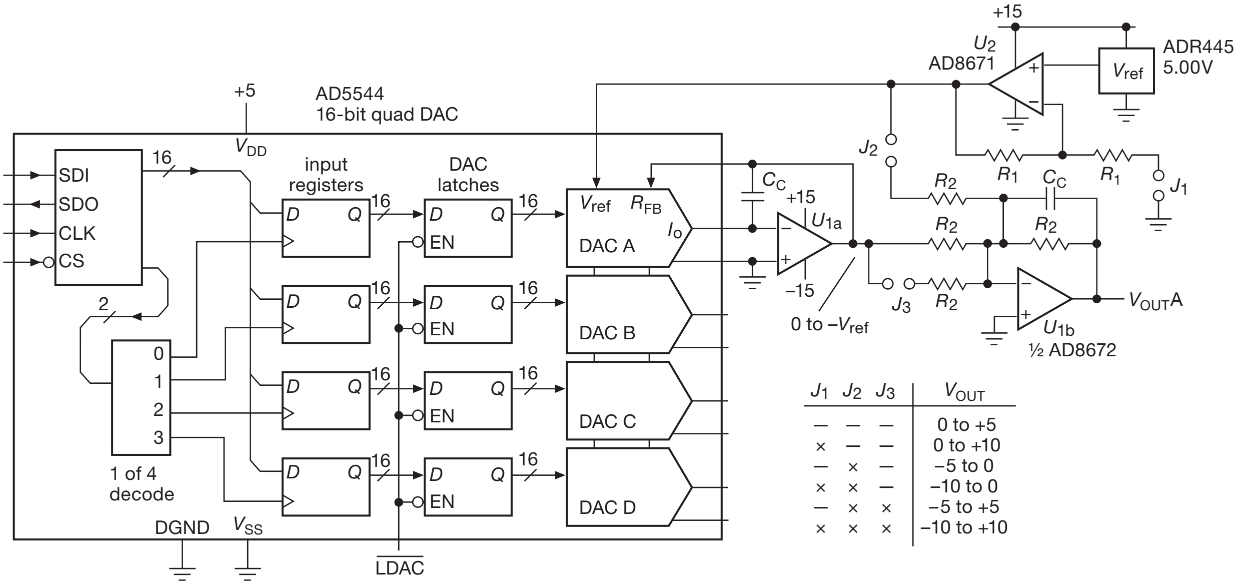
Figure 13.13 shows the business end of a popular product (the "BabyDAC"[18]) from our university's Electronic Instrument Design Laboratory. The core is the AD5544, a quad 16-channel current-output multiplying DAC that accepts an external reference voltage and outputs a set of four currents into an external node held at ground. You generate a voltage output with an external op-amp, using the matched internal feedback resistor. The internal structure for each channel is an R–$2R$ ladder driven by V_{ref}, and a set of switches that connect each $2R$ leg either to the I_{o} output or to ground. The external reference voltage can be of either polarity, in the range ± 10 V. In fact, it can be a *signal* whose instantaneous voltage is multiplied by the digital input code to produce an output signal (hence a "multiplying DAC"); in such an application it has a signal bandwidth to audio frequencies and beyond.[19]

For this application the designer used a static V_{ref}, derived from the low-noise ADR440-series of voltage references (§9.10.3) based on a JFET analog of the standard BJT bandgap reference. The outboard circuitry is rigged up to provide jumper-selectable output-voltage ranges, both unipolarity and bipolarity; so J_1 selects a reference voltage of +5 V or +10 V. Jumper J_3 selects a gain of -2 for the output amplifier, doubling the span of U_{1a}'s output; and jumper J_2 offsets the output by the selected value of V_{ref}. With these three jumpers you can select any of the six V_{out} ranges listed in the figure.

That's the basic circuit. With ideal components the output would be precise, noiseless, and drift-free. We live in a real world, though, in which we have to choose among available components to provide the best balance of necessary compromises. For the kind of laboratory applications we have in mind, stability and low noise are paramount. In the noise department the voltage reference is usually the biggest troublemaker, thus the choice of the

[18] One of hundreds of elegant circuits and instruments from the ever-prolific Jim Macarthur, EIDL's design guru.

[19] Caution, here: the datasheet lists a "reference multiplying bandwidth" of 2 MHz, but that's measured at full-scale digital code. The bandwidth is more like 20 kHz if you want 0 to -50 dB digital control. See §13.2.4.



J_1	J_2	J_3	V_{OUT}
—	—	—	0 to +5
×	—	—	0 to +10
—	×	—	−5 to 0
×	×	—	−10 to 0
—	×	×	−5 to +5
×	×	×	−10 to +10

C_C: 10nF (~1kHz); R_1, R_2: precision 10.0k arrays, e.g. Vishay type PRA100I4

Figure 13.13. General-purpose programmable laboratory source, designed with the four-channel DAC5544 for low noise and good stability. Use an isolator for the SPI digital link signals to minimize coupled digital noise.

"ultra-low-noise" ADR445, among the quietest with its ~2 μVpp typical low-frequency (0.1–10 Hz) noise. Its drift spec is also quite good (1 ppm/°C typ, 3 ppm/°C max).[20] Reference noise can be reduced by *RC* filtering (see Figure 13.19 on page 898 for an example), or, if you're desperate, by paralleling the outputs of several references (with small ballasting resistors to ensure current sharing). The op-amps are quiet by comparison: 0.1 μVpp (max) low-frequency noise; similarly, the noise contributed by the DAC is negligible.

In this circuit the op-amp gains are rolled off around 1 kHz to minimize high-frequency noise at the output (originating in the reference, and in the DAC's code-change glitches). This bandwidth was chosen somewhat arbitrarily, on the assumption that the outputs are not going to be changing rapidly. The output bandwidth could be extended by a decade if the DAC were being programmed near its maximum rate. Or, for an application that is basically quasi-static, you could limit the bandwidth still further.

For the output *stability* it's common to look for specifications of drift with temperature and with time. The typical voltage tempcos here are 1 ppm/°C for the reference and for the DAC (thus the gain: the corresponding full-scale tempco is 5 μV/°C or 10 μV/°C), and 0.3 μV/°C for the op-amps (which scales according to the gain jumper, either ×1 or ×2). With BJT-input op-amps you have to worry about input bias current and its tempco. For these op-amps the bias current is 3 nA (typ), with no tabulated tempco; however, there's an unsatisfying graph of I_B versus temperature that suggests a tempco in the range of 5 pA/°C. With the circuit's 5k (or less) impedance seen at the op-amp inputs, this amounts to a 25 nV/°C drift, entirely negligible in comparison with everything else that's going on.

Manufacturers tend to be shy about specifying drift over *time*. For the components here there's no long-term drift spec for the DAC or op-amps. The ADR445 voltage reference specifies 50 ppm typical drift over 1000 hours, but with an interesting footnote that reads "The long-term stability specification is noncumulative. The drift in the subsequent 1000-hour period is significantly lower than in the first 1000-hour period."[21]

[20] You'd get lower drift with the best-in-class MAX6350, at 1 ppm/°C max, but with somewhat greater noise: 3 μVpp.

[21] Curiously, some manufacturers prefer to specify the long-term drift per *square root* of time, suggestive either of a decreasing drift as the part ages, or perhaps a random walk. An example is the astonishing LTZ1000 ovenized zener, with a specified long-term drift of 2 μV/$\sqrt{\text{kHr}}$ (typ), and a claimed typical tempco of 0.05 ppm/°C.

Table 13.2 Selected D-to-A Converters[a]

					Cost	Total Supply													Packages, Pins						
Part #	Bits	settle[t] (μs)	Channels	Output[b]	qty25 ($US)	min (V)	max (V)	Single supply[c]	Supply Current (mA)	Interface[d]	Bipolar sig	Latch	Internal Ref[e]	External Ref[g]	Rail-rail out	Sleep Mode	Multiplying	5V inputs OK?	DIP	SOIC	TTSOP	SOT-23	Smaller	Monotonic	Comments
AD5300	8	6	1	V	2.28	2.7	5.5	•	0.14	S	-	•	-	V_S	•	•	-	-	-	-	8	6	-	•	A,J
MCP4706	8	6	1	V	0.60	2.7	5.5	•	0.21	I	-	•	-	•	•	•	-	-	-	-	-	6	6	•	M,Z6
AD8801	8	0.6	8	V	6.20	3	5.5	•	0.001	S	-	•	-	•	•	•	-	-	16	16	-	-	-	•	B,D
DAC0808	8	0.15	1	I	1.30	7	36	-	2.6	P	•	-	-	•	-	-	•	•	16	16	-	-	-	•	E
DAC08	8	0.085	1	I	1.61	9	36	-	3.6	P	•	-	-	•	-	-	•	•	16	16	-	-	-	•	E,F
LTC1663	10	30	1	V	3.70	2.7	5.5	•	0.06	2	-	•	1.25	•	•	•	-	-	-	8	-	5	-	•	A,G
LTC1669	10	30	1	V	2.39	2.7	5.5	•	0.06	I	-	•	1.25	•	•	•	-	•	-	8	-	5	-	•	A,G
AD5620	12	10	1	V	3.81	4.5	5.5	•	0.55	S	-	•	2.5	-	•	•	-	-	-	-	-	8	-	•	C,H
DAC121	12	8	1	V	2.07	2.7	5.5	•	0.20	S	-	•	-	V_S	•	•	-	-	-	-	8	6	-	•	A
DAC7611	12	7	1	V	6.05	4.75	5.25	•	0.50	S	-	•	4	-	-	-	-	•	8	8	-	-	-	•	K,L
MCP4725	12	6	1	V	0.80	2.7	5.5	•	0.20	I	-	•	-	V_S	•	•	-	-	-	-	-	6	-	•	M
AD7845	12	5	1	V	10.00	28	32	-	4.6	P	•	•	-	•	-	-	•	•	-	-	-	-	-	•	N
MCP4921	12	4.5	1,2	V	1.64	2.7	5.5	•	0.20	S	-	•	-	•	•	•	•	-	8	8	8	-	-	•	O,P
TLV5638	12	1	2	V	10.56	2.7	5.5	•	4.3	S	-	•	2	•	•	•	-	•	-	8	20	-	-	•	Q
TLV5630	12	1	8	V	14.87	2.7	5.5	•	16	S	-	•	2	•	•	-	-	•	-	20	20	-	-	•	A,G,Q,R
DAC5672	14	0.020	2	I	22.42	3.3[n]		•	100	P	-	•	1.2	•	-	•	-	-	-	-	48	-	-	-	S
AD9739	14	0.013	1	I	56.00	1.8 and 3.3[n]		-	400,70	L	-	•	1.2	•	-	•	-	-	-	-	-	-	160	-	T
AD5660	16	10	1	V	5.90	4.5	5.5	•	0.6	S	-	•	2.5	-	•	•	-	-	-	-	-	8	-	•	C,U
DAC8564	16	10	4	V	11.91	2.7	5.5	•	1.0	S	-	•	2.5	•	•	•	-	-	-	-	16	-	-	•	A,V
LTC2656	16	8.9	8	V	29.50	2.7	5.5	•	3.0	S	-	•	2.5	•	•	•	-	•	-	-	20	-	20	•	C,W
AD5686R	16	5	4	V	9.90	2.7	5.5	•	1.1	S	-	•	2.5	o	•	•	-	•	-	-	16	-	16	•	C,X
AD5541A	16	1	1	V	15.39	2.7	5.5	•	0.13	S	-	•	-	•	•	-	•	•	-	-	10	-	8	•	A,Y
LTC1668	16	0.020	1	I	18.81	+5 and -5[n]		-	3.33	P	•	•	2.5	•	-	-	-	•	-	-	28	-	-	-	Z
DAC5682	16	0.010	2	I	45.14	1.8 and 3.3[n]		-	500,133	L	-	•	1.2	•	-	•	-	-	-	-	-	-	64	-	Z2
AD5780	18	2.5	1	V	32.23	5	33	-	10,10	S	•	•	-	•	-	•	-	•	-	-	-	-	24	•	A,Z3
DAC9881	18	5	1	V	28.98	2.7	5.5	•	0.85	S	-	•	-	•	•	•	-	-	-	-	-	-	24	•	C,Z4
AD5791	20	1	1	V	53.53	10	33	-	4.2	S	•	•	-	•	-	-	-	-	-	-	20	-	-	•	Z5

Notes: (a) see also MDAC Table 13.3; listed by increasing resolution, then speed. (b) V-voltage; I-current. (c) operates from a single positive supply. (d) 2 - 2-wire serial; I - I^{2}C; L - parallel LVDS; P - parallel; S - SPI. (e) 2 = 2.048; 4 = 4.096. (g) can use ext ref. (n) nominal. (o) non-R version. (t) typical.

Comments: A: power-on to 0V. **B:** power-on to midscale. **C:** power-on to 0V or midscale. **D:** TrimDAC, pot replacement, Z_{out}=5kΩ. **E:** multiplying, to ~1MHz. **F:** compliance to –10V and +18V. **G:** double buffered for simultaneous updating. **H:** 14-bit=AD5640, 16-bit=AD5660; 0.2%, 5ppm/°C ref. **J:** digital gain & offset adjustment. **K:** DAC8512 2nd source. **L:** power-on to 0V plus a CLR pin. **M:** power-on state from on-chip EEPROM. **N:** multiplying, to 600kHz. **O:** multiplying, to 450kHz; 14-pin dual=4922. **P:** power-on to hi-Z. **Q:** programmable settling time. **R:** 10-bit=TLV5631; 8-bit=TLV5632. **S:** 275Msps; 1.5ns to 90%; DAC2904, AD9767 2nd source. **T:** RF synthesis, 2.5Gsps. **U:** ext ref version =5662; 0.2%, 5ppm/°C ref. **V:** 14-bit=DAC8164; 12-bit=DAC7564; low glitch; 0.004%, 2ppm/°C ref. **W:** 0.2%, 2ppm/°C ref; also 4.096 Vref and 12-bit versions. **X:** 2ppm/°C ref. **Y:** multiplying, to ~1MHz; low glitch; 0.1ppm/°C; 12nV/√Hz; unbuffered output. **Z:** low glitch; 50Msps; 12-bit and 14-bit versions. **Z2:** 1Gsps; FIFO; clock PLL; on-chip digital filters. **Z3:** 8nV/√Hz; 0.02ppm/°C; 3 pwr supplies; “system ready.” **Z4:** 0.3ppm/°C; 24nV/√Hz. **Z5:** 0.05ppm/°C; 7.5nV/rtHz. **Z6:** 10-bit=MCP4716; 12-bit=MCP4726.

As we remarked at the outset, for the intended application it is noise and drift that are most important. By contrast, the absolute accuracy is not particularly impressive: the worst-case specs are ± 200 ppm for the voltage reference, $\pm 75\,\mu$V for the op-amps, and ± 3 mV full-scale gain error for the DAC. Translated to units of 16-bit LSB steps, and taking the output range to be ± 10 V (for which an LSB is 0.3 mV), these correspond to ± 13 LSB, ± 0.5 LSB, and ± 10 LSB.

13.3.2 Eight-channel source

If your application does not need the flexibility of output polarities and scale factors of the last example, and if you can tolerate a bit more noise and drift, you can do pretty well with a fully integrated multiple-section voltage-output DAC, for example the LTC2656 in Figure 13.14. This particular IC includes an internal voltage reference of good stability (± 2 ppm/°C typ, ± 10 ppm/°C max), with a 1 ppm/°C typical full-scale tempco for the DAC

Table 13.3 Multiplying D-to-A Converters[a]

		t_{settle}	Channels	Output Type	Cost	Power Supplies				V_{in} Range	V_{out} Range	I_{supply}	Interface[c]	Distortion (1kHz)		Noise (nV/√Hz at 1kHz)	Bandwidth		Feedthrough[b]	Packages, Pins				Comments
						Positive		Negative																
Part #	Bits	typ (µs)			qty25 ($US)	min (V)	max (V)	min (V)	max (V)	(V)	(V)	typ (µA)		typ (dB)	@V_{pp} (V)		typ (MHz)	@V_{pp} (V)	(MHz)	DIP	SOIC	TTSOP	Smaller	
AD8842	8	2.9	8	V	13.71	4.75	5.25	4.75	5.25	±3	±3	10	S	-80	4	78	1.5	0.1	0.25	24	24	-		A
TLC7528	8	0.1	2	I	3.47	4.75	15	-	-	±25	±25	2000	P	-85	6	-	-	-	-	20	20	20	-	B
MCP4921	12	4.5	1	V	1.64	2.7	5.5	-	-	0 to V_S	0 to V_S	200	S	-73	0.4	-	0.45	0.4	-	8	8	8	-	C
AD/LTC7541	12	0.6	1	I	6.00	5	16	-	-	±10	±0.5	100	P	-	-	-	-	-	-	18	18	-	20	D
AD7943,5,8	12	0.6	1	I	7.78	3	5.5	-	-	±10	±0.3	0.005	S,P	-83	17	35	-	-	2	16	20	20	-	E
LTC8043	12	0.25	1	I	8.10	4.75	5.25	-	-	±10	±0.5	100[m]	S	-108	17	17[m]	-	-	1	8	8	8	-	F
AD5443	12	0.06	1	I	4.87	3	5.5	-	-	±10	±0.3	0.6[m]	S	-81	3.5	25	10	7	10[e]	-	-	10	-	G
DAC7821	12	0.05	1	I	6.38	2.5	5.5	-	-	±15	±0.3	0.8	P	-105	-	18	10	7	0.8	-	-	20	-	H
AD5544	16	0.9	4	I	27.43	2.7	5.5	0	5.5	±15	±0.3	5[m]	S	-98	5	7	5	5	-	-	-	28	32	I,J
DAC8814	16	0.5	4	I	29.06	2.7	5.5	-	-	±15	±0.3	2	S	-105	5	12	10	0.3	2	-	-	28	-	J
DAC8820	16	0.5	1	I	16.98	2.7	5.5	-	-	±15	±0.3	5[m]	P	-105	17	10	8	5	0.15	-	-	28	-	K
LTC2757	18	2.1	1	I	57.43	2.7	5.5	-	-	±15	±0.3	0.5	P	-110	5	13	-	-	1[e]	-	-	-	48	L

Notes: (a) listed by increasing accuracy and speed; all are monotonic, and all have latches except AD/LTC7541; see also DAC Table 13.2. (b) capacitive coupling causes a 6dB/octave rising output (from the desired digital value of attenuation) at high frequencies; the listed value is the frequency at which there is a +3dB increase relative to a –40dB programmed attenuation (i.e., an actual attenuation of –37dB). (c) 2 - 2-wire serial; I -I^2C; P - parallel; S - SPI; L - parallel LVDS. (d) 65dB feedthrough at 100kHz. (e) 3dB loss, rather than feedthrough. (m) min or max. (n) nominal. (t) typical.

Comments: A: current conveyor, V_{out}=V_{in} x (D/128 – 1). **B:** 2nd source for AD7528. **C:** MCP4922=dual. **D:** 80dB feedthru @ 10kHz and full-swing. **E:** improved AD7543, -45, and -48; AD7545 & -48 are parallel interface. **F:** also DAC8043. **G:** 10-bit=AD5432, 8-bit=AD5426. **H:** parallel readback. **I:** 14-bit=AD5554. **J:** reset to zero or midscale. **K:** reset to zero. **L:** double buffer, readback, reset to zero.

outputs. The DAC's low-frequency (0.1–10 Hz) output noise is 8 μVpp, typical. This is four times the noise voltage of the previous example; taking into account the more limited output span (0–2.5 V), it represents a larger relative noise contribution still. The good news is the relative simplicity of this part: no external reference or amplifiers, and operation from a single positive supply.

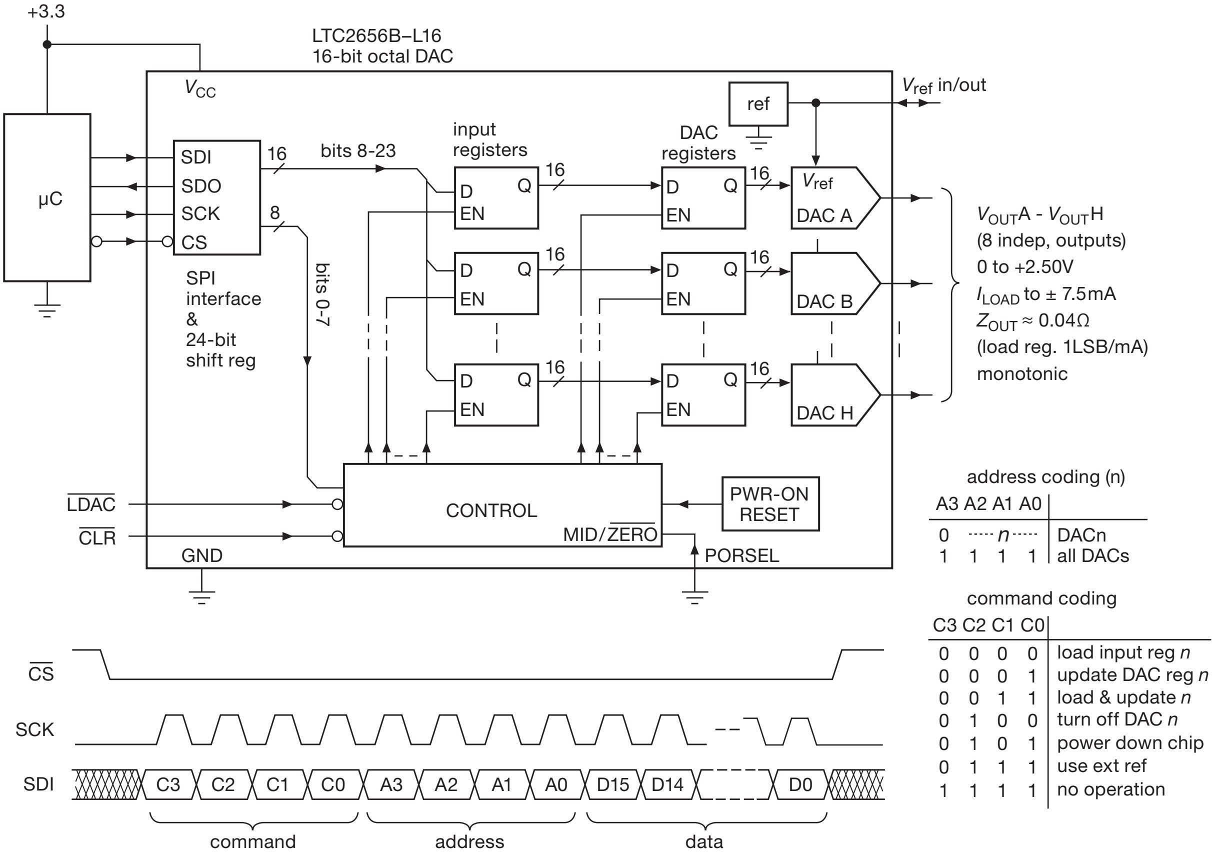
The SPI digital interface is simple and clean: every transfer is 24 bits, with 4 bits to specify the channel number (with an option to load all channels with the same value), 4 bits to specify the operation, and 16 bits to carry the digital value. Each channel is double buffered, so you can load the next value into each channel's input buffer, then transfer them simultaneously to the DAC's register so that all outputs change to their new values at the same time.

13.3.3 Nanoamp wide-compliance bipolarity current source

Here's an unusual application, and a circuit implementation of considerable subtlety, using a dual-supply current-output DAC: suppose you need a programmable current source that can operate over a wide voltage range (say ±10 V) while sourcing (or sinking) very small currents (in the nanoamp range, say). You might need this to measure the V–I characteristics of a semiconductor at the low end of its current range, or perhaps for a research application such as the electrical properties of nanofibers. Another application might be to cancel the input leakage current of a high-impedance measuring device, for example an 8-digit digital multimeter with a JFET frontend (a discrete matched JFET pair or a precision JFET-input op-amp), in which the leakage current rises rapidly (but predictably) with temperature.[22] Such an instrument could store a table of leakage versus temperature, measured during initial calibration, to be used in conjunction with a temperature sensor to program the cancellation current during normal operation. Current-output DACs don't operate successfully down at these currents, and furthermore they don't generally provide outputs that can both source and sink current under control of a digital input code.

This circuit (Figure 13.15) is tricky, and seriously confusing at first. Work with us, here, and you'll get it (eventually). An essential component is the simple floating current source circuit (Figure 13.16), in which an op-amp follower with an added voltage V_0 at the output bootstraps a feedback resistor R to create a current V_0/R. That voltage can be created by biasing a zener-type reference, as shown; or

[22] It was an analogous circuit in Agilent's 34420A multimeter (see §5.12.5) that inspired this example. (Their circuit used a voltage reference in place of R_3, allowing the use of a single DAC output. They also used a smaller R_s, so that the output range was ±2 nA.)



address coding (n)

A3	A2	A1	A0	
0	·····	*n*	·····	DACn
1	1	1	1	all DACs

command coding

C3	C2	C1	C0	
0	0	0	0	load input reg *n*
0	0	0	1	update DAC reg *n*
0	0	1	1	load & update *n*
0	1	0	0	turn off DAC *n*
0	1	0	1	power down chip
0	1	1	1	use ext ref
1	1	1	1	no operation

Figure 13.14. Eight-channel voltage-output DAC.

it can arise from a current flowing through a resistor that returns to the op-amp's output.

Now for the complete circuit of Figure 13.15. The DAC08 is an oldie (circa 1984), with a pair of opposite-ramping current-sinking outputs that sum to a constant current $I_{\rm ref}$ (set by the current sourced through R_1, here equal to 5 V/39.2 kΩ). An 8-bit offset-binary input code sets the individual output currents. For example, the minimum code (00h) causes $I'_{\rm o}$ to sink 128 μA and $I_{\rm o}$ to sink 0; for a quarter-scale code (20h, or 32 decimal) the corresponding currents are 96 μA and 32 μA; and so on. With +18 and −15 V supplies, the output compliance extends from −12 V to +18 V.

The external circuitry works the real magic, by (a) converting this unipolar current-sinking pair into a bipolarity (sourcing or sinking) output current, and (b) simultaneously scaling down the current by a factor of 10,000, to generate an output terminal that programmably sources or sinks currents with a full-scale range of ±12.8 nA, in LSB steps of 0.1 nA.

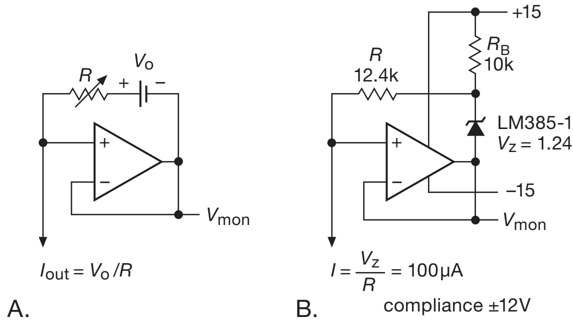


Figure 13.16. A follower with a voltage applied in series with a resistor creates a simple floating current source, and the op-amp's output pin provides a "voltage monitor." A. The basic scheme. B. Implementation with a bandgap voltage reference.

To understand first the current scaling, disconnect U_{2b} and look at the upper op-amp alone: the current sunk by the DAC generates a small voltage across $R_{\rm s}$, approximately $I_{\rm o}R_{\rm s}$ (for $R_{\rm o} \gg R_{\rm s}$). That's the $V_{\rm o}$ of the floating current

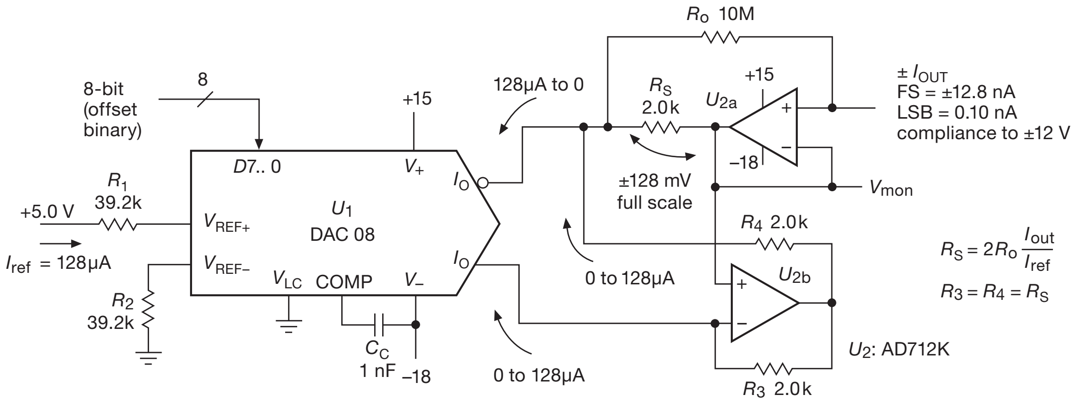


Figure 13.15. Wide-compliance programmable nanoamp current source/sink. The output voltage is buffered at U_{2a}'s output, for "source-measure" applications (i.e., source a current, measure the corresponding voltage). For a 2 nA range, set R_s=R_3=R_4=316Ω (or increase R_o).

source of Figure 13.16, here sinking a current I_oR_s/R_o from the load; that is, the current is scaled down by a factor of 5000. This is really just a "current divider" in fancy clothing. (For the complete circuit the ratio is 10,000:1, because R_4 is in parallel with R_s.)

Now reconnect the lower op-amp, and for the moment ignore the DAC's upper output, I'_o. The current sunk by the DAC puts U_{2b}'s output above its inputs by I_oR_3, which *sources* a current through the series pair R_4 and R_s.[23] So, with everything reconnected, the net current sourced into the left-hand side of R_s is the (positive) current equal to the magnitude of the DAC's I_o output, minus the (negative) current equal to the magnitude of the DAC's I'_o output. So the net current goes from $-128\,\mu$A (at minimum input code, 00h) to $+128\,\mu$A (at maximum input code, FFh). That current is divided by a factor of 10,000 ($R_o/[R_s\|R_4]$) to generate the net output current, with its full-scale range of ±12.8 nA.

If you need more precision, substitute the similar 10-bit DAC10. It differs in having somewhat less negative compliance, and in providing a full-scale output sink current equal to *twice* the reference current.

And a final comment: there are other ways to make a wide-compliance nanoamp current source; see, for example, the relatively simple design of Figure 5.69A.

A. Variations on the floating current source

Going back to the simple floating current source circuit of Figure 13.16, there are several useful variations that permit control via a dc input voltage (with respect to ground), or by a digital code. Figure 13.17 shows how to substitute the floating output of a difference amplifier (§5.14) in place of the biased zener of Figure 13.16B. Here the G=0.1 difference amplifier converts the programming voltage, in the range ±10 V with respect to ground, to a ±1 V output referenced to the op-amp's output, thus an output current of $I = V_{\text{prog}}/10R$.

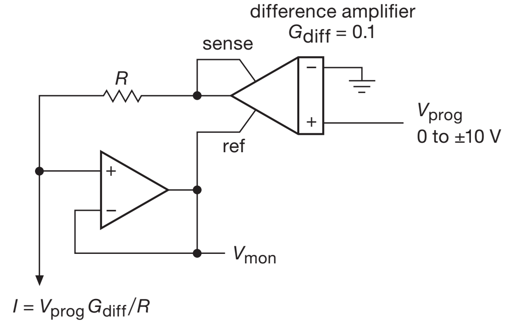


Figure 13.17. Floating current source analogous to Figure 13.16, programmed with a difference amplifier. See also Figure 5.69.

Some commercial instruments are capable of providing programmable currents over a very wide voltage range (say ±200 V). This they do (Figure 13.18) by powering the op-amp with a floating dc supply (say ±5 V), with a DAC (powered from the same supply) substituting for the difference amplifier in Figure 13.17. Both the op-amp and the DAC fly over the wide compliance range, with the DAC's digital input data fed via opto-isolators. In this scheme the op-amp's output is buffered and used to bootstrap the dc supply common, thus keeping the op-amp's

[23] This op-amp configuration is in fact a current mirror, sourcing into the node at its noninverting input a current proportional to the current sunk from its inverting input, in the ratio of the output resistor to R_3.

inputs and outputs near mid-supply; the bootstrap acts also to prevent dynamic currents caused by the supply's capacitance to its primary power source. These programmable current sources with voltage-monitor outputs are examples of what's called a "source-measure unit" (SMU), typified by units from Keithley (their 2400 series) or Keysight (their B2900 series).

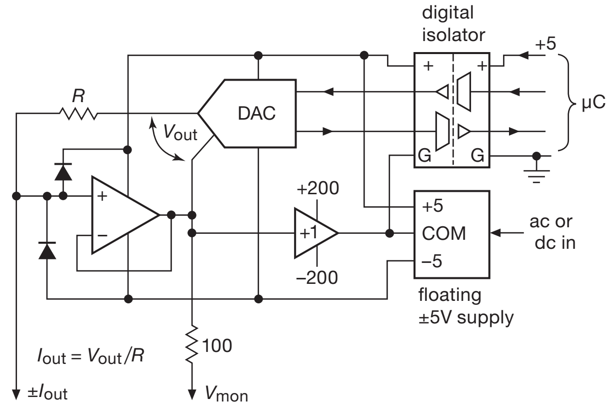


Figure 13.18. High-voltage floating current source scheme used in "source-measure" instruments. The unity-gain high-voltage buffer can be a simple push–pull follower, as its job is only to bootstrap the ±5 V floating supply. Adding a divider and buffer to the V_{mon} output produces a low-voltage replica.

13.3.4 Precision coil driver

Here's an application that pushes the limits of DAC resolution and stability: a current-source driver to provide a settable and stable current (of either polarity) through a pair of coils that trim the magnetic field in a magnetic resonance apparatus. This kind of application can require parts-per-million (ppm) resolution and stability. It's an instructional example worthy of some detailed discussion.

A. DAC and voltage reference

Figure 13.19 shows an implementation, with the remarkable AD5791 20-bit DAC at its core. Looking first at the circuitry surrounding the DAC, we've again chosen the ADR445 reference (in the best-performing B-grade) for its low noise and excellent stability (an alternative is the MAX6350, with the recommended capacitor from its noise reduction pin to ground). We added a 10 Hz *RC* lowpass filter to suppress wideband noise: according to the datasheet the ADR445's noise in the band from 0.1–10 Hz is about 2.3 μVpp, increasing to 66 μV when that band is extended to 10 kHz.

When you're working at ppm error levels, you have to worry about everything! For example, just 1.5 nA of leakage current in the 10 μF filter capacitor (C_3) would cause nearly 1 ppm error in the +5.0 V reference driving the DAC (from the IR drop across R_{10}). The noise filter shown exploits a nice trick[24] to eliminate that error: the lower leg ($R_{11}C_4$) bootstraps the bottom of C_3 so that there's essentially no dc voltage across it, hence no leakage current; this is analogous to the technique of a *guard* electrode, which is used to eliminate leakage currents in sensitive low-current measurements (or to eliminate the effects of shunt capacitance where signals are present).

The AD5791 can operate with a single positive reference, or with both positive and negative references. For our application we want an output range of ±5 V, but we can exploit the DAC's internal precision matched resistor pair (which returns to the positive reference) so that only a positive reference is needed. The DAC provides a "sense" output from both internal reference nodes, used with feedback as shown to eliminate IR errors. Note the matched input resistances at U_2's inputs, to take advantage of input current matching ($\Delta I_B < 1$ nA); the input-current tempco, estimated from graphs in the datasheet, is less than 10 pA/°C. The op-amp's offset voltage is 12 μV typ (50 μV max), and the offset voltage tempco is 0.6 μV/°C max (0.2 μV/°C typ).

On the scale of the +5.0 V reference that the op-amp is buffering, these figures translate into 0.7 ppm, 0.007 ppm/°C, 2.4 ppm, and 0.12 ppm/°C, respectively. In other words, the bias current and offset voltage errors add up to about 3 ppm, or 3 LSBs; but the drifts are just 0.13 ppm/°C (max), or an LSB for an 8°C change in temperature. We're happy with that stability, which is what we really care about for this application. The ∼3 ppm scale error is unimportant, because in practice the current will be adjusted until the coil current is doing the right thing. Finally, we've got to consider the error and drift contributed by the DAC itself, which are comparable: its full-scale and zero-scale errors are each ±2 ppm (2 LSB) max, and its output tempco (at zero, midscale, or full-scale) is ±0.05 ppm/°C typ, ±0.5 ppm/°C max. Its typical low-frequency output noise is 0.6 μVpp at mid-scale, roughly half as much as that contributed by the reference.

B. Amplifier loop

The job here is to use the DAC's stable output voltage to control the coil current[25] over a range of ±0.1 A while

[24] Originated by Walter Jung, as far as we know.

[25] We took as a reference design a pair of coils, each 30 cm in diameter with 500 turns, spaced axially by 15 cm in the so-called "Helmholtz"

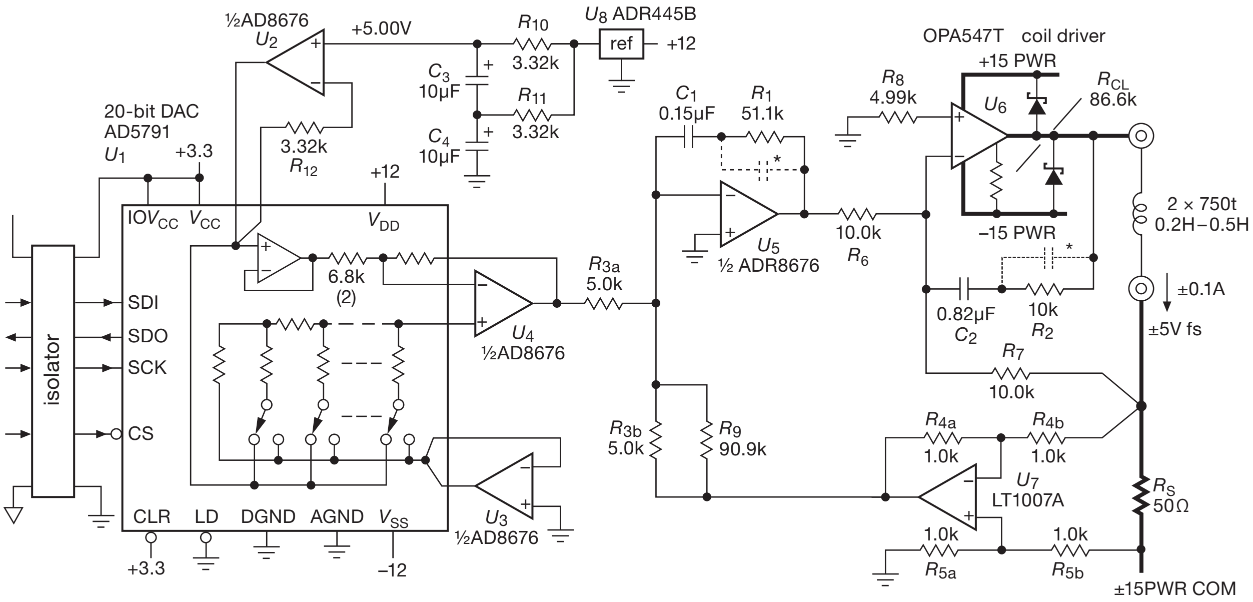


Figure 13.19. Precision programmable Helmholtz coil driver featuring the AD5791 20-bit DAC. The high current path is indicated with heavy lines. Resistors R_3–R_5 are Vishay MPM matched pairs (2 ppm/°C tracking), and R_s is a Vishay VPR221 (Y0926) bulk foil power resistor (2 ppm/°C tempco) with heat-sinking.

preserving part-per-million dc stability and noise. There's also the *other* kind of stability – freedom from oscillation.

Ignoring the latter for the moment (i.e., neglecting all of the capacitors in the circuit), at the top level the amplifier loop works like this: a temperature-stable current-sensing resistor $R_s = 50\,\Omega$ generates a full-scale voltage of ±5 V, a replica of which is subtracted from the DAC's output by the matched resistor pair R_{3ab}. Error amplifier U_5 supplies the loop gain, applying the amplified error to the coil driver U_6, which operates as a unity-gain inverter. The phases are correct: too much current through R_s drives the output of U_7 down, U_5 up, and U_6 down.

At the next level there's the dc accuracy and drift to worry about. Once again we've chosen precision BJT op-amps, the now-familiar AD8676 for the error amplifier, and the vintage LT1007A for the unity-gain difference amplifier. The latter has the lower noise and offset voltage, at the expense of a higher input current; we circumvent the latter problem by reducing the resistance seen at the inputs, which is possible because of the low (50 Ω) source impedance. The power op-amp U_6 need not be accurate, because it is within an overall feedback loop whose gain rises at low frequencies as $1/f$.

The sense resistor R_s is a Vishay "bulk metal foil" 4-wire (Kelvin connection) precision power resistor in a TO-220 package, rated at 8 W dissipation; the best grade is accurate to ±0.01%, with a typical tempco of 2 ppm/°C. The resistor pairs R_3–R_5 are each precision matched duals in convenient SOT23 packages, with 0.05% matching, and 2 ppm/°C (typ) tracking. The resistor shunting R_{3b} compensates for the reduction of R_s's effective resistance that is due to loading by R_7 and R_{4b}.

Assuming there's plenty of loop gain (and there is, see next paragraph), the ppm-scale dc stability of the DAC output is reasonably well preserved with these op-amps, assisted by resistors of this stability and tracking.

Finally, there's the serious business of stability against oscillations, which is complicated here by the inductive load. The latter alone causes a 6 dB/octave rolloff beginning at the frequency at which the inductive reactance equals the sense resistance, about 20 Hz for the nominal 400 mH coil pair. We've addressed it by making the error amplifier an integrator at lower frequencies (for plenty of dc gain), flattening to a gain of ×10 at 20 Hz.[26] That prevents the curve of overall loop gain versus frequency from dropping at more than 6 dB/octave, at least until well beyond the unity-gain frequency. The output amplifier U_6

configuration. With #20 gauge wire, the total dc resistance is 30Ω, the inductance is about 400 mH, and a current of 0.1 A produces a central field of 3 gauss (about six times Earth's field).

[26] In fancy language, a pole at dc and a zero at 20 Hz.

gets the same treatment. An additional small capacitor across R_1 (and similarly across R_2) rolls off the local gain at higher frequencies still, to stabilize against high-frequency oscillations. Suitable values are 150 pF and 4.7 nF (rolling off at 20 kHz and 3 kHz, respectively).

Rather than prattle on about this, we present the self-explanatory Bode plots of Figure 13.20. Readers who are likely to encounter situations like this can benefit from the thinking presented there.

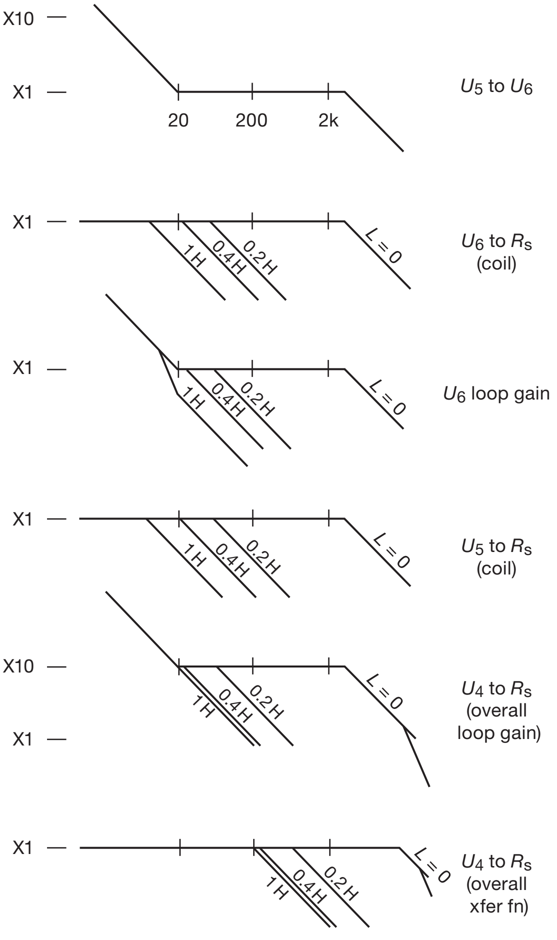


Figure 13.20. Bode plots for the coil driver amplifier in Figure 13.19, shown with several values of load inductance.

13.4 Converter linearity – a closer look

In §13.1.3 we mentioned, briefly, the sorts of errors that afflict DACs (and ADCs as well). The business of *linearity errors* deserves some more discussion.

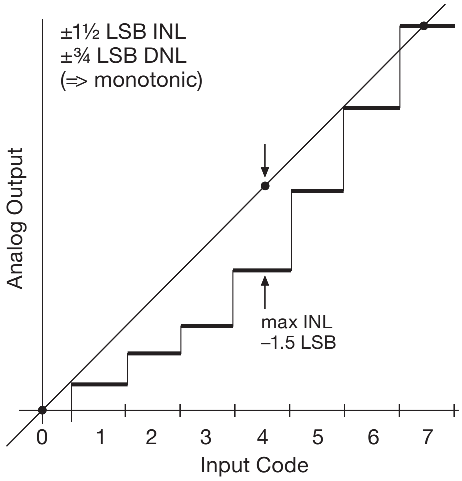


Figure 13.21. A DAC can exhibit both monotonicity (DNL<1 LSB) and relatively large integral nonlinearity (here INL=1.5 LSB).

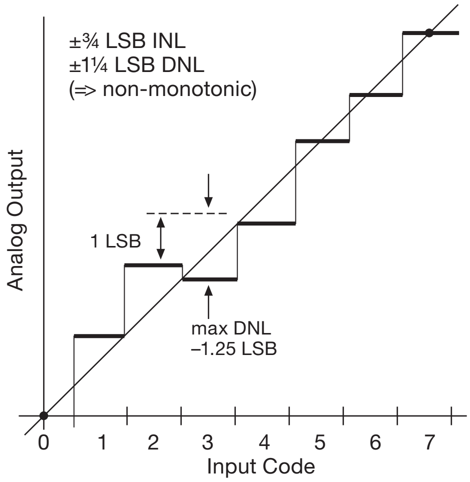


Figure 13.22. This converter has less INL (0.75 LSB) than in Figure 13.21, but its larger DNL (1.25 LSB) permits non-monotonicity. Which matters more depends on the application.

Take a look at Figures 13.21 and 13.22. Both 3-bit DACs suffer from linearity errors. But there's some subtlety here. We need a couple of definitions: *integral* nonlinearity (INL) is the maximum deviation from the ideal straight line of analog output versus digital input, over the full conversion range;[27] whereas *differential* nonlinearity (DNL) is the maximum error in any single digital step (i.e., from n=2 to n=3 in this example) from its proper step size of 1 LSB.

When do you care about INL versus DNL? If you need

[27] There's a bit of wiggle room here, because you can define the line as passing though the endpoints ("endpoint linearity," as used here), or you can make things look a bit better by using the best-fit straight line.

a DAC to hit a desired voltage with minimal error, the INL and gain-error terms will dominate, and you don't care about monotonicity. If, however, you're closing a control loop, it's the exact opposite: the servoing action of the loop will remove the INL, but large DNL could cause hidden zones of instability, which are particularly hard to debug.

A DAC's architecture influences the mix of INL/DNL. Consider two good 16-bit DACs, the DAC8564 and the AD5544. The first uses a string of resistors, so you have to work pretty hard to end up with a DNL greater than 1 LSB. And you are *guaranteed* monotonicity. However, nothing is controlling the INL except the statistical distribution of the resistor values, so it's not surprising that the INL is ±8 LSBs, and that's for the expensive part. It's 12 LSBs in the bleachers.

In contrast, in the *R*–2*R* architecture a large INL will often translate to a large DNL; the same process that keeps the DNL under control also keeps the INL down, to some extent, so the INL spec of the AD5544 is ±4 LSBs, with a DNL of 1.5 LSBs. So all other specs being equal (which they aren't), one would choose the AD5544 for precisely setting voltages, and the DAC8564 for use in a control loop.

And, while we're issuing warnings, beware, Beware, *BEWARE* using audio DACs in non-audio applications. If a DAC doesn't provide a DNL spec, it's because it's embarrassingly large. This is often acceptable in audio, but not for either control loop use or voltage setting. Likewise, gain-drift specs for audio DACs are often too large for use in voltage-setting applications.

13.5 Analog-to-digital converters

Look back at the earlier section on "Preliminaries" (§13.1) to remind yourself of some of the things to think about when choosing a converter (whether DAC or ADC). At the top level you're concerned much less with the details of how the thing actually does its conversion, and much more with the major questions of (a) performance (speed, accuracy, etc.), (b) digital interface (parallel or serial; single-ended or LVDS; etc) and (c) integration (single or multiple units; stand-alone, or integrated into a microcontroller or other complex function). In most cases you'll use a commercial ADC chip or module rather than building your own. But it's important to know about the inner workings of the various A/D conversion methods, in order not to be caught unaware by their idiosyncrasies.

13.5.1 Digitizing: aliasing, sampling rate, and sampling depth

We'll get into the nitty-gritty of analog-to-digital conversion soon enough, but first a short riff on the business of *sampling*, which will come up again and again as we visit various ADC methods.

When you convert an analog signal (e.g., an audio waveform) to a series of digital quantities (i.e., numbers corresponding to the instantaneous voltage at successive moments in time), you need to choose both the precision of the voltage measurements (the sampling *depth*) and the rate at which such samples are taken (the sampling *rate*). We saw this briefly in Chapter 6, in connection with anti-alias lowpass filters (§6.3.7A); let's look a bit more deeply here, in the context of ADC sampling of analog waveforms.

A. Sampling depth

Let's look first at the effects of bit depth (because they are the more easily understood): sampling to n equally spaced bits quantizes the waveform samples to 2^n levels, effectively limiting the dynamic range to $6n$ dB. A waveform so sampled, when properly scaled to exploit the full conversion range, will also exhibit quantization distortion, to the tune of 2^{-n} (i.e., $100/2^n$ percent).

As an example, 16-bit quantization of audio (the standard used in CD audio) has a dynamic range limited to 96 dB, and minimum distortion of 0.0015%. Of course, the signal itself is typically limited in both dynamic range and distortion; a well designed digitizing system should have sufficient bit depth (and sampling rate) so that it does not degrade the signal's quality.

At a deeper level there is more to the story than mere bit depth: nonlinearity (even nonmonotonicity!), noise, spurs, etc., all contribute to the fidelity of the digitized signal. A common metric that captures much of this is "ENOB" (effective number of bits); we'll see more of this later (see for example Figure 13.56).

B. Sampling rate and filtering

The story here is more subtle (and more interesting). Contrary to intuition, a waveform that is perfectly sampled at a rate at least twice that of the highest frequency component present suffers *no loss of information whatsoever*. Nothing is lost in the unsampled portion of the waveform in between the samples; this is the Nyquist sampling theorem (which has its cadre of unbelievers, who swear that digital audio removes the very soul of music).[28]

[28] It can be shown mathematically that the original signal (excluding a set of pathological waveforms) is perfectly recovered:

Readers harboring contempt for authority may wonder what happens if one violates statute by *under*sampling. Easy enough to try: look at Figure 13.23, where we've sampled a 100 Hz sinewave (requiring $f_{\text{samp}} \geq 200$ sps) at 90 sps, in serious violation of Nyquist's prescription. The sampled points trace out a false signal, in this case 10 Hz. This is called an "alias," and most of the time it's something you don't want.[29] Put simply, for a given sampling rate f_{samp}, the analog input signal must be lowpass filtered (with an "anti-alias filter") such that no significant signal remains above $f_{\text{samp}}/2$. Conversely, for an analog signal that extends to some maximum frequency $f_{\max}$, the minimum sampling rate is $2f_{\max}$. (You can, of course, sample more rapidly than the Nyquist limit $f_{\text{samp}}>2f_{\max}$, and in fact it's wise to do a modest degree of "oversampling" because that permits more relaxed lowpass filtering of the analog signal, as we'll see presently.)

It's helpful to look at the business of aliasing in the *frequency* domain. In Figure 13.24A we've attached a wimpy 2-section *RC* filter to a broadband signal, putting the -3 dB point of each section at the Nyquist limit ($f_{\text{samp}}/2$). Frequency components in the forbidden region get falsely digitized as shown,[30] contaminating the intended signal band; they cannot be removed later by filtering – in the digitized output they are now "in-band."

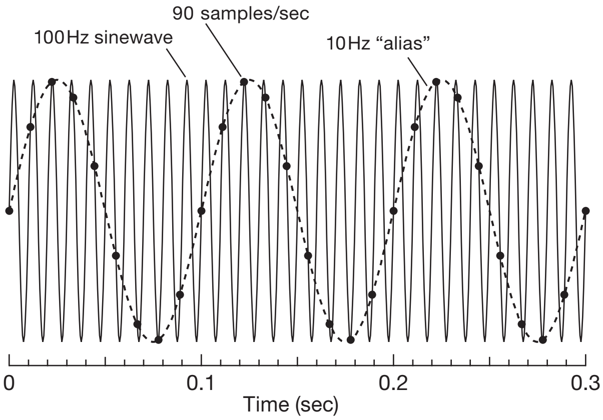


Figure 13.23. Digitizing at less than the Nyquist rate produces "aliases." A 100 Hz sinewave (solid line) sampled at 90 sps (far below the 200 sps Nyquist rate) produces a 10 Hz alias (dots, connected by a dashed line).

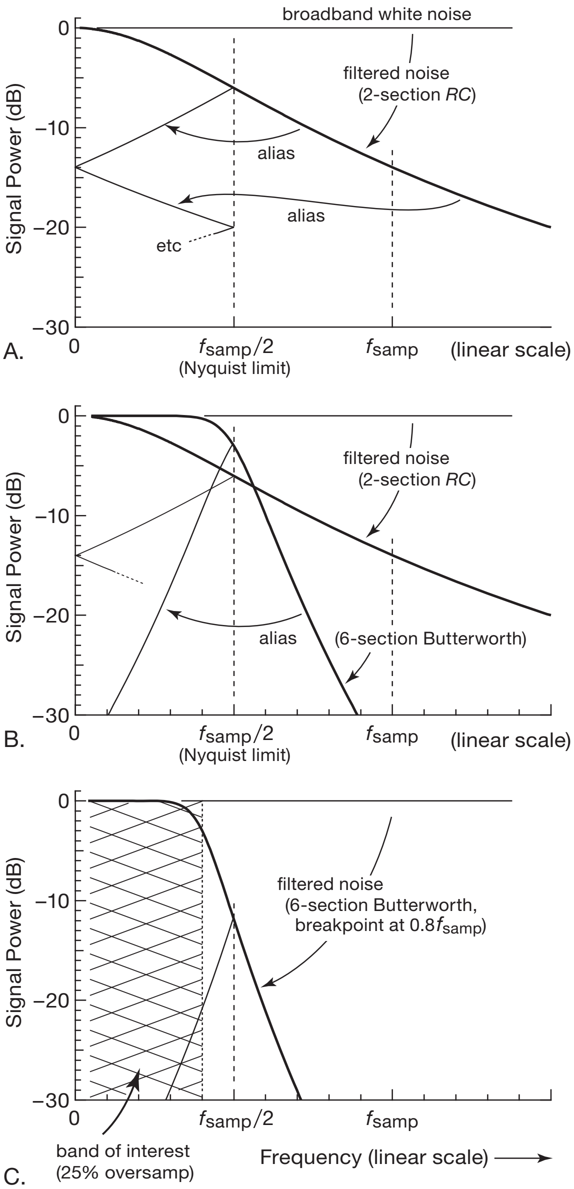


Figure 13.24. Undersampling, oversampling, and aliasing. A. Sampling a signal with frequency components above the Nyquist limit ($f_{\text{samp}}/2$) produces aliased digitized versions that fall within the properly-sampled signal; here a gentle 2-section *RC* rolloff allows significant aliased out-of-band signal energy to contaminate the intended signal. B. A steeper filter is more effective; but aliasing still contaminates the Nyquist band edge. C. Oversampling (setting the Nyquist frequency above the band of interest, here by 25%) gives the anti-alias filter a guard band in which to roll off, greatly reducing aliasing.

$v(t)=\sum v_i \operatorname{sinc} \pi(f_s t - i)$, where f_s is the sampling rate and v_i is the amplitude of the ith sample.

[29] See §13.6.3 for an important exception.

[30] To draw these "elevation sketches," just mirror the contours seen past the multiples of the Nyquist frequency.

A steeper anti-alias filter does a better job, as seen in Figure 13.24B, where we've included a 6-pole Butterworth whose -3 dB point is set at $f_{\text{samp}}/2$. But the situation is still far from ideal, with plenty of aliased signal present, especially at the high frequency end.

What you do, then, is to run the sample clock somewhat faster than the Nyquist minimum, as seen in Figure 13.24C, illustrating 25% oversampling relative to the signal band we care about. That gives the anti-alias filter a guard band in which to transition from passband to stopband. Note that we've set the filter's -3 dB point at the band edge, not at the Nyquist frequency.

This is how it's done, in applications where signal purity is important. Using again the example of CD audio, the 20 kHz audio band, if limited by a perfect brickwall lowpass filter at 20 kHz, could be sampled at the Nyquist limit of f_{samp}=40 ksps; but the CD standard sets the rate at 44.1 ksps (10% oversampling), allowing for a 20% filter guard band.[31] Later, in Figure 13.60, we'll use a frequency-domain view of aliasing to understand some benefits of delta–sigma conversion.

Note that there are compromises involved in anti-alias filter design. For example, an analog multi-section filter with steeper transition to cutoff (e.g., a Chebyshev filter) exhibits poorer performance in the time domain (overshoot and ringing, poor phase characteristics, sensitivity to component values, etc.) – see Figures 6.25 and 6.26. For much more on filter types and characteristics see the extensive discussion in Chapter 6 (particularly §6.2.5). And, while you're paging through earlier chapters in search of wisdom, be mindful always of the degrading effects of noise (Chapter 8).

13.5.2 ADC Technologies

There are half a dozen basic techniques of A/D conversion, each with its peculiar advantages and limitations. In the following subsections we'll take each in turn, along with some application examples. Here, in outline form, is a compact summary of these techniques.

Flash, or "parallel" (§13.6) The analog input voltage is compared with a set of fixed reference voltages, most simply by driving an array of 2^n analog comparators to generate an n-bit result. Variations on this theme include pipelined or folded architectures, in which the conversion is done in several steps, each of which converts the "residue" of the previous low-resolution conversion.

Successive approximation (§13.7) Internal logic generates successive trial codes, which are converted to voltages by an internal DAC and compared with the analog input voltage. It requires just n such steps to do an n-bit conversion. The internal DAC can be implemented as a conventional n-stage R–$2R$ resistor ladder or, interestingly, as a set of 2^n binary-scaled capacitors; the latter method is known as a *charge-redistribution* DAC.

Voltage-to-frequency (§13.8.1) The output is a pulse train (or other waveform) whose frequency is accurately proportional to the analog input voltage. In an *asynchronous* V/F the oscillator is internal and free running. By contrast, a *synchronous* V/F requires an external source of clock pulses, gating a fraction of them through such that the *average* output frequency is proportional to the analog input.

Single-slope integration (§13.8.2) The time required for an internally generated analog ramp (capacitor charged by a current source) to go from zero volts to the analog input voltage is proportional to the value of the analog input. That time is converted to an output number by gating a fast fixed-frequency clock, and counting the number of clock pulses. Note that pulse-width modulation employs the same ramp-comparator scheme as single-slope integration to generate the ON time of each cycle.

Dual-slope and multislope integration (§§13.8.3–13.8.4, 13.8.6) These are variations on single-slope integration, effectively eliminating errors from comparator offsets and component stability. In *dual-slope integration* the capacitor is ramped up for a fixed time with a current proportional to the input signal, and ramped back down with a fixed current; the latter time interval is proportional to the analog input. In *quad-slope integration* the input is held at zero while a second such "auto-zero" cycle is done. The so-called *multislope* technique is somewhat different, with a single conversion consisting of a succession of fast dual-slope cycles (in which the input is integrated continuously, combined with subtractive fixed-current cycles), and with a correction based upon the partial-cycle residue at both ends. In some aspects it's a close cousin of the delta–sigma method.

Delta–sigma (§13.9) There are two parts: a *modulator* converts the analog input voltage to a serial *bitstream*; then a digital lowpass filter accepts that bitstream as input, producing the final n-bit digital output. Most simply (and it's never very simple!), the modulator consists of an integrator acting on the difference between the

[31] Much higher oversampling – at *many* times Nyquist rate – is exploited in the technique of delta–sigma conversion, see §13.9.

analog input voltage and the value of the 1-bit output serial bitstream to determine the next output bit. Variations include higher-order modulators (a succession of weighted integrators), or bitstreams that are several bits wide (a "wordstream"?), or both. Delta–sigma converters are both popular and confusing, and they deserve the extensive section later in the chapter.

13.6 ADCs I: Parallel ("flash") encoder

This is probably the simplest ADC concept; it's also the fastest (see Table 13.4). In this method the input signal voltage is fed simultaneously to one input of each of *n* comparators, the other inputs of which are connected to *n* equally spaced reference voltages. The output levels of the *n* comparators form a "thermometer code," which is converted (in a *priority encoder*) to a $(\log_2 n)$-bit binary output corresponding to the highest comparator activated by the input voltage. Figure 13.25 shows the idea *notionally* – here clumsily implemented with discrete comparators and standard logic. You wouldn't do that, of course; much better to let the silicon wizards do their integrated magic. In this simple (single-stage) scheme the delay time from input to output is the sum of comparator, encoder, and output latch (if provided) delays. An example of a commercial flash encoder using this scheme is the MAX1003: it does 6-bit conversions on each of two input channels, at sampling rates to 90 Msps, with the digitized and latched result available one clock cycle after sampling.

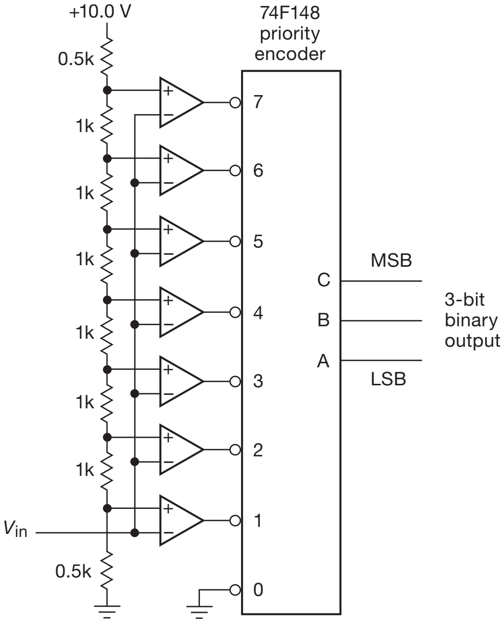


Figure 13.25. Parallel-encoded ("flash") ADC.

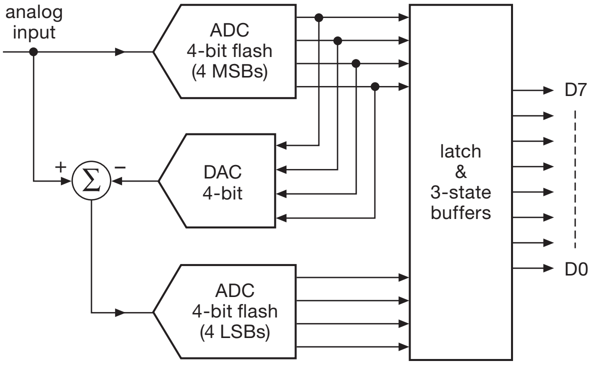


Figure 13.26. Half-flash ADC.

13.6.1 Modified flash encoders

In practice, the simple flash scheme has been largely supplanted by modified flash variants, with names like "half flash," "subranging flash," "folding/interpolating architecture," or "pipelined flash." These generally involve successive stages of partial conversion, so there is some amount of delay (or *latency*) from the moment of input sampling to valid digital output. This does not necessarily reduce the maximum sampling rate. In fact, quite the contrary: by subdividing the conversion into a succession of coarser quantizations, these converters can achieve very high sampling rates, with the partially quantized analog "residues" propagating along in a capacitor-based pipeline as newer samples begin their conversion. In such a converter an initial coarse conversion (say to 2-bit resolution) is followed by successive stages that operate on the residue (the difference between the analog input and that coarse estimate). An example is the ADC10D1500, a dual 10-bit 1.5 Gsps ADC (the two sections can be interleaved, to achieve 3 Gsps); it has a latency of 35 clock cycles.

Perhaps the simplest of these converter architectures is the *half-flash*, a two-step process in which the input is flash-converted to half the final precision; an internal DAC converts this digital approximation back to analog, where the difference "error" between it and the input is flash-converted to obtain the least significant bits (Figure 13.26). This technique yields low-cost converters that operate at relatively low power. Examples are TI's TLC0820, ADI's AD7820, and TI's TLC5540, inexpensive 8-bit ADCs with latencies of two or three clock cycles, and modest conversion speeds (40 Msps for the latter).

As remarked above, more sophisticated ADC architectures perform the conversion with various pipeline schemes, in which the analog residues are carried along through a succession of relatively coarse quantizers. An example is ADI's AD9244, which uses a 10-stage pipeline to achieve 14-bit conversions at 65 Msps, with a latency of eight clock cycles. Their AD9626 trades some accuracy for speed: 12 bits and 250 Msps, with six clocks of latency. Its datasheet says it all:

> The pipelined architecture permits the first stage to operate on a new input sample, while the remaining stages operate on preceding samples. Each stage of the pipeline, excluding the last, consists of a low resolution flash ADC connected to a switched capacitor DAC and interstage residue amplifier (MDAC). The residue amplifier magnifies the difference between the reconstructed DAC output and the flash input for the next stage of the pipeline.... The last stage simply consists of a flash ADC.

Similar technology is offered by TI's ADS5547, a 14-stage pipelined ADC that achieves 14-bit conversions at 210 Msps, with a latency of 14 clocks (these guys like the number fourteen). Their fastest high-resolution ADCs at the time of writing are 14-bit 400 Msps (ADS5474) and 16-bit 370 Msps (ADC16DX370).

The "folding" architecture (usually implemented as a combined folding/interpolating scheme) achieves a similar goal (creating the final conversion via coarse and fine quantizations), but by a clever method that does not involve a pipeline of successive steps. Rather, the analog input passes through an analog folding circuit (made from a string of cross-connected differential pairs) that maps the full input-voltage range to an output consisting of a set of repeating folds. That output is flash-converted to produce the lower-order bits, while a coarse flash conversion of the full-range input signal simultaneously determines in which fold the signal lies (i.e., the higher-order bits); see Figure 13.27. The "Ultra High Speed ADC" family from National Semiconductor uses these techniques, with current offerings going to speeds of 3.6 Gsps at 12-bit resolution (ADC12D1800). There are lots of tricks involved in making this work well; as the saying goes, they are "well beyond the scope of this book."

Flash encoders are worth considering in waveform digitizing applications even when the conversion rate is relatively slow, because their high speed (or, more precisely, their short *aperture* sampling interval) ensures that the input signal is effectively not changing during the conversion. The alternative – the slower converters we describe next – usually requires an analog sample-and-hold circuit to freeze the input waveform while conversion is going on. Note that a converter's latency may or may not matter, depending on the application: latency would not be of concern in an oscilloscope frontend, or in a "software radio"; but it would be a disaster in a fast digital control loop.

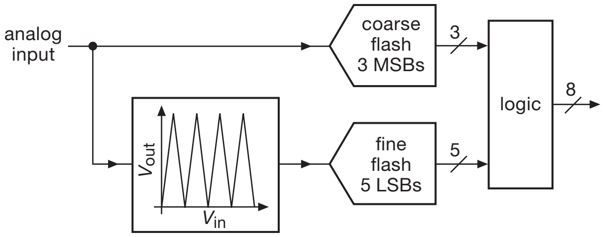


Figure 13.27. Flash ADC with "folding" architecture.

13.6.2 Driving flash, folding, and RF ADCs

Today's ADCs are not your father's simple converters. It's true they're much more capable than converters of previous generations, but they're cantankerous, they no longer observe the rule of speaking only when spoken to, and they may be very demanding of resources, especially power and digital assets. This is especially true for high-speed low-voltage converters with differential inputs. No longer can you count on simply taking an op-amp and connecting it to your ADC.

To illustrate with an example, Figure 13.28 shows a dual-channel 16-bit flash ADC capable of working up to low RF frequencies, ideal for digitizing I,Q signals for software radios. It employs a differential amplifier IC discussed in §5.17, performing the tasks shown in Figures 5.102 and 5.103. As we draw these circuits, at first glance they may look alike, but we find ourselves spending time on the details, looking up datasheet rules and suggestions, considering alternative parts with new rules, and honing the design.

The AD9269 converter[32] is a flash type, but later in the chapter we'll encounter the same issues with the other primary ADC types. For example, some successive-approximation register (SAR) converters (and many of the delta–sigma converters as well) will require the distinctive two-resistor plus capacitor configuration at their inputs. Many SAR types go further, being quite picky about

[32] The AD9269 (listed in Table 13.4) is a dual ADC with conventional CMOS logic outputs, but ADCs with slightly faster data rates generally upgrade to differential LVDS outputs, or 32 lines per 16-bit ADC channel.

Table 13.4 Selected Fast A-to-D Converters[a]

Part #	Mfg	ADC per pkg	Bits	Conv rate (Msps)		Aperture jitter (ps, typ)	Analog BW (MHz, typ)	Latency (clks)	SFDR (dB, typ)	SNR (dB, typ)	@ MHz	ENOB, typ	@ MHz	DNL (LSBs, max)	INL (LSBs, max)	Input CM to rails?	Internal Vref?	External Vref OK?	Gain		Supply Voltages		Pdiss typ (mW)	Output	Pkg	Price qty 1 (US$)	Comments
				max	min														Error (%, max)	Drift (ppm/°C, typ)	analog (+V)	digital (+V)					
ADC14L020	TI	1	14	20	5	0.7	150	7	93	74	10	12	10	1	3.8	-	●	●	3.3	2.5	3.3	$2.5–V_{ana}$	150	C	32LQFP	20	-
AD9225	ADI	1	12	25	0	1	105	3	83	70	10	11.3	10	1	2.5	B	●	●	1.7	0.4	5	3-5	280	C	28SO	25	-
ADC12040	NSC	1	12	40	0.1	1.2	100	6	84	69	10	11	10	1	1.8	N	-	●	2.1	-	5	$2.5–V_{ana}$	340	C	32LQFP	17	-
ADC14L040	TI	1	14	40	5	0.7	150	7	90	73	20	11.9	20	1	3.8	-	●	●	3.3	2.5	3.3	$2.5–V_{ana}$	235	C	32LQFP	20	-
TLC5540	TI	1	8	40	5	30	75	3	42	47	3	7.5	6	1	0.75	B	-	●	3.5	-	5	5	85	C	24SO	7	S
LTC2192	LTC	2	16	65	5	0.07	550	7	90	77	70	-	-	0.9	6	-	●	●	1.7	10	1.8	1.8	200	G	52QFN	140	-
AD9244	ADI	1	14	65	0.5	0.3	750	8	86	72	70	11.5	100	1	4	-	●	●	2	2.3	5	3-5	550	C	48LQFP	33	-
AD9269	ADI	2	16	80	3	0.1	700	9	90	76	70	12.3	70	1.7	6.5	-	●	●	2^t	2	1.8	1.8-3.3	225	C	64LFCSP	77	-
ADS5562	TI	1	16	80	1	0.09	300	16	80	83	30	13.1	10	3	8.5	-	●	●	2.5	100	3.3	3.3	865	C,L	48VQFN	76	-
AD9283	ADI	1	8	100	1	5	475	4	-	47	40	7.3	40	1.25	1.25	-	●	●	6	80	3.3	3.3	90	C	20SOP	10	A
AD9650	ADI	2	16	105	10	0.075	500	12	90	82	30	13	70	1.3	6	-	●	●	1.3	15	1.8	1.8	328	C,L	64LFCSP	170	P
AD9268	ADI	2	16	125	10	0.07	650	12	88	78	70	12	200	1.2	5.5	-	●	●	2.5	15	1.8	1.8	750	C,L	64LFCSP	130	P
ADS6245	TI	2	14	125	5	0.25	500	12	86	72	100	11.7	50	2.5	5	-	●	●	1	50	3.3	3.3	1W	O	48VQFN	100	P
LTC2165	LTC	1	16	125	1	0.07	550	6	84	76	140	-	-	0.9	6	-	●	●	1.3	10	1.8	1.2-1.8	194	C,D	48QFN	125	P
ADC1610	NXP	1	16	125	100	-	650	13.5	84	70	170	11.4	170	0.95	4^t	-	●	●	0.5^{ty}	-	3	1.8-3.3	630	C,D	40QFN	40	P
ADS5485	TI	1	16	200	10	0.08	730	5	82	75	150	12.1	10	1	10	-	●	●	6	100	V1	3.3	2.8W	D	64VQFN	130	A
ADC08200	NSC	1	8	200	20	2	500	6	54	44	100	7.3	50	0.95	1.9	B	-	●	3	-	3	$2.5–V_{ana}$	1/Msps	C	24SO	14	R,S
ADS5547	TI	1	14	210	1	0.15	800	14	70	68	400	11.8	70	2.5	5	-	●	●	2	100	3.3	3.3	1.2W	C,D	48QFN	120	P
ADS6149	TI	1	14	250	1	0.17	700	18	76	68	300	11.2	170	2	5	-	●	●	0.2^t	10	3.3	1.8	687	C,D	48VQFN	140	P
AD9626	ADI	1	12	250	40	0.2	700	6	80	64	70	10.5	70	0.6	1.7	-	●	-	1.4^t	210	1.8	1.8	364	C	56LFCSP	230	P
AD9467	ADI	1	16	250	50	0.06	900	16	90	74	300	12	300	1.3	12	-	●	●	3.5^y	360^y	V2	1.8	1.3W	L	72LFCSP	170	P
ISLA216P25	ISL	1	16	250	40	0.075	700	10	67	69	600	10.7	600	0.99	10^t	-	●	-	6^y	180^y	1.8	1.8	770	C,D	72QFN	210	P
AD9211	ADI	1	10	300	40	0.2	700	7	80	59	170	9.6	170	0.5	0.7	-	●	-	4.3^y	180^y	1.8	1.8	420	L	56LFCSP	64	P
ADS5474	TI	1	14	400	20	0.1	1.4G	3.5	80	70	230	10.9	230	1.5	3	-	●	●	5^y	200^y	V1	3.3	2.5W	L	80TQFP	190	-
ISLA112P50	ISL	1	12	500	80	0.09	1.15G	17	71	65	500	10.3	500	0.8	2	-	●	-	2^{ty}	325^y	1.8	1.8	475	C,L	72QFN	215	P
ADS5400	TI	1	12	1000	100	0.125	2.1G	7	66	58	1.2G	9.3	850	2	4.5	-	●	●	5^y	300^y	V1	3.3	2.2W	L	100TQFP	930	A
ADC08D1520	NSC	2	8	1500	200	0.4	2G	13	55	47	750	7.4	750	0.6	0.9	-	●	-	3.3	-	1.9	$1.8–V_{ana}$	2W	L	128LQFP	830	I
ADC10D1500	NSC	2	10	1500	200	0.2	3.1G	13	65	55	750	8.8	750	0.55	1.4	-	●	-	3.3	-	1.9	$1.8–V_{ana}$	3.6W	L	292BGA	1k	I
ADC12D1800	NSC	2	12	1800	300	0.2	2.8G	34	66	57	500	9	500	0.4^t	2.5^t	-	●	-	3.3	-	1.9	$1.8–V_{ana}$	4.4W	L	292BGA	-	I
HMCAD5831	ADI	1	3	26G	-	-	20G	2	27	57	19G	2.9	12G	0.2^t	0.2^t	-	-	●	-	-	–5	–3.3	4.2W	L	64QFN	-	J
Fujitsu 'Robin'	FUJ	2	8	56G	-	0.1	15G	-	-	36	17G	-	-	0.5	1	-	●	-	-	-	±1.2	3.3	4W	F	flip-chip	-	K

Notes: (a) listed in order of increasing sample rate, for the fastest member of a family; unless noted otherwise, inputs are differential, with sampling capacitors at the inputs introducing switching transients that require low drive impedance and RC input networks; all guarantee no missing codes. (m) min or max. (t) typical. (y) includes int ref.

Comments: **A:** input buffer, isolates sampling transients. **B:** both. **C:** *n*-line parallel CMOS. **D:** n/2-lane DDR LVDS/ch. **F:** 1024-lane LVDS at 440MHz/ch. **G:** 1, 2, or 4 LVDS lanes/ch. **I:** interleave for 2x sampling rate. **J:** non-interleaved. **K:** fastest ADC; 320 interleaved 175Msps SARs. **L:** *n*-lane LVDS/ch. **N:** negative only. **O:** one LVDS lane. **P:** programmable operating parameters, usually via SPI. **R:** resistor ladder and comparators. **S:** single-ended input. **V1:** 3.3V and 5V. **V2:** 1.8V and 3.3V.

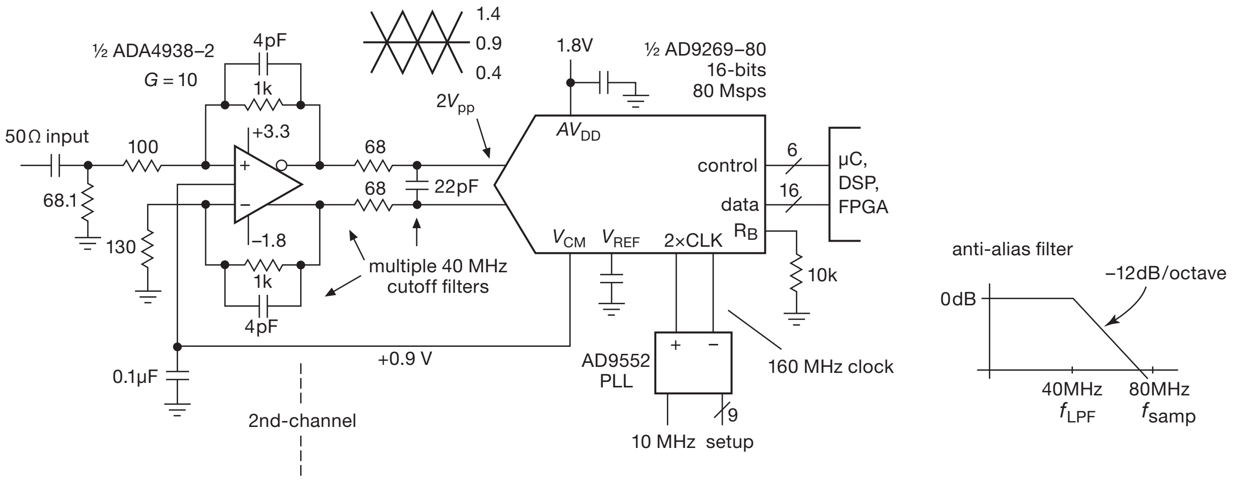


Figure 13.28. Fast ADCs are usually driven differentially, as in the two-channel 16-bit 80 Msps (40 MHz) RF digitizer. Radiofrequency applications like this require an accurate and stable clock source at some multiple of the conversion rate, here provided by a phase-locked-loop (§13.13).

what *R* and *C* values are allowed to achieve specified performance.

An 80 Msps converter has a 40 MHz Nyquist limit, set approximately with our $2R{+}C$ differential lowpass input filter. The filter R,C parts play two other roles: the ADC responds to noise (white and other) all the way up to its 700 MHz input bandwidth, so we need to quiet the output of the amplifier aggressively above 40 MHz; and the ADC's input switched-capacitor S/H needs to grab some charge from an input capacitor to work properly. The two *R*'s also serve to isolate the capacitor from the op-amp, important because 1000 MHz op-amps don't kindly tolerate direct capacitive loads. So we have three motivations for these new parts, not seen in the good old days.

Why do we use a differential-*output* amplifier? Generally, when driving a device that presents a differential input, we have the option of grounding one side and feeding the other. But doing this with today's ADCs will cost us a substantial distortion penalty, and half the full-scale input range as well. But in choosing the differential amplifier IC we were boxed into a corner: looking in the 500–1500 MHz region of Table 5.10 on page 375, we couldn't find a part with high "$Z_{\text{in(diff)}}$." We wanted additional 40 MHz anti-alias filtering, which ruled out attractive parts with internal gain-setting resistors because the summing junctions aren't exposed. And we wanted a gain of at least 10, or 20 dB. So we chose, finally, an Analog Devices ADA4938 with a 1000 MHz rated bandwidth.[33] Looking at its frequency-response plots, we see its GBW=800 MHz, thus an $f_{-3\text{dB}}$=GBW/G=80 MHz for G=10, so we have some loop gain left at 40 MHz.

Configuration-*D* amplifiers like this (Figure 5.96) have rather low input impedances, especially at high gains, because $Z_{\text{in}}=2R_{\text{g}}$, and $R_{\text{g}}=R_{\text{f}}/G$. Having thrown in the towel on high input impedances, we opt for matching the ubiquitous 50 Ω source impedance of wideband signals. If we choose R_{g}=100 Ω, the Johnson noise[34] of two of these will be 1.8 nV/$\sqrt{\text{Hz}}$, or nicely below the amplifier's rated e_{n}=2.6 nV/$\sqrt{\text{Hz}}$.

We have to provide a 50 Ω load for the input, and recognizing that Z_{in} isn't exactly R_{g},[35] we turn to the formula provided in the datasheet, $Z_{\text{in}}=R_{\text{g}}/(1-R_{\text{f}}/2[R_{\text{g}}+R_{\text{f}}])$, to determine that we need a 68 Ω load resistor; then we match the impedance seen driving the noninverting input by using a 130 Ω resistor to ground at the inverting input. This added resistance disturbs the usual $R_{\text{f}}=GR_{\text{g}}$ relationship, and we are forced to increase R_{f} by 11% to maintain $G=10$, as detailed in §5.17.4. Finally we evaluate the single-ended to differential conversion by examining the datasheet's V_{ocm} specifications. The V_{ocm} −3dB spec is 230 MHz, which means that with our feedback attenuation the full-differential ADC drive will be down by 3 dB at 24 MHz.[36]

[33] We could also have chosen a TI LMH6552 or LMH6553.

[34] See §§8.1 and 8.2.

[35] A fraction of the differential-output voltage appears at the inputs as a common-mode signal, partially bootstrapping the voltage across the input resistor R_{g}.

[36] If this is not acceptable, we need to reduce our amplifier gain by half or

Our AD9269 ADC needs a sampling clock, for which we choose the capable AD9552 PLL upconverter (see §13.13.6H and Table 13.13). It seemed a good idea to take advantage of the ADC's internal divide-by-two option, to help ensure an internal 50% duty cycle, so we'll need a 160 MHz clock for 80 Msps sampling; thus, if we use a 10 MHz reference, we set the PLL multiplication to 16. If we want other sampling rates we can employ the AD9552's powerful delta–sigma modulator fractional-frequency synthesis capabilities, and we can also choose other ADC division ratios.

A final worry (if you haven't had enough already) is *clock jitter*. The AD9269 datasheet's graphs show that, to obtain the desirable best-available 75–78 dB signal-to-noise ratio (SNR), the clock jitter should be (gasp!) no more than 0.2 ps (about 15 ppm of the sampling period). Our AD9552 PLL datasheet specifies the jitter (for a 4–80 MHz reference input to be 0.11 ps, so we slip under the wire on that one (but without much to spare).

13.6.3 Undersampling flash-converter example

Figure 13.29 shows, in somewhat simplified form, an "undersampling converter" application, in which an input signal centered at some rather high frequency (say 500 MHz) is digitized at a rate far less (say 200 Msps) than would appear necessary from the Nyquist criterion. This works successfully if two conditions are satisfied: (a) the signal's *bandwidth* must satisfy the Nyquist sampling criterion, i.e., the sampling rate must be at least twice the bandwidth occupied by the signal; and (b) the signal's full spectrum (including the high carrier frequency) must fall within the ADC's analog input bandwidth.

The first condition requires that the input signal be strictly limited in bandwidth, usually with a bandpass filter. The second condition implies that the ADC has been designed for an undersampling application. The ADC08200 in the figure, for example, specifies a full-power bandwidth of 500 MHz, even though its maximum sampling rate is 200 Msps (which would normally be appropriate for signals only to 100 MHz). You can think of this as exploiting the aliased spectrum produced by undersampling; that's OK, as long as there are no other spectral components competing for that "baseband" piece of spectrum (see Figure 13.30).[37]

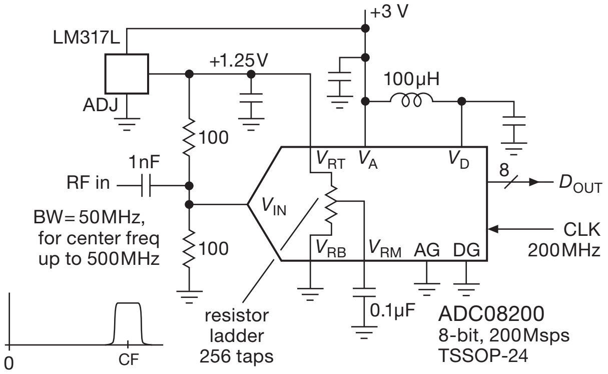


Figure 13.29. An inexpensive flash ADC digitizes a band-limited signal well above the Nyquist cutoff frequency, a job that traditionally calls for frequency downconversion with a local oscillator ("LO") and mixer.

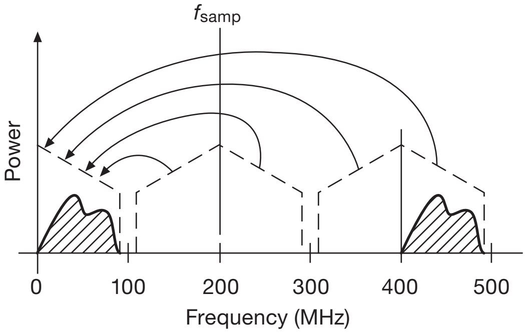


Figure 13.30. Putting the alias to work: a sampling rate of 200 Msps properly samples signals in the "baseband" extending to 100 MHz; but it creates aliases of successive 100 MHz bands. You can use this to your advantage by filtering out all input signals except those in the 400–500 MHz band (for example); that band is then properly digitized, and appears as a 0–100 MHz signal stream.

In the example circuit we've used a relatively slow member from National Semiconductor's family of flash converters. This puppy runs from a single +3 V supply, converts at rates to 200 Msps with byte-wide output through a simple parallel output port, and costs under $15 in single quantities.[38] You get to supply the top and bottom of the conversion range (here ground and +1.25 V), and the datasheet sternly advises you to bypass the midpoint of the 256-tap resistor string. The 100 μH choke decouples the analog pin from the noisy V_D digital supply pin. With the positive-only conversion range it's necessary to bias the input signal to half the conversion range, as shown; the pair

seek out a faster amplifier IC, such as an ADA4937, with a 440 MHz V_{ocm} spec.

[37] Sometimes called "super-Nyquist operation." See, for example, Application Note AN-939 from Analog Devices.

[38] A related part, the ADC08B200, includes a 1024-byte output buffer, a handy thing if you want to sample in bursts and need to read the output stream at less than full speed.

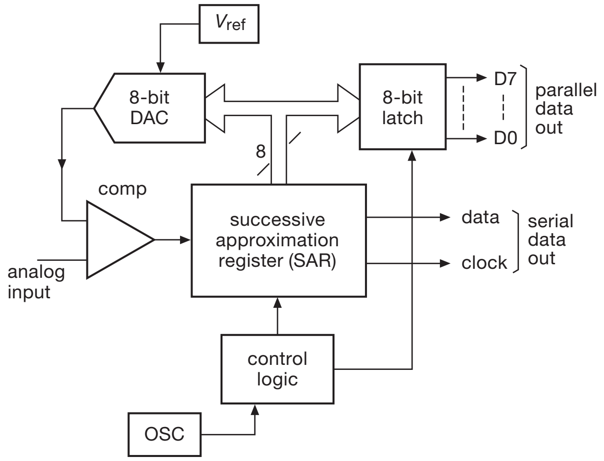


Figure 13.31. Successive-approximation ADC.

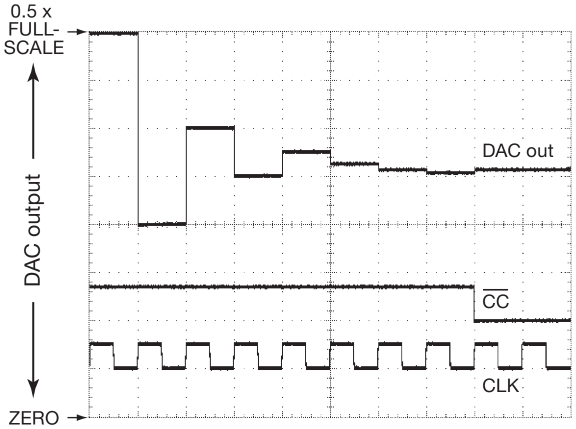


Figure 13.32. 'Scope trace of an 8-bit successive-approximation DAC's analog output converging to the final value. It's a binary search, with first guess equal to half of full scale. Note clock waveform and conversion-complete flag.

of 100 Ω resistors terminates the signal input with the usual 50 Ω expected by RF signals.

13.7 ADCs II: Successive approximation

In the classic successive-approximation technique (sometimes called "SAR") you try various output codes by feeding them into a DAC and comparing the result with the analog input present at an input comparator (Figure 13.31). The way it's usually done is to set all bits initially to 0. Then, beginning with the most significant bit, each bit in turn is set provisionally to 1. If the D/A output does not exceed the input signal voltage, the bit is left as a 1; otherwise it is set back to 0. For an n-bit ADC, n such steps are required. What you're doing is called a *binary search*, in the language of computer science.[39] A successive-approximation ADC has a BEGIN CONVERSION input and a CONVERSION DONE output. The digital output may be provided in parallel format (all bits at once, on n separate output lines), in serial format (n successive output bits, MSB first, on a single output line), or both.

In our electronics course the students construct a successive-approximation ADC, complete with DAC, comparator, and control logic. Figure 13.32 shows the successive outputs from the DAC, along with the eight clock pulses, as the trial analog output converges to the input voltage. And Figure 13.33 shows the full 8-bit "tree," a pretty picture you can generate by watching the DAC output while driving the input with a slow ramp that runs over the full analog input range.

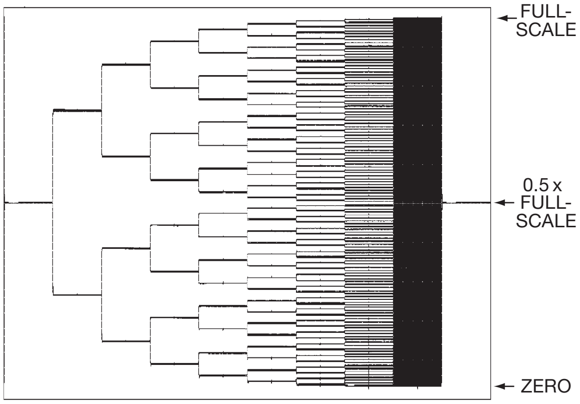


Figure 13.33. Accumulated 'scope trace of the 8-bit SAR's full "tree."

Successive-approximation ADCs are intermediate in speed and accuracy (compared with the faster flash converters, or the more accurate but slower techniques used in "delta–sigma" converters, and multislope integrating converters); see Tables 13.5 and 13.6. They require n settling times of the DAC for n-bit precision. Typical conversion times are in the vicinity of 1 μs, with accuracies of 8 to 18 bits commonly available. This type of converter operates on a brief sample of the input voltage, and if the input is changing during the conversion, the error is no greater than the change during that time; however, spikes on the input are disastrous. Although generally quite accurate,

[39] Historically this goes *way* back: in 1556 a mathematician by the name of Tartaglia proposed using a set of weights (1 lb, 2 lb, 4 lb...32 lb) in just such a manner to determine the weight of an object in the minimum number of trials on a balance.

these converters require accurately trimmed resistor networks, and they can have strange nonlinearities and "missing codes." One way to prevent missing codes is to use a chain of 2^n resistors and analog switches to generate the trial analog voltages, in the manner of the resistor-string DACs of §13.2.1; this technique was used in NSC's ADC0800-series 8-bit ADCs.

In contemporary successive-approximation ADCs, the conventional resistive DAC (*R*–2*R* or resistor string, used internally to generate the analog voltages for the trial codes) is usually replaced with a *charge-redistribution* DAC architecture (Figure 13.34).[40] This scheme requires a set of binary-weighted *capacitors*, which nowadays is easy enough to fabricate and trim on-chip. (So an 18-bit converter like the AD7641 contains, remarkably enough, a set[41] of 18 binary-scaled capacitors going from some C_0, $2C_0$, ..., to a final $131{,}072C_0$; these capacitors are really small, with the capacitance of C_0 measured in the *femto*farads – f F, or 0.001 pF.)

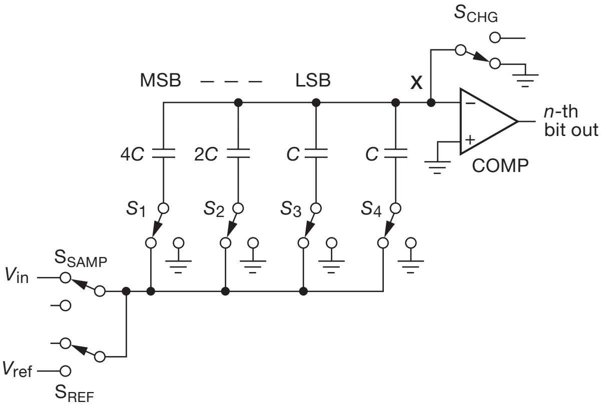


Figure 13.34. A capacitor-based "charge-redistribution" scheme replaces the *R*–2*R* resistor ladder in many successive-approximation ADCs. The capacitor beyond the LSB is not used in the bit testing, but is needed to preserve the exact fractional ratios.

To understand how it works, look at the operation of the simplified 3-bit converter in the figure.

1. The switches are shown in the *sample* part of the cycle, during which the voltage across each of the capacitors follows (or *tracks*) the input signal.
2. Switch $S_{\rm SAMP}$ is opened, leaving the capacitors all holding the sampled input voltage.
3. Switch $S_{\rm CHG}$ is then opened, so the input to the comparator can move around as trial codes are applied to the bit switches S_1–S_3; for example, if the bit switches are all set to ground, the comparator input X would be at a voltage $-V_{\rm in}$.
4. Now, to measure the held value of $V_{\rm in}$, the bit switches are operated in turn: first the MSB switch S_1 is switched to $+V_{\rm ref}$ (the full-scale range of the ADC), while S_2, S_3, and S_4 are switched to ground; this adds an offset of $V_{\rm ref}/2$ to that $-V_{\rm in}$ (it's a capacitive voltage divider: call it "charge redistribution" if you prefer).
5. The comparator's output now indicates the MSB: HIGH if $V_{\rm in}>V_{\rm ref}/2$, LOW otherwise.
6. As with the classic successive-approximation procedure, that switch is either returned to ground or left at $V_{\rm ref}$, accordingly; the next-lower bit value is then tested similarly, with the process continuing in n steps (here n=3) to determine the full n-bit converted value.

13.7.1 A simple SAR example

Successive-approximation ADCs can be extremely easy to use, as illustrated by the circuit in Figure 13.35. The SPI serial interface is simplicity itself: assertion of CS′ starts conversion, with successive bits clocked out by SCK (as each clock pulse triggers the SAR conversion of a new bit). The timing allows you to keep both serial lines quiet before assertion of CS′, as shown, to minimize coupled digital noise. This family of relatively low-speed converters integrates on-chip track-and-hold, and includes three speed grades, three resolutions (8, 10, and 12 bits), and four packaging options (single, dual, quad, and octal): 36 choices! (The figure shows how to construct the part numbers.) The single units, like the 12-bit 1 Msps specimen in the figure, come in tiny SOT23-6 packages, with prices (in single quantities) ranging from about \$2 (8 bit, 200 ksps) to \$4.50 (12 bit, 1 Msps).

The input of an ADC is often less benign than something like an op-amp, where we've come to expect a high impedance (very low input current), and low capacitance. Figure 13.36 shows the equivalent input circuit of this converter, with its 26 pF sampling capacitor that the input signal must drive. This is not much of a burden at the relatively low frequencies here; but it's something to keep in mind, for example in the circuit of Figure 13.37 with its much higher resolution (18 bits) and somewhat higher speeds.

13.7.2 Variations on successive approximation

A variation known as a "tracking ADC" uses an up–down counter to generate successive trial codes; it is slow in responding to jumps in the input signal, but it follows

[40] There are also hybrid designs, in which a charge redistribution DAC is used to subdivide the steps of a coarse resistor-string DAC.

[41] Actually, *two* such sets, because its input is differential.

Table 13.5 Selected Successive-approximation A-to-D Converters[a]

										Power		Analog Input			Supply			Pkgs, Pins					
Part #	ADCs per pkg	Bits	Conv Rate max (Msps)	Analog BW −3dB (MHz)	Channels/ADC	Single-ended	Differential	Internal Ref	Interface[k]	typ (mW)	at sps, V_S	V_{in} min (V)	V_{in} max (V)	I_{bias} max (μA)	V_S min (V)	V_S max (V)	I_{PD}[h] (μA)	SOIC	TTSOP	SOT-23	Smaller	Cost qty 25 ($US)	Comments
AD7927	1	12	0.2	8.2	8	•	-	-	S	3.6^{m}	200k, 3	0	V_{ref}^{e}	1	2.7	5.25	0.5^{m}	-	20	-	-	5.33	A
ADS7866	1	12	0.2	8	1	•	-	-	S	0.4	200k, 1.6	0	V_S	1	1.2	3.6	0.3^{m}	-	-	6	-	3.69	B
ADC121S	1	12	1	11	1	•	-	-	S	2	1M, 3	0	V_S	1	2.7	5.25	1^{t}	-	-	6	6	3.17	C
ADS7881	1	12	4	50	1	•	p	•	P	95	4M, 5	0	V_{ref}	0.5^{t}	4.75	5.25	2.5^{m}	-	48	-	48	14.52	D
MAX11131	1	12	3	50	16	•	•	-	S	15	3M	0	V_{ref}	1.5	2.4	3.6	6^{m}	-	28	-	28	12.24	E,Q
ADS7945[b]	1	14	2	15	2	-	•	-	S	20	2M, 5	0	V_{ref}	0.002	2.7	5.25	2.5^{m}	-	-	-	16	7.50	F
MAX1300[d]	1	16	0.11	0.7	8	•	•	•	S	17^{m}	100k, 5	−16	16	1250	4.75	5.25	0.5^{t}	-	24	-	-	11.00	M
LTC1609	1	16	0.2	1	1	•	-	•	S	65^{c}	200k, 5	−10	10	-	4.75	5.25	10^{t}	20	28	-	-	20.42	N
AD7685	1	16	0.25	2	1	-	•	-	S	2.7^{c}	200k, 2.5	0	V_{ref}	0.001^{t}	2.3	5.5	0.05^{m}	-	10	-	-	11.31	N
MAX11046	8	16	0.25	4	1	•	-	•	P	240	250k, 5	−5	5	1	4.75	5.25	10^{m}	-	64	-	56	19.48	O
MAX11166	1	16	0.5	6	1	•	-	•	S	26	500k, 5	−5	5	10	4.75	5.25	10^{m}	-	-	-	12	35.20	G,P
ADS8326	1	16	0.2	0.5	1	-	•	-	S	3.8^{c}	200k, 2.7	0	V_{ref}	0.05^{t}	2.7	3.6^{g}	0.1^{t}	-	8	-	-	9.88	N
AD7699	1	16	0.5	14	8	•	-	•	S	5.2	100k, 5	0	V_{ref}	0.001^{t}	4.5	5.5	0.05^{t}	-	-	-	20	12.00	H,N
ADS8319	1	16	0.5	15	1	-	•	-	S	18^{c}	500k, 5	0	V_{ref}	0.001^{t}	4.5	5.5	0.3^{m}	-	-	-	10	11.86	N,L
AD7985	1	16	2.5	19	1	•	-	•	S	28	2.5M, 5	0	V_{ref}	0.25^{t}	4.75	5.25	1^{t}	-	-	-	20	28.18	J,R
AD7690	1	18	0.4	9	1	-	•	-	S	4.3^{c}	100k, 5	0	V_{ref}	0.001^{t}	4.75	5.25	0.05^{m}	-	10	-	9	30.19	N,U
AD7982	1	18	1	9.0	1	-	•	-	S	7^{c}	1M, 2.7	0	V_{ref}	0.2^{t}	2.38	2.63	0.35^{t}	-	10	-	-	33.52	N
ADS8881	1	18	1	30	1	-	•	-	S	5.5^{c}	1M, 3	0	V_{ref}	0.005^{t}	2.7	3.6	100^{m}	-	10	-	10	36.54	N,V
LTC2379-18	1	18	1.6	34	1	-	•	-	S	18^{c}	1.6M	0	V_{ref}	1	2.38	2.63	100^{m}	-	16	-	16	42.79	K,N
AD7641	1	18	2	50	1	-	•	•	P,S	75^{c}	2M, 2.5	0	V_{ref}	18^{t}	2.37	2.62	0.6^{t}	-	48	-	48	43.56	N
AD7767-2	1	24	0.032	-	1	-	•	-	S	8.5	1M	0	V_{ref}	-	2.38	2.63	1^{t}	-	16	-	-	14.32	S
AD7767	1	24	0.128	-	1	-	•	-	S	15	1M	0	V_{ref}	-	2.38	2.63	1^{t}	-	16	-	-	14.32	T

Notes: (a) listed by accuracy and speed; all feature "no missing codes"; all permit external Vref input. (b) the ADS7946 is the same, without diff'l input. (c) power scales linearly with sample rate. (d) MAX1301 has half the number of inputs, in 20-pin TSSOP. (e) or to 2Vref, see datasheet. (f) with ext ref. (g) or 4.5-5.5V. (h) supply current during shutdown, power-down, or quiescent. (k) S=serial, P=parallel. (p) pseudo-differential.

Comments: A: sequencer; AD7928=1Msps. **B:** 10-bits='67, 8-bits='68, $1.80; 280ksps at V_S>1.6V; 8nA typ off, power-off after each conversion, 0.4μW at 100 per second, 44μW at 20ksps and V_S=1.2V. **C:** ADC121S051=500ksps, ADC121S021=200ksps. **D:** digital I/O supply 2.7V-5.25V. **E:** digital I/O supply 1.5V-3.6V. **F:** digital I/O supply 1.65V to V_S. **G:** digital I/O supply 2.3V-5.25V. **H:** digital I/O supply 1.8V to V_S. **J:** digital supply 2.4V-2.6V; digital I/O supply 1.8V-2.7V. **K:** digital I/O supply 1.7V-5.3V. **L:** digital I/O supply 2.4V-5.5V. **M:** PGA, 7 gain choices; 8 singled-ended or 4 diff'l inputs; V_{in} up to ±3V_{ref} or 6V_{ref}, or up to ±16V with V_S=5V. **N:** charge-redistribution (capacitive) SAR; power scales linearly with sample rate. **O:** 8 independent ADCs, simultaneous sampling; 6-ch=MAX11045, 4-ch=MAX11044. **P:** true "Beyond-the-Rails" without input dividers, ±5V with single V_S=+5; int V_{ref} 17ppm/°C max. **Q:** FIFO; averaging; 1, 2, 4, .. 32 channel sequencer. **R:** int ref 10ppm/°C,typ; digital supply 2.4-2.6V. **S:** 32x oversampling, on-chip FIR filter. **T:** 8x oversampling, on-chip FIR filter. **U:** 100dB min SNR, 125dB typ THD. **V:** digital I/O supply 1.65-3.6V; ADS886x are 16-bit diff'l and single-ended versions; family includes slower versions, to 100ksps.

smooth changes somewhat more rapidly than a successive-approximation converter. For large changes its slew rate is proportional to its internal clocking rate. The succession of up–down bits is itself serial, a simple form of *delta modulation*.

Another variation is *CVSD* (continuously variable-slope delta modulation), a simple scheme that is sometimes used for 1-bit serial encoding of speech, for example in wireless phones. With CVSD modulation the 1s and 0s represent steps (up or down) of the input waveform, but with the step size adaptively changing according to the past history of the wave. For example, the step size corresponding to a 1 increases if the last few bits have all been 1s, according to a preset rule. The decoder knows the rule, so it can recreate an approximate replica of the (quantized) original analog input. In the past you could get CVSD chips, but in contemporary practice this is implemented in software in a microcontroller or DSP chip.

13.7.3 An A/D conversion example

Before continuing on to the important "integrating" conversion techniques (V-to-F, multislope, and *delta–sigma*), let's look at a demanding application example using a successive-approximation ADC: a low-noise, high-stability 18-bit converter with 2 Msps conversion rate.

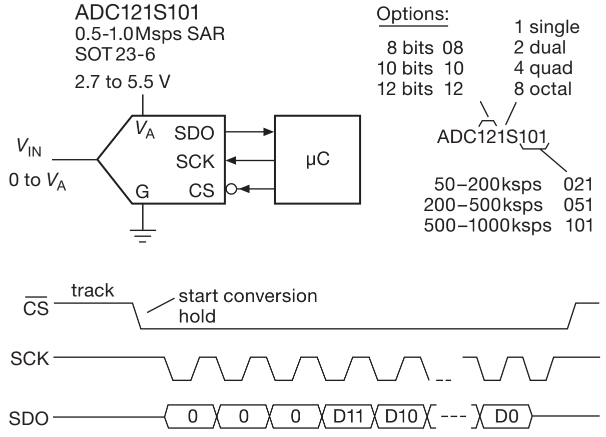


Figure 13.35. National Semiconductor's ADC08/10/12S family of simple-to-use successive-approximation ADC with SPI serial output.

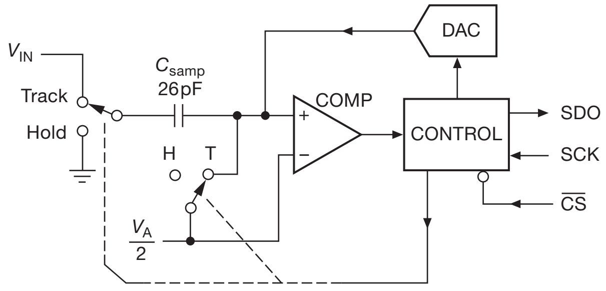


Figure 13.36. Block diagram of the ADC in Figure 13.35. The input signal drives the track-and-hold capacitor C_{samp} during acquisition.

Figure 13.37 shows a typical high-performance ADC, in this case Analog Devices' 18-bit 2 Msps PulSAR-series AD7641 converter. The AD7641 uses three positive power supplies,[42] with the nice characteristic that they may be turned on and off in any sequence.

The AD7641 has a full scale range of $\pm V_{\text{ref}}$, as is common with low-voltage ADCs. To maintain quiet conversions it's desirable to use a large voltage reference and signal voltage ranges. The AV_{DD} supply is 2.5 V, so the (differential) analog inputs are limited to 0 V to +2.5 V. If we use the maximum allowed +2.5 V reference, we get up to ±2.5 V (differential) full scale: as +IN goes from 0 V to +2.5 V, −IN will have to go from +2.5 V to 0 V (otherwise we'd have only a 17-bit converter). For an 18-bit converter this corresponds to a differential LSB step of just 19 μV. You have to take great care with such small signals, especially when the converter's sample rate is 2 MHz, and the −3dB bandwidth corresponding to its aperture time is 50 MHz – there's plenty of analog noise in these bandwidths,[43] aided and abetted by coupled digital noise from the happenings at the output end.

Both the signal and voltage reference inputs experience charge-injection pulses from the charge-redistribution conversion process, so we use sizable capacitors (the datasheet's recommended 2.7 nF) on these pins to maintain a quiet voltage.[44] Op-amps don't like direct capacitive loads, because they cause ringing in combination with the inductive closed-loop output impedance (see §4.6.2 and the section on capacitive loads in Chapter *4x*), hence the 15 Ω series resistors. This *RC* also acts as a 4 MHz low-pass filter to reduce out-of-band noise; in this bandwidth an LSB corresponds to a more relaxed noise density of 9.6 nV/$\sqrt{\text{Hz}}$. Note that the series resistor in the V_{ref} path is larger (120 Ω) because we need to limit the peak current during power-supply startup, and the dc reference does not need the bandwidth of the signal paths.

The circuit shows an amplifier setup optimized for wideband operation with a single-ended input in the range 0 V to +1.25 V. The AD8021 is a wideband low-noise op-amp that is suggested in the ADC's datasheet. This may not be the best part to use,[45] but we'll continue our narrative with the manufacturer's suggested op-amp. The amplifier pair generates an accurate unipolarity differential output from the unipolarity single-ended input: the top stage has a noninverting voltage gain of +2, and the bottom inverting stage a gain of −2. Note the low resistor values, to maintain bandwidth and also reduce Johnson noise. To help ensure equal time delays, separate signal paths are used, instead of the alternative of cascaded amplifiers. Note how the inverting op-amp is biased at $V_{\text{ref}}/3$ to create the desired +2.5 V to 0 V signal. The two op-amp pathways have different noise gains, but the AD8021 lets us add a 10 pF capacitor to its compensation node to rolloff the response to achieve approximately equal bandwidths. To deal with the op-amp's high 7.5 μA input bias current, we've rigged

[42] Separate +2.5 V pins for the analog and digital sections, and a digital I/O pin that accepts +2.3 V to +3.6 V. Low-voltage ICs often need several supply voltages, requiring separate regulators.

[43] In a 50 MHz bandwidth, 19 μV rms noise corresponds to a noise density e_n of just 2.7 nV/$\sqrt{\text{Hz}}$.

[44] Consider what's going on inside this successive-approximation ADC when operating at its full "warp-mode" speed of 2 Msps: its comparator has to make a new 19 μV decision every ~20 ns. Hectic!

[45] The choice is a bit curious, because this op-amp is not "precision" – its maximum offset voltage is an unimpressive 1 mV, and its input bias current is a rather high 7.5 μA – evidently design tradeoffs needed to achieve far more speed than is needed here (100 MHz bandwidth, 20 ns settling time).

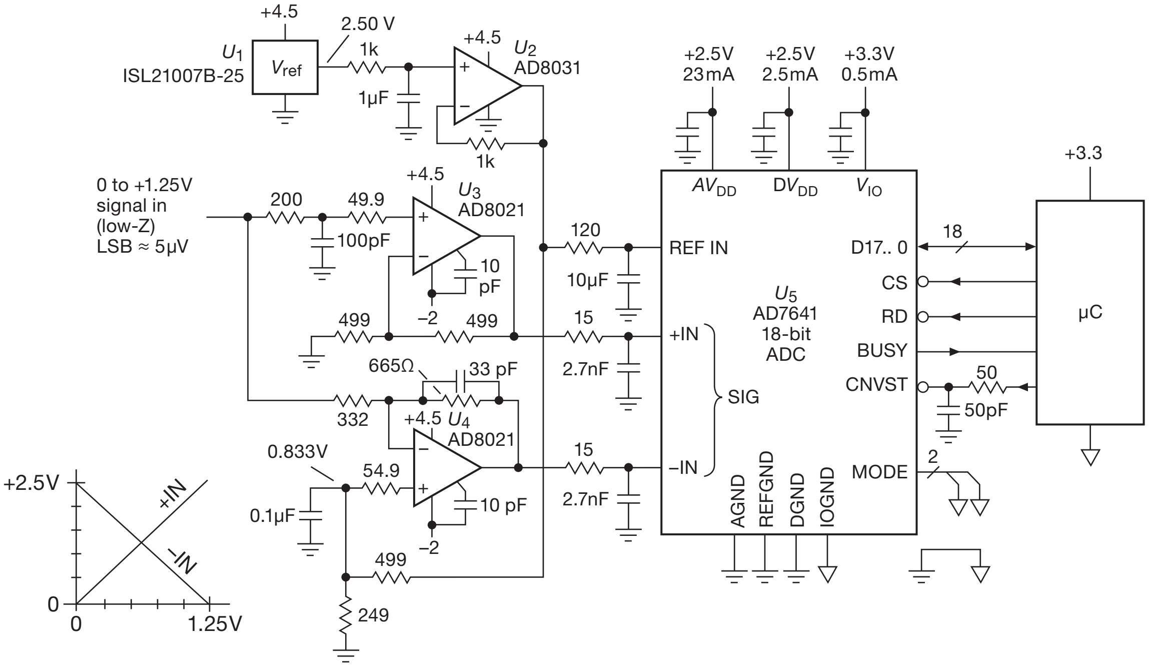


Figure 13.37. The AD7641 18-bit successive-approximation ADC, configured with fast op-amps for 2 Msps conversions.

up equal dc resistances seen at the inverting and noninverting inputs: this is effective here, because the typical offset current (0.1 μA) is 75 times smaller than the bias current itself.

The Intersil ISL21007/9BFB825 voltage reference (Table 9.8 on page 678) exploits floating-gate technology (i.e., a buried charged capacitor, §9.10.4) to achieve remarkably low drift over time ($<$10 ppm/$\sqrt{\text{kHr}}$). It has excellent initial accuracy (0.02%), and low tempco (3 ppm/°C). We've added a noise-quieting filter, and an op-amp is used to buffer the 3.3 mA load current to minimize power dissipation in the reference IC. The op-amps are powered from +4.5 V and −2.0 V, derived from the same ±5 V power that provides the regulated dc for the ADC (Figure 13.38), so that the op-amps are powered up at the same time as the ADC, thus minimizing clamp currents in the converter's input diodes at startup. Another way to prevent ADC input overdrive is to use a clamping op-amp like the AD8036, but this part has even larger dc errors than the AD8021. But there's a nice solution here, namely to clamp the C_{COMP} pin of the AD8021 op-amp with a pair of low-capacitance (2 pF) SD101 Schottky diodes, one to ground and the other to the ADC's +2.5 V supply, as shown in Figure 13.39.[46]

The AD7641 converter is shown in its 18-bit parallel data mode, selecting by grounding both MODE pins.[47] The CNVST′ start-conversion signal is *RC* filtered (2.5 ns) to slow its fall-time and help prevent undershoot, etc., as suggested by Analog Devices. The CNVST′ signal should not return to the HIGH state until conversion is finished, about 400 ns in its high-speed "warp" mode.

13.8 ADCs III: integrating

13.8.1 Voltage-to-frequency conversion

We continue our tour of A/D conversion techniques with the V-to-F (or V/F) converter. In this method an analog input voltage is converted to an output pulse train whose frequency is proportional to the input level. This can be done simply by charging a capacitor with a current proportional to the input level and discharging it when the ramp reaches a preset threshold. For greater accuracy, a feedback method is generally used. In one technique you compare

[46] The AD8021 datasheet does not tell you about this trick. But it does show a simplified schematic, from which you can see that the signal at the C_{COMP} pin is the (high-impedance) output of the gain stages, on its way to the zero-offset complementary emitter followers that form the output stage (Figure 13.39); i.e., it is a clampable high-*Z* replica of the output signal.

[47] The other choices are 16-bit parallel (two READ cycles), 8-bit parallel (three READ cycles), or SPI (clocked out over 18 clock cycles).

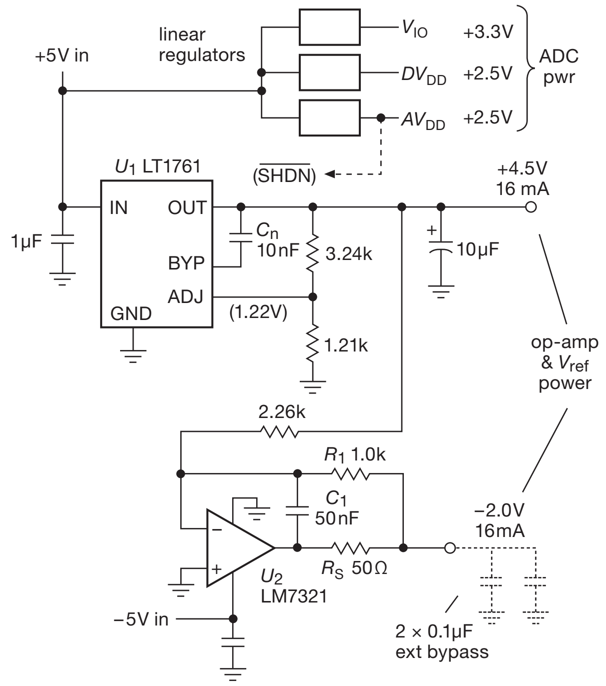


Figure 13.38. Linear regulators provide low-noise dc for the op-amps and ADC. The LM7321 is a high-current op-amp, good for 50 mA of output current. The split-feedback path (crossover at ∼3 kHz) keeps it stable into the capacitive load of the op-amps' bypass capacitors while maintaining dc accuracy.

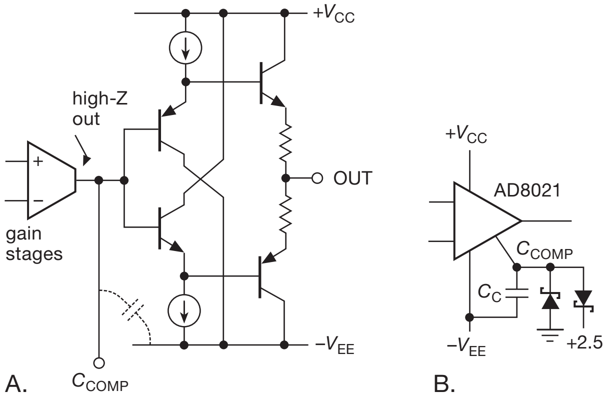


Figure 13.39. The compensation pin of some op-amps can be used to clamp the output signal. A. The AD8021's output stage is a push–pull arrangement of "zero-offset" complementary followers, with the C_{COMP} pin connected at the high-impedance output of the high-gain transconductance stages. B. Clamping that node with Schottky diodes limits the output swing to the reference voltages, here set to the ADC's conversion range.

the output of an F/V circuit with the analog input level and generate pulses at a rate sufficient to bring the comparator inputs to the same level. In the more popular methods, a "charge-balancing" technique is used, as will be described in greater detail later (in particular, the "capacitor-stored charge-dispensing" method).

Typical V/F output frequencies are in the range 10 kHz to 1 MHz for full-scale input voltage. Commercial V/F converters are available with the equivalent of 13-bit resolution (0.01% accuracy); they are examples of high-quality voltage-controlled oscillators (§7.1.4D). For example, the excellent AD650 from Analog Devices has a typical nonlinearity of 0.002% when operating from 0 to 10 kHz. They are inexpensive, and they are handy when the output is to be transmitted digitally over cables or when an output frequency (rather than digital code) is desired. If speed isn't important, you can get a digital count proportional to the average input level by counting the output frequency for a fixed time interval. This technique is popular in moderate-accuracy (3-digit) digital panel meters.

A VCO like the AD650 is an *asynchronous* V-to-F converter: Its oscillation is free running and internally generated, and it has no clocking input. But you can do things differently, namely by having a clock input and gating through clocking pulses such that the *average rate* coming out is proportional to an analog input voltage. For such a *synchronous* V/F converter, the output pulses, when present, occur coincident with the input clock; but the pulses are present or absent, as needed to keep their average rate proportional to V_{in}. In general the pulses are not equally spaced (though their spacings are exact multiples of the input clock period); that is, you don't get a single frequency out. The pulse train has "jitter." That's fine for some applications, particularly those that inherently average the output; an example is a resistive heater element, perhaps within a temperature controlled loop with an analog temperature sensor.

We rigged up an AD7741 synchronous V/F converter, clocked at 5 MHz, and measured its output frequency (averaged over several seconds) versus input voltage. It's pretty good (Figure 13.40).

The synchronous V/F converter is a simple example of a "1-bit" ADC. There are better ways to generate a bitstream whose average value represents the conversion of an analog input signal. In particular, the so-called *delta–sigma* converters do a far better job. Better job, but quite a bit harder to wrap your brain around. We'll get to these shortly, in §13.9, where we'll try valiantly (but perhaps unsuccessfully) to deconfuse the situation.

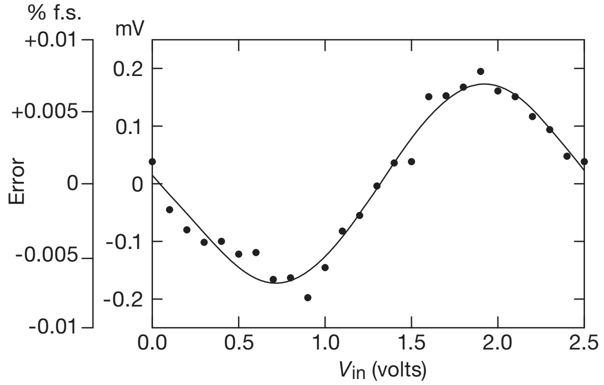


Figure 13.40. Measured nonlinearity of an AD7741 synchronous V/F converter as a function of input voltage. The specified linearity is ±0.015%.

13.8.2 Single-slope integration

In this technique an internal ramp generator (current source + capacitor) is started to begin conversion, and at the same time a counter is enabled to count pulses from a stable clock. When the ramp voltage equals the input level, a comparator stops the counter; the count is proportional to the input level, i.e., it's the digital output. Figure 13.41 shows the idea.

At the end of the conversion the circuit discharges the capacitor and resets the counter, and the converter is ready for another cycle. Single-slope integration is simple, but it is not used where high accuracy is required because it puts severe requirements on the stability and accuracy of the capacitor and comparator. The method of "dual-slope integration" eliminates that problem (and several others as well) and is now generally used where precision is required.

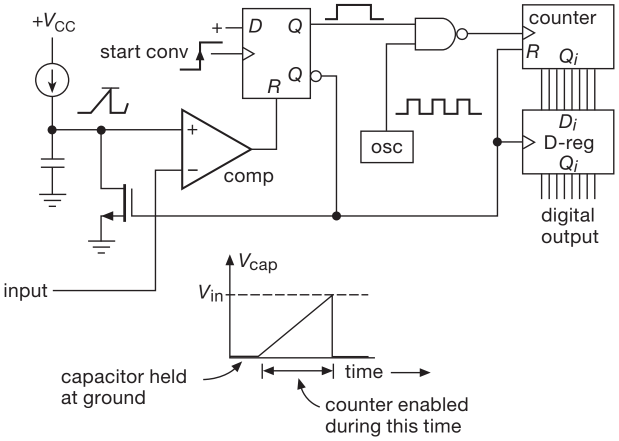


Figure 13.41. Single-slope ADC.

Single-slope integration is still alive and well, particularly in applications that don't require absolute accuracy, but rather need conversion with good resolution and uniform spacing of adjacent levels. A good example is pulse-height analysis, where the amplitude of a pulse is held (peak detector) and converted to an address. Channel-width equality is essential for this application, for which a successive-approximation converter would be totally unsuitable. The technique of single-slope integration is also used in time-to-amplitude conversion (TAC).

13.8.3 Integrating converters

There are several techniques that have in common the use of a capacitor to keep track of the ratio of an input signal level to a reference. These methods all average (integrate) the input signal for a fixed time interval for a single measurement. There are two important advantages.

1. Because these methods use the same capacitor for the signal and reference, they are relatively forgiving of capacitor stability and accuracy. These methods also make fewer demands on the comparator. The result is better accuracy for equivalent-quality components, or equivalent accuracy at reduced cost.
2. The output is proportional to the *average* input voltage over the (fixed) integration time. By choosing that time interval to be a multiple of the powerline period, the converter becomes insensitive to 60 Hz powerline "hum" (and its harmonics) on the input signal (Figure 13.42).

This nulling of 60 Hz interference requires accurate control of the integration time, since an error of even a fraction of a percent in the clock timing will result in incomplete cancellation of hum. One possibility is to use a crystal oscillator. An elegant alternative is the use of a *phase-locked loop* (§13.13) to synchronize the workings of an integrating converter to a multiple of the powerline frequency, making the rejection perfect.

These integrating techniques have the disadvantage of slow speed, as compared with successive approximation; but they excel in precision, particularly in dual-slope or multislope incarnations, or as sophisticated delta–sigma converters (§13.9).

13.8.4 Dual-slope integration

This elegant and very popular technique eliminates most of the capacitor and comparator problems inherent in single-slope integration. Figure 13.43 shows the idea. First, a current accurately proportional to the input level charges a

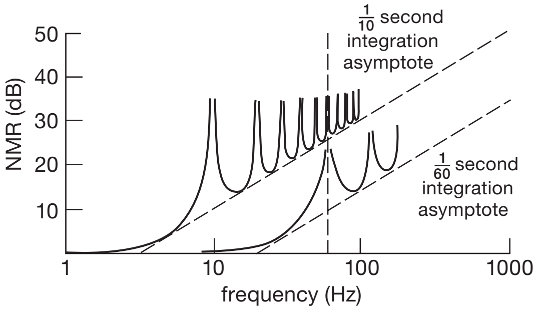


Figure 13.42. Normal-mode rejection with integrating A/D converters.

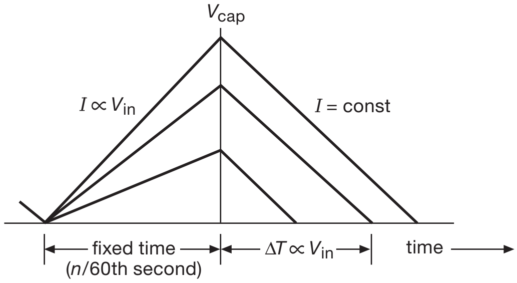


Figure 13.43. Dual-slope conversion cycle.

capacitor for a fixed time interval; then the capacitor is discharged by a constant current until the voltage again reaches zero. The time to discharge the capacitor is proportional to the input level and is used to enable a counter driven from a clock running at a fixed frequency. The final count is proportional to the input level, i.e., it's the digital output.

Dual-slope integration achieves very good accuracy without putting extreme requirements on component stability. In particular, the capacitor value doesn't have to be particularly stable, because the charge cycle and the discharge cycle both go at a rate inversely proportional to C. Likewise, drifts or scale errors in the comparator are cancelled out by beginning and ending each conversion cycle at the same voltage and, in some cases, at the same slope. In the most accurate converters, the conversion cycle is preceded by an auto-zeroing cycle in which the input is held at zero. Because the same integrator and comparator are used during this phase, subtracting the resulting "zero-error" output from the subsequent measurement results in effective cancellation of errors associated with measurements near zero; however, it does not correct for errors in overall scale.

Note that even the clock frequency does not have to be of high stability in dual-slope conversion, because the fixed integration time during the first phase of the measurement is generated by subdivision from the same clock used to increment the counter. If the clock slows down by 10%, the initial ramp will go 10% higher than normal, requiring 10% longer ramp-down time. Since that's measured in clock ticks that are 10% slower than normal, the final count will be the same! Only the discharge current has to be of high stability in a dual-slope converter with internal auto-zeroing. Precision voltage and current references are relatively easy to produce, and the (adjustable) reference current sets the scale factor in this type of converter.

When choosing components for dual-slope conversion, be sure to use a high-quality capacitor with minimum dielectric absorption ("memory" effect; see §5.6.2 and the expanded discussion in §*1x.3*) – polypropylene, polystyrene, or Teflon capacitors work best. Although these capacitors are not polarized, you should connect the outside foil (indicated with a band) to the low-impedance point (the output of the integrator op-amp). To minimize errors, choose integrator R and C values to use nearly the full analog range of the integrator. A high clock frequency improves resolution, although you gain little once the clock period becomes shorter than the comparator response time.

When using precision dual-slope converters (and, indeed, any kind of precision converter) it is essential to keep digital noise out of the analog signal path. Converters usually provide separate "analog ground" and "digital ground" pins for this purpose. It is often wise to buffer the digital outputs (say with a '541 three-state octal driver, asserted only when reading the output) to decouple the converter from the digital roar of a microprocessor bus (see Chapters 14 and 15). In extreme cases you might use opto-couplers (§12.7) to quarantine the noise of a particularly dirty bus. Be sure to use liberal power-supply bypassing right at the converter chip. And be careful not to introduce noise during the critical endpoint of the integration, as the ramp reaches the comparator trip point. For example, some converters let you check for end-of-conversion by reading the output word: *don't do it!*[48] Instead, use the separate BUSY line, suitably isolated.

Dual-slope integration is used extensively in precision digital multimeters. It offers good accuracy and high stability at low cost, combined with excellent rejection of powerline (and other) interference, for applications where speed is not important. The digital-output codes are strictly monotonic with increasing input.

The alternative, for highest precision, is the delta–sigma

[48] OK, if you insist, go ahead and check it – but only after you're sure it's done.

Table 13.6 Selected Micropower A-to-D Converters[a]

								Supply Current,[c] I_S													
Part #	Bits	Conv Rate max (ksps)	Channels	Delta-Sigma	Successive Approx[b]	Differential	PGA Gains	at V_S (V)	typ (μA)	@speed (sps)	@ max speed typ (μA)	Power at 10sps[g] (μW)	Auto Sleep (uA)	Interface[d]	Internal Ref	Ext Ref	Internal Osc	Ext osc	Cost[f] qty 100 (US$)	Package[h]	Comments
MCP3021	10	22	1	-	•	-	-	2.7	17	5k	175	0.092	0.01	I	-	V_S	-	I^{2}C	0.81	SOT23-5	-
ADS7866	12	200	1	-	•	-	-	1.8	6.3	5k	275	0.023	0.01	S	-	V_S	-	SPI	2.75	SOT23-6	-
ADS7466	12	200	1	-	•	-	-	1.6	3.8	5k	150	0.012	0.01	S	-	V_S	-	SPI	4.80	SOT23-6	-
ADC121S	12	1000	1	-	•	-	-	2.7	15	5k	600	0.081	0.5	S	-	V_S	-	SPI	2.30	SOT23-6	-
AD7091R	12	1000	1	-	•	-	-	3.0	57	20k	350	0.086	0.26	S	1%	•	•	-	4.10	MSOP-10	-
ADS1100	16	0.128	1	•	-	•	1-8	3.0	70	128	70	16.4	0.05	I	-	V_S	•	-	4.37	SOT23-6	-
MCP3425	16	0.24	1	•	-	•	1-8	3.0	0.6	1[e]	145	18	0.1	I	0.05%	-	36%	-	1.65	SOT23-6	w
ADS8326	16	250	1	-	•	•	-	2.7	150	20k	1850	0.20	0.1	S	-	•	-	SPI	9.10	MSOP-8	-
AD7685	16	250	1	-	•	•	-	2.5	0.6	100	1350	0.15	0.01	S	-	•	•	-	8.20	MSOP-10	-
LTC2379-18	18	1600	1	-	•	•	-	2.5	25	5k	7200	0.13	0.9	S	-	•	•	-	36.37	MSOP-16	x
MCP3551	22	0.014	1	•	-	•	-	2.7	100	14	100	190	10	S	-	•	1%	-	3.01	SOIC-8	-
LTC2412	24	0.008	2	•	-	•	-	2.7	170	8	170	570	1.5	S	-	•	2%	SPI	5.05	SSOP-16	y
MAX11210	24	0.48	1	•	-	•	1-16	3.6	235	480	235	18	0.4	S	-	diff	•	•	3.29	QSOP-16	z

Notes: (a) sorted by resolution and maximum speed. (b) all SAR types have S/H or T/H, and no pipeline delay. (c) most SAR types have power proportional to sample rate. (d) I=I^{2}C, S=SPI. (e) in 12-bit mode (10μA for 16-bits). (f) cheapest grade. (g) assuming linearity, and at same V_S as listed for I_S. (h) other packages may be available, consult datasheets. (w) MCP3422 = 18 bits. (x) has digital gain compression, see datasheet. (y) no latency, ping-pong channel selection. (z) four I/O bits, can use for external MUX.

converter. There's a lot of confusion surrounding this elegant technique. In a subsequent section (§13.9) we aim to blow away the smoke and provide some intuition into the workings of these things. First, though, a look at the ultimate in integrating converters – the so-called "multislope" technique devised by Hewlett–Packard (subsequently Agilent, now Keysight), and commercialized in their world-class $8\frac{1}{2}$-digit multimeters – preceded by a relevant detour into the use of analog switches in conversion applications.

13.8.5 Analog switches in conversion applications (a detour)

Analog switches, first seen in §3.4.1, figure importantly in conversion applications, both as components of the converter itself (see, e.g., Figures 13.2, 13.9,, 13.34, and 13.36), and as external helpers. In their former role they are an essential part of the precision multislope converter (§13.8.6), and of the delta–sigma converter (§13.9). Here we explore briefly some converter applications in which a discrete logic-family CMOS analog switch is particularly useful.

A. Logic-family analog switches

The widely available '4051 to '4053 family of CMOS switches is particularly useful for analog applications, because these parts have a negative V_{EE} supply line for the switches, along with internal logic-level shifters; so the switches work over an analog range of $-V_{EE}$ to $+V_{DD}$, and in fact an additional 0.25 V or so beyond these supply rails. Table 13.7 shows the logic families in which these switches are available. There are three parts in the family: the '4053 is especially attractive, with its three independently controlled SPDT switches; there's also the '4052 with a pair of 4-to-1-line switches, and the 4051 with a single 8-to-1 switch. Although these switches are attractive because they are inexpensive (less than $0.50) and available from a half-dozen companies, they're even more attractive to designers because they're very fast, and have low capacitance.

For example, the 74HC4053 typically has 40 Ω of ON resistance, switches in 20 ns, and has 8 pF of capacitance to ground. Compared with ICs officially intended for analog switching and multiplexing, a '4053 has a more limited voltage swing, and no electrostatic discharge (ESD) protection. When compared with CMOS power switches, it has higher ON resistance, but it doesn't suffer from their high capacitance. It's in a sweet spot that's ideally suited for switching signals between the circuits on the same circuit board.

Single SPDT versions are available in space-saving SOT23 and other SMT packages. These parts, for example the '1G3157 types (the 1G means single gate), do not

Table 13.7 4053-style SPDT Switches

	Supply[m]		TTL in?[a]	$t_{d(on)}$[f]	R_{ON}	ΔR_{ON}	@	Cap	leakage			avail in DIP?
	V_{CC}	V_{tot}[c]		typ	typ	typ	V_{tot}[c]	typ[b]	typ	max[e]	Q_{inj}	
	(V)	(V)		(ns)	(Ω)	(Ω)	(V)	(pF)	(pA)	(nA)	(pC)	
16-pin triple												
CD4053	3-15	20	-	120	120	10	10	8	10	50		●
74HC4053	2-7.5	15	●	18	40	8	9	5[d]		100		●
74VHC4053	2-7.5	15	-	18	20	10	9	30[d]		500		●
74VHC4053[o]	2-7	14	-	49	120	10[m]	9	45[d]		100		●
DG4053A	2-7	14	●	31	66	3	10	4	20	1	0.25	-
MAX4053A	3-17	17	●	50	60	6[m]	10	2	2	0.1	2	●
ISL84053	2-15	15	●	75	60	6[m]	10	9	2	0.1	2	-
ADG633	2-6	13	●	70	52	0.8	9	7	5	0.2	2	-
MAX4583	2-12	13	●	90	80	4[m]	10	6		1	0.5	●
74LV4053	2-7	14	-	4.6	35	1.3	9	8[d]		100		●
74LVX4053	2-7	7	-	23[m]	26	10[m]	6	10		100	9	-
NLAS4053	1.7-7	7	-	23[m]	26	10[m]	6	10		0.1	9	-
MAX4693[g]	2-11	11	●	55	25	2.5	10	20		2	1.8	-
MAX4619	2-6	na	●	7	10	1	5	8.5	2	1	8	●
MAX4783	1.6-4.6	na	●	17	1	0.2	3	75	2	2	20	-
SMT, single												
ISL84544	2-15	na	●	35	30	0.8	5	8	10	0.1	1	●
74LVC2G53	1.7-6	na	-	2	13	3[m]	3	10		100		na
74AUC2G53	0.8-3.6	na	●	1.1	15	1[m]	2.3	4.5		100		na
74LVC1G3157	1.7-6	na	-	3.6	9	0.1	3	6[f]		2μA	4	na
NX3L1G3157	1.4-4.6	na	●	14	0.5	0.02	2.7	35		10	9	na

Notes: (a) TTL logic levels, or TTL available. (b) OFF, common node to gnd. (c) $V_{tot} = V_{CC}–V_{EE}$. (d) depends on manufacturer. (e) at 25°C if shown, reflects ATE capability. (f) delay time from logic input to switch ON, at V_{tot}; $t_{d(OFF)}$ is slightly less. (g) in 16-pin QFN pkg; quad also available. (m) maximum. (na) no V_{EE} pin. (o) ON Semi.

include the negative supply feature, so they do not use "4053" in their name.

Let's look at two examples in which '4053-style switches form a nice bridge between the analog and digital worlds. The second example (sawtooth generation with current-steering switches) will take us directly to the multislope and delta–sigma converters.

B. Programmable high-voltage pulse generator

It's nice to be able to generate a pulse that is gated by a logic signal, but whose amplitude is set separately. For the latter you might use a DAC under computer control, or maybe just a panel knob.[49] The simple circuit in Figure 13.44 does the job, in this case allowing output amplitudes up to +100 V.

The '4053-style analog switch applies the level selected by panel switch S_1 to the OPA454 high-voltage op-amp, here configured for a noninverting gain of 20. This op-amp is not terribly fast (∼ 10 μs switching times), but it's quite inexpensive ($5), and it can supply 100 mA pulsed outputs to charge capacitive loads. You could substitute a faster op-amp to exploit the fast switching capability (∼ 20 ns) of the analog switch.[50] Center-off panel toggle switch S_2 lets you turn pulses on or off, or enable continuous switch conduction for reading and setting the high-voltage (HV) level with a DMM, etc. We are fond of easy-to-use instruments that combine panel controls with CMOS signal switching.

C. Current-steering sawtooth generator

Here's a circuit (Figure 13.45) that exploits the nice switching characteristics of '4053-style analog switches, in the

[49] Back in Chapter 5 we showed a way to generate programmable high-voltage waveforms (Figure 5.47), however without the gating feature.

[50] For example, the Cirrus/Apex PA85 slews at 1000 V/μs (with closed-loop gain of 100), and can run from 450 V total supply. Here you could power it from +400 V and −15 V rails. Be prepared to shell out some serious cash, though: this puppy costs about $300. (See the selection of high-voltage op-amps in Table 4.2b.) A more economical solution is to build your own high-voltage amplifier, along the lines of Figure 3.111; see also the section on a precision high-voltage amplifier in Chapter *4x*.

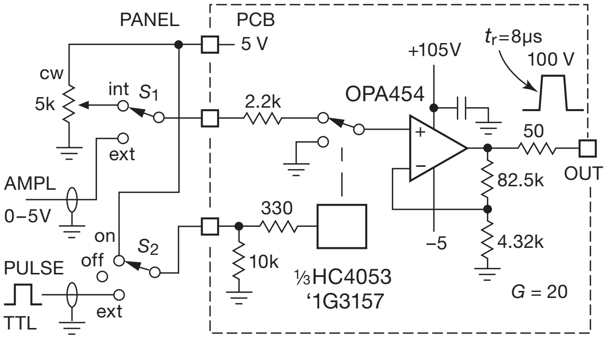


Figure 13.44. Simple high-voltage pulse generator, with programmable pulse amplitude and waveform. The '4053 switch includes a $-V_{\rm EE}$ pin for bipolarity signal switching (to ±5 V), whereas the single-section '3157 operates with positive polarities only.

same current-steering arrangement that's used in the remarkable multislope converter we'll see in the next subsection. Op-amp U_1 is an integrator, with its summing junction biased at half the supply voltage (for single +5 V supply operation). Switches S_1 and S_2 are sections of an 'HC4053, running at the same +5 V; they individually program the rising and falling ramp rates, set by resistors R_1 and R_2. Closing S_1 sources a current $V_{\rm cc}/2R_1$, causing the integrator to ramp down according to $dV_{\rm ramp}/dt = I/C$; S_2 causes an analogous upward ramp. The comparator has thresholds at 1/3 and 2/3 of $V_{\rm cc}$, turning the ramp around after it goes through $\Delta V = V_{\rm cc}/3$. It's easy to show that the resulting ramp intervals are given by $t_{\rm rise} = \frac{2}{3}R_2C$ and $t_{\rm fall} = \frac{2}{3}R_1C$, and $f = 1.5/C(R_1+R_2)$.

Exercise 13.4. Go ahead, show it!

Both the switches and the comparator are fast, permitting operation to at least a few megahertz, for which suitable values might be resistors of a few thousand ohms, and C in the range of 100–500 pF. With such a small integrator capacitor, you have to worry about the effects of the switch capacitance $C_{\rm sw}$, typically in the range of 5–10 pF. Consider for example switch S_1, in the position shown in the figure: its capacitance is charged to +5 V, and so it transfers a slug of charge $\Delta Q = C_{\rm sw}\Delta V$ (where $\Delta V = V_{\rm cc}/2$) to the summing junction when the switch moves to the lower terminal. That charge transfer causes a step in the integrator's output, as in Figure 13.46. The cure here (and in the multislope ADC we'll see next) is to hold the other switch terminal at the same voltage as the summing junction (the dotted circuit).

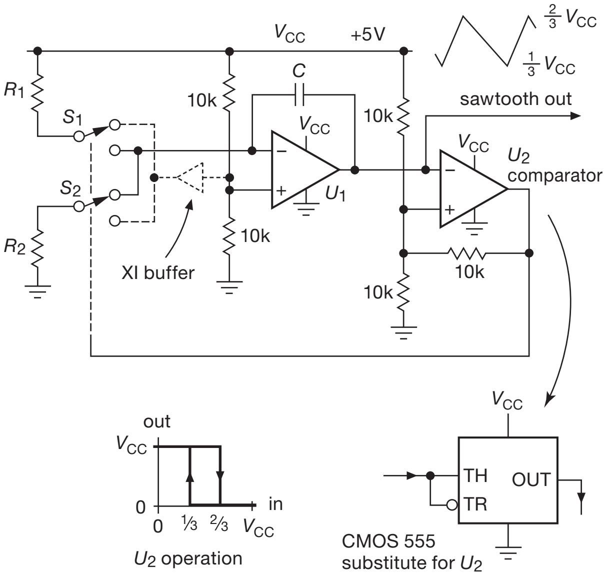


Figure 13.45. Sawtooth generation with current-steering switches. Comparator U_2 is configured as a Schmitt trigger with thresholds at $V_{\rm cc}/3$ and $2V_{\rm cc}/3$, for which a suitable part (with active rail-to-rail outputs) is the fast TLV3501 ($t_{\rm p}$=4.5 ns); a CMOS 555 could be substituted, though it's not nearly as fast ($t_{\rm p}$=100 ns).

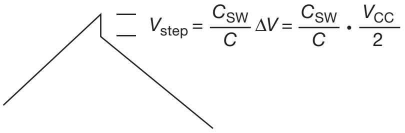

$V_{\rm step} = \frac{C_{\rm SW}}{C}\Delta V = \frac{C_{\rm SW}}{C} \cdot \frac{V_{\rm CC}}{2}$

Figure 13.46. Charge injection produces a step change in integrator output, when a circuit node (of capacitance $C_{\rm sw}$) at a different voltage is switched on.

13.8.6 Designs by the masters: Agilent's world-class "multislope" converters

With these analog-switch applications in mind, we're in good shape to understand the "multislope" techniques used in instruments like the Keysight[51] 34420 $7\frac{1}{2}$-digit and 3458A $8\frac{1}{2}$-digit multimeters.[52] This has been Agilent's top-of-the-line instrument for more than twenty years, with a current price (year 2015) of \$9.5k. A simplified variant ("Multislope III") is used in Keysight's contemporary high-performance DMM series of instruments (the 34420A 7.5-digit nanovolt and micro-ohm meter, the 34401A 6.5-digit

[51] Formerly Agilent, 1999–2014, and prior to that Hewlett–Packard or "HP," 1939–1999.

[52] The latter described on their website as "Recognized the world over as the standard in high performance DMMs."

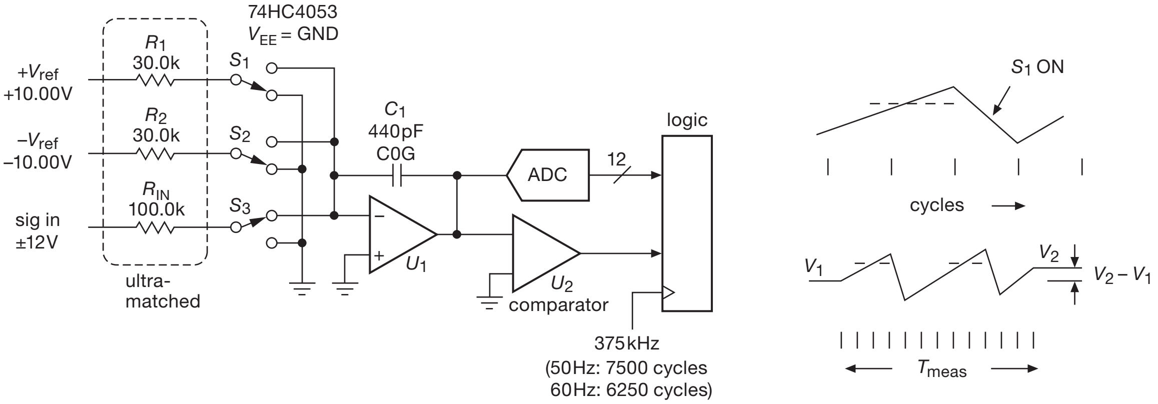


Figure 13.47. Keysight's "Multislope III" converter, a clocked charge-balancing integrator with endpoint correction via a low-precision ADC.

industry-standard bench DMM, and the 34970A 6.5-digit data-acquisition system). We'll see in detail how Multislope III works, and reveal a bit of the operation of next-generation Multislope IV (introduced in 2006).

A. The basic technique

To put it in perspective, the multislope technique is an evolution of the dual-slope integrator, exploiting a multicycle charge-balancing integration scheme that is both more forgiving of capacitor imperfections, and that takes account of the residue remaining after the final cycle of integration. It blends aspects of dual-slope and delta–sigma conversion, and it is a natural stepping-stone to the latter.

The basic circuit is extremely simple (Figure 13.47) and uses mostly low-cost parts (except for the voltage references and precision resistors). There's an integrator U_1, a "logic engine" that keeps track of the integrator output (via comparator U_2) at each 375 kHz clock tick, and a pair of switches (S_1 and S_2) that are operated synchronously by the logic to keep the integrator approximately balanced (by sinking or sourcing an accurate current into the integrator). There's also an ADC of only modest precision (12 bits) that is used to read the integrator's output voltage at the beginning and end of a multicycle measurement.

Here's how it works, at a basic level: to begin a measurement, switch S_3 is closed,[53] causing the integrator to ramp up or down (according to $dV/dt = -I_{in}/C_1 = -V_{in}/R_{in}C_1$). At each successive clock the logic engine causes either switch S_1 or S_2 to close (depending on the polarity of the integrator's output), thus adding or subtracting the corresponding reference current (±10 V/30 kΩ) to force the integrator to come back toward ground. This goes on for many clock cycles (for maximum rejection of powerline pickup it's desirable to use a measurement time corresponding to an integral number of powerline cycles; e.g., 6250 clock ticks equals 1/60 of a second), after which the logic tallies up the number of positive (n_+) and negative (n_-) cycles. This gives a *first-order* estimate of the average input voltage during the measurement time:

$$V_{sig}(1) \approx V_{ref}\,\frac{n_- - n_+}{N_{cycles}}\,\frac{R_{in}}{R_1} = V_{ref}\,\frac{n_- - n_+}{6250}\,\frac{100k}{30k}.$$

This is not terribly accurate: a full-scale input of ±12 V produces a net count ($n_- - n_+$) of ±2250 (figure out why), so the resolution is approximately 12 bits. Now for a nice trick: because the measurement is timed by an integral number of clock cycles (rather than by a zero crossing, as in the dual-slope method), the residual integrator level contains additional information. It allows us effectively to perform a vernier-like subdivision of a clock cycle. That's the reason for the ADC in Figure 13.47, which is used to measure the integrator voltage at the beginning and end of the measurement cycle. For the 12-bit ADC in the figure, this provides some 512 levels of subdivision of the first-order LSB, adding 9 bits to the ∼12-bit first-order estimate, for a final result of ∼21 bit resolution.[54]

More precisely, the second-order (and final!) answer is given by

$$V_{sig}(2) = V_{sig}(1) + \frac{R_{in}C_1}{T_{meas}}(V_f - V_i) = V_{sig}(1) + 0.00264(V_f - V_i),$$

[53] We use the term "closed" to mean that a given switch is connected to the summing junction.

[54] The ADC's conversion range matches that of the integrator, but both are ∼8× larger than the integrator's ramp over a clock tick when the input signal is at ±full scale. That is why the ADC effectively loses 3 bits of resolution when digitizing the residue ($V_f - V_i$).

where the coefficient of the ΔV "vernier" term represents the decreasing contribution of the endpoint correction with increasing measurement time. More specifically, it's easy to show that the ΔV term by itself yields the correct input voltage for a measurement whose duration equals a single clock cycle (i.e., $T_{\text{meas}} = 1/f_{\text{clk}}$).

Exercise 13.5. Take the challenge: show that this is correct.

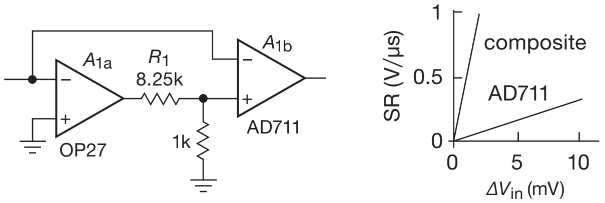


Figure 13.48. A "composite amplifier" greatly increases slew rate. The interstage attenuator's 8.25k (nominal) value is chosen large enough to ensure U_1's stability

B. Details, details...

That's the top-level view of the multislope conversion technique. There is much more to say – there's the usual devil in the details; and there are many possible refinements, to squeeze the most performance from this core idea. Here we restrain ourselves, restricting our commentary to a terse summary of the more interesting and instructive aspects.

Noncritical components For S_1–S_3, Agilent uses a standard 74HC4053 switch array (from NXP), and for C_1 a commodity ceramic chip capacitor of the stable NP0/C0G variety (from AVX). This type of capacitor, which is stunningly inexpensive,[55] exhibits a low temperature coefficient (±30 ppm/°C) and negligible dielectric absorption ("memory," see Figure 5.4 in §5.6.2 and the more extensive figures in §*1x.3*), unmeasurable on the switching time scales here. Likewise, no great accuracy or stability is needed for the comparator or endpoint-quantizing ADC.

Critical components The voltage references set the scale of the measurement, and must be highly stable. In practice these instruments use a single 7.0 V zener-type reference and a pair of precision op-amps to produce the ±10.0 V reference voltages.[56] The "10.0 V" voltage does not need to be precise to the ultimate accuracy of the instrument, which undergoes a factory calibration; it does need to be *stable*, of course, to maintain that calibrated accuracy.[57]

Two other critical components are the matched resistor arrays (R_1–R_3, and the gain-setting resistors in the voltage references) and the integrator op-amp. The latter is in fact a composite amplifier (an OP27+AD711), achieving high slew rate and high loop gain along with very low offset voltage (Figure 13.48). The resistors are specially packaged arrays for close matching and tracking. What matters here is the drift (over time and temperature) of the resistor ratios, because minor mismatches in the initial resistor ratios are handled in the factory calibration.

Not shown, but just as critical, is the selectable-gain input amplifier. It must have precise and stable gains, and means to carry out calibrations and corrections; see §5.12.

The switches The 74HC4053 switch is used in a current-steering scheme like that in Figure 13.45, arranged so that the voltages on all of the '4053 switch pins remain always close to zero volts. The switches serve only to steer the currents, adding or subtracting a slew rate to the integrator output that is in a precise ratio to the slew rate produced by an input voltage. The on-resistance of the switches does, of course, figure into the value of these currents. But as long as the R_{on} of the switches is well matched, stable, and small compared with R_1–R_3, the effect is corrected by the calibration cycle that the instrument performs automatically before each measurement (see next paragraph). For the NXP 'HC4053 used in the Keysight instruments, for example, R_{on} is typically 85 Ω, matched to 8 Ω. It's important that these switches operate as "break-before-make," so that their output terminal pairs are not momentarily shorted (which would connect the integrator summing junction to ground, acting like a differential input signal equal to the op-amp's offset voltage). Some manufacturers include such a specification, others don't. For example, the same NXP datasheet specifies turn-ON and turn-OFF times, whose difference represents a break-before-make interval of 4 ns; but it does not specify that interval directly, whereas the DG4053 datasheet from Siliconix lists a "Break-Before-Make Time Delay" t_{D} of 6 ns typ (2 ns min).

Calibration The simple current-switched input topology is well suited to calibrating, and eliminating, the effects of resistor ratio mismatches, voltage reference

[55] $0.06 in 100-piece quantities, some 50,000 in stock at DigiKey this morning.

[56] We are told by reliable sources that Keysight uses the spectacular LTZ1000, see §9.10.1B.

[57] For the 34401A standard bench DMM, for example, the initial factory-calibrated dc accuracy is within ∼2 ppm; it is specified to drift no more than ±0.0015% over 24 hours, but ±0.0035% after one year.

mismatches, op-amp offsets, switch delays, and the like. For example, when S_3 is turned off (i.e., no input signal) and S_1 and S_2 are alternated for some number of successive cycles, the endpoint ΔV is a measure of the mismatch of positive and negative reference currents. Likewise, by routing V_{ref} to the signal input and performing a voltage measurement, you get a measure of the mismatch of signal and reference currents. The Keysight instruments perform a suite of such calibrations before each measurement on its higher-resolution settings (where it makes a difference). Of course, there's no way it can determine the drift of its primary voltage reference; for that you need an external source of known voltage. This is the business model of calibration laboratories.

Measurement interval We used the example of a measurement time T_{meas} equal to the period of one powerline cycle (PLC), in this case 6250 cycles of 375 kHz, or 1/60 of a second. A measurement time that is an integral number of PLCs ("NPLCs") powerfully rejects coupled interference, and the longer measurement times improve the ultimate accuracy; see Table 13.8. But, as the table shows, you can make more rapid measurements, at the expense of powerline rejection and of accuracy.[58] It's also possible to make *continuous* measurements, in which signal switch S_3 is always on (in this mode the endpoint ADC must take accurately timed samples).

"*Multi*multislope" Circling back to the original game-changing HP3458A 8.5-digit DMM, it uses an amusing bit of trickery: it has four sets of input resistors and switches, such that it can reduce the integrator slew rate dramatically (by a factor of ~600) as it nears the endpoint.[59] Oddly, it does not measure the endpoint residue; it ramps down to 0 V instead, missing out on a very powerful trick.

Miscellany There's a lot of detail in the final implementation of this conversion technique, as described in the service manuals and *HP Journal* articles, as well as the relevant patents.[60] For example, it's necessary only to prevent integrator saturation; so you can use more than one comparator, and turn on the reference current switches (S_1 and S_2) only as needed to keep the integrator in range. This minimizes the number of switching cycles and the errors that go with it. There are also some curious circuit quirks; for example, there are lossy ferrite beads on the '4053 analog outputs, and there's a capacitor from the integrator summing junction to ground. Go figure. . . .

Evolution of the technique In 2006 Agilent introduced faster "Multislope IV" versions,[61] the 34410A, 34411A, and the 34972A, all with USB and Ethernet data links; they cost more, and the classic 34401A and 34970A were reduced in price. They subsequently introduced the 34460A and 34461A models, with additional features such as display-panel signal processing and sensor interfaces (the 34461A has the same measurement speeds and capabilities as the 34401A, listed in Table 13.8). The 34420A remained the only 7.5-digit (20 ppm) meter in the lineup, and was not upgraded.

Table 13.8 Keysight's Multislope-III ADCs[a]

measurement duration			digits[d]	rdgs per sec[c]	50/60Hz reject[e]
(PLC[b])	(ms[c])	(clocks[c])			(dB)
0.024	0.4	150	4.5	1000	–
0.2	3	1500	5.5	300	–
1	16.7	6250	5.5	60	60
10	167	62.5k	6.5	6	95
100	1.67s	625k	6.5[f]	0.6	105
200[g]	3.33s	1.25M	7.5	0.3	110

Notes: (a) 34401A DMM, 34420A MicroVolt, 34970A DAQ. (b) powerline cycles; selectable 50/60Hz via setup menu. (c) when set for 60Hz; clock rate is 375kHz. (d) reported. (e) normal-mode rejection, at selected powerline freq. (f) 7.5 digits for the 34420A. (g) 34420A only.

C. From multislope to delta–sigma

The multislope converter takes us naturally to the wildly popular delta–sigma conversion technique, with which it has much in common. At the most basic level, both are

[58] Improving on the basic multislope technique, the recent Keysight 34411A bench DMMs do better: 1000 readings/s at 6.5-digit resolution, and 50,000 readings/s at 4.5-digit resolution.

[59] This suite of slopes may be the origin of the "multi" in multislope. They created the name "Multislope III" for the subsequent (and simpler) scheme outlined earlier.

[60] *HP Journal*, April 1989; US patents 4,357,600 and 4,559,521.

[61] Looking at Multislope IV waveforms with a scope, you see a much different beast. A_{1b} has been replaced with a faster AD829 op-amp (120 MHz, 230 V/μs), with the integrating capacitor C_1 reduced by 5×. A hardware engine forces the error integrator to produce consistent 10 Vpp ramps with a 2 μs period, using coarse data from an 80 MHz AD9283 converter digitizing the 10 V ramp at 14 ns intervals, and fine data from an AD9200 10-bit converter with a limited 2 V range, clocked at 75 ns intervals near zero volts. Slope changes and the counter record are made at 75 ns intervals, and the AD9200 makes starting and ending readouts with 0.02% resolution. As a result, a Multislope IV converter can measure 4.5 digits in 20 μs (0.001 PLC), or 20× faster than Multislope III.

integrating methods in which a discrete offset is applied to the input at periodic intervals, based on the integrator's output level. As we'll see, however, the delta–sigma technique has several subtle tricks up its sleeve, enabling it to deliver astonishing performance.

13.9 ADCs IV: delta–sigma

Now, finally, an extended section on what has become a favorite A/D (and sometimes D/A) conversion technique: the "delta–sigma" converter. These things are confusing, but they're worth some serious effort to understand, because they deliver top-notch performance in resolution and accuracy (e.g., monotonic to 31 bits or more) from "voltmeter" through audio speeds and beyond. Their "oversampling" architecture greatly simplifies the input anti-alias lowpass filter, and performs some magic in shifting the noise spectrum out of the passband. And they provide this performance at surprisingly low cost. In the coming subsections we introduce the basic idea; then we take a serious look at how these converters deliver performance far better than would appear possible. We conclude with some application examples.

13.9.1 A simple delta–sigma for our suntan monitor

To get started, let's revisit the saga of our suntan monitor (see §§4.8.4; to be revisited yet once again, and finally set to rest, in §15.2), this time implemented with the simplest delta–sigma digital integrator. Figure 13.49 shows the implementation, using a clocked charge-dispensing integrator that operates in the same way as the multislope integrator. In fact, it is simpler, because it does not bother with the fractional-cycle endpoint correction: it simply accumulates the integrated sunlight dose, by counting the number of clock cycles during which it is obliged to inject a reference current (here V_{cc}/R) to balance the outgoing photodiode current I_{PD}. Not shown here is the circuitry to sound an alarm when the preset count is reached: the reader who has come this far knows well how to do that!

This is the simplest delta–sigma integrator: it accumulates (sigma) the difference (delta) between the analog input and the measured current that is combined at the summing junction. It *could* become a complete analog-to-digital *converter* (rather than a mere *integrator*), if some circuitry were added to (a) clear the counter to begin a conversion, and (b) read the counter's value after a fixed time interval that is much longer than the clock period. Indeed, that scheme would work as an ADC. But in practice you get far better performance by replacing the counter with a digital filter; and you improve things still further by cascading several stages of difference amplifier plus integrator.

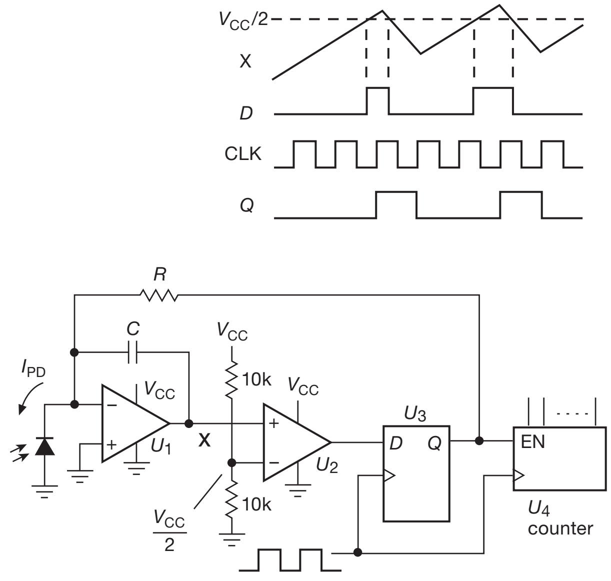


Figure 13.49. Discrete delta–sigma photocurrent-integrating suntan monitor.

We'll get to all of that, soon enough. First, though, let's take the time to understand this simplest example.

In this circuit U_1 is a single-supply op-amp that operates with inputs to (and slightly beyond) the negative rail, and U_2 is a comparator with active pullup. For a low-speed application like this you could use a dual RRIO op-amp like our favorite LMC6482, running on the same +3.3 V or +5 V supply as the digital logic. The integrator ramps up (with slope proportional to the photodiode current I_{PD}) until the next clock rising edge at which its output is greater than $V_{CC}/2$, at which point it ramps down with slope proportional to the net current at the summing junction, $V_{CC}/R - I_{PD}$. The result is that the duty cycle D (fraction of time that the Q output of U_3 is HIGH), averaged over many cycles, is $D{=}I_{PD}R/V_{CC}$, thus $I_{PD}{=}DV_{CC}/R$. The duty cycle D is gotten from the count N in U_4 during time interval T by $D{=}N/f_{clk}T$; note that this result does not depend on the comparison voltage $V_{CC}/2$ (or the voltage at point X).

The design goes like this.

(a) Choose a clock period that is much shorter than the expected baking time, for example $f_{clk}{=}10\,\text{Hz}$; faster is OK, but then you need a larger counter.

(b) Choose R to source more current than the anticipated full-scale input current; for $I_{FS}{=}1\,\mu\text{A}$ and $V_{CC}{=}5\,\text{V}$, R must be less than $5\,\text{M}\Omega$.

(c) Choose C to keep the maximum integrator excursion safely less than $V_{CC}/2$ during one clock cycle.

Here we might choose f_{clk}=10 Hz, R=3.3 MΩ, and C=100 nF. The integrator ramps through at most 1.5 V per clock period (at minimum I_{PD}), so it cannot saturate. The peak count rate equals the clock frequency (and the average count rate is somewhat less, in this case $0.6f_{clk}$), so a 16-bit counter is conservatively adequate for bake settings up to 2 hours full-sunlight equivalent.

Some important points.

(a) The overall calibration depends on the supply voltage V_{CC}, which we've assumed is a stable +5 V; and we've taken advantage of CMOS logic's clean saturation to the rails.

(b) Note that the integrator waveform is not accurately periodic; its excursions above and below the threshold at $V_{CC}/2$ wander somewhat, the guarantee being only that it will be turned around at the next clock following a threshold crossing. This does not, however, degrade its overall accuracy, averaged over many cycles: the integrating nature of the delta–sigma system properly keeps track of the deficits and surpluses; the integrator gets credit for its extra mileage.

(c) The dynamic range of the converter is limited by the op-amp's offset voltage, which causes an equivalent input-current error of V_{os}/R; for this design that is about 0.2 nA (worst case) for the -A grade, thus a dynamic range of 5×10^3. The op-amp's bias current is negligible by comparison (4 pA, max, over temperature).

(d) The dynamic range would be greatly extended if R were replaced with a switched current source, assuming of course that the input signal remains in the form of a current.

(e) In this circuit the comparator U_2 need not be accurate; in fact, it could be omitted altogether, with the flip-flop's logic threshold taking its place. Similarly, the operation does not require an accurate comparison threshold voltage; we chose $V_{CC}/2$ for convenience.

We'll see some additional examples of delta–sigma conversion in §13.9.11. The impatient reader may wish to skip over the following subsections, in which we explore more deeply the operation and performance of the often-confusing delta–sigma technique.

13.9.2 Demystifying the delta–sigma converter

As we noted, the delta–sigma integrator becomes a *converter* of the average analog input voltage, if you capture the count accumulated over a fixed measurement time T_{meas}. The measurement time must be much longer than the clock period, of course, to achieve decent resolution, because the maximum count is just T_{meas}/T_{clk}. So, for example, if you were to design an ADC to convert at 100 ksps, you might use a 10 MHz clock, cleared at the beginning of each conversion, and read out 10 μs later. The full-scale count would be 100, which you could describe as a (nearly) 7-bit conversion. To achieve 16 bits, you'd need to run the clock at $2^{16}\times100$ kHz, or 6.5536 GHz!

This does not sound promising. It just seems like a bad idea to design a "1-bit" converter – which is what you have in the stream of bits that is driving the counter in this design. So it will come as a surprise that the impossible *can* be done: there are plenty of 16-bit delta–sigma ADCs that convert at audio rates (e.g., 96 ksps), and in fact there are some that achieve 20 bits or more of resolution at that speed (see §13.10.1, and Tables 13.9 and 13.10). How can this be?! Read on. . . .

Sometime in the 1990s the promotional materials for consumer-grade CD audio players began to trumpet their use of "1-bit digital-to-analog converters," as if there was something good about *reducing* the resolution from the previously trumpeted 16 bits.[62] This seemed puzzling to many of us; but we didn't complain because, well, the players sounded pretty good.

As Bob Pease would say, what's all this 1-bit converter stuff, anyhow?

13.9.3 ΔΣ ADC and DAC

As we'll see, ΔΣ (also known as ΣΔ) conversion can go either way – D/A, or A/D. In contemporary practice ΔΣ *DAC*s are used primarily for audio applications, where they excel in linearity, monotonicity, and low cost. A typical audio ΔΣ DAC might integrate six 24-bit 192 ksps converters, with 114 dB effective dynamic range, for about $10.[63] By contrast, ΔΣ *ADC*s cover a broad range of applications, ranging from precision (24-bit) dc-accurate slow converters, to audio-rate converters of high resolution (e.g., 24-bit, 96 ksps), and up to speedy ADCs with more accuracy than you would imagine (e.g., 16-bit, 20 Msps).

[62] Trumpet blasts are still echoing. Here's a contemporary example: "There is virtually no noise or sound degradation during the signal transmission and amplification process as 1-bit signals are digital." As if n-bit signals are *not* digital?!

[63] For audio applications the dc performance is irrelevant, and in general is not even specified. An exception is the excellent DAC1220 20-bit ΔΣ DAC from Texas Instruments.

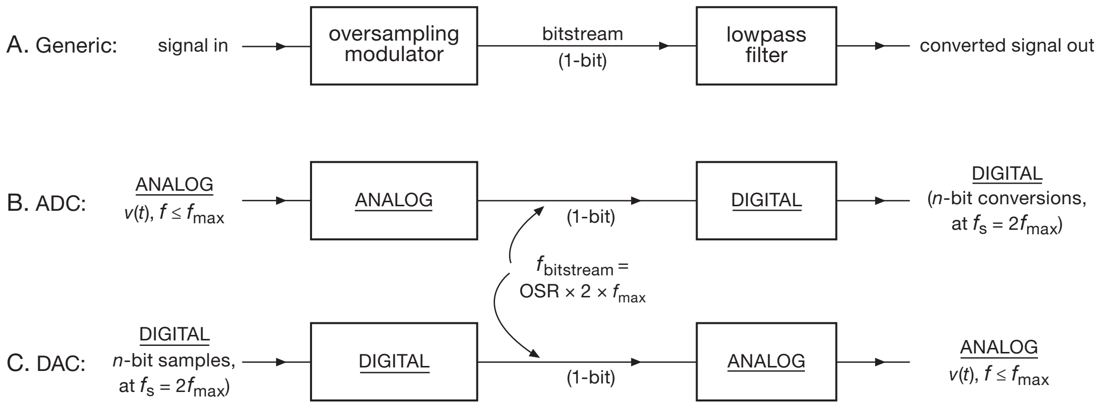


Figure 13.50. A delta–sigma converter, whether A/D or D/A, consists of two parts: an oversampling *modulator* that produces an intermediate *bitstream*, followed by a lowpass filter that recovers the converted output.

In the discussion that follows, we talk primarily about ΔΣ ADCs, both because of their importance and also because their architecture exploits the ideal characteristics of digital filtering.

Along the way, we will try to get to the bottom of what, to us, had seemed like a *big mystery*, namely: *How can it be that 1-bit conversions, at some modest "oversampling" rate (say 64 times the usual Nyquist rate of* $2f_{\max}$*), can produce digitized output samples of great accuracy (say 16 bits)?* To rephrase the mystery: naively one might expect that 1-bit conversions at a 64-fold oversampling rate might allow us to recover a final digital output with 6-bit resolution (because 2^6=64), but no better.[64] We'll see, though, that it's possible (and mandatory, for audio applications!) to do considerably better.[65]

13.9.4 The ΔΣ process

Figure 13.50 shows the basic ΔΣ process. An incoming signal, bandwidth limited to some maximum frequency $f_{\max}$ (usually by an anti-alias filter[66]), is converted to a bitstream[67] by a *modulator*. The latter is clocked at some multiple of the minimum Nyquist sampling rate $2f_{\max}$, generating an output bitstream of rate f_{bit}=OSR$\times 2f_{\max}$, where OSR is called the *oversampling ratio*. This bitstream is the intermediate step in the overall converter: to get the converted output, the bitstream must be lowpass filtered.

Note that both the modulator and the lowpass filter may each be analog or digital, depending on the type of converter: a delta–sigma ADC consists of an analog modulator followed by a digital filter, whereas a delta–sigma DAC consists of a digital modulator followed by an analog filter.[68] In what follows we will deal primarily with the modulator portion of the overall converter.

A. The modulator

In either case, the lowpass filter is "just a filter," which simply bandwidth limits the incoming bitstream.[69] The

[64] That is, in fact, the case with filtered PWM, as used, for example, for motor control or LED dimming: there one might divide the Nyquist period into 64 time slots, setting the first few as 1s and the rest 0s. Delta–sigma is more subtle, and better, with 0s and 1s sprinkled through the Nyquist period in such a way as to produce a filtered output of high accuracy.

[65] Plot spoiler, for the impatient: thinking in the time domain, it's a willing conspiracy between an output lowpass filter that views a long stretch of the bitstream, and a very clever bitstream-building modulator whose filtered output represents an accurate conversion. A better (and quantitative) understanding comes in the frequency domain, where the oversampling modulator acts to reduce in-band quantization noise, "shaping" it to higher (out-of-band) frequencies.

[66] Which, as we'll see later, need not cut off sharply, thanks to the beneficial effects of oversampling; see Figure 13.60.

[67] Here shown as 1 bit wide, for simplicity, though in practice it may be several bits wide (i.e., more than 2 levels).

[68] An interesting third possibility (analog/analog) is exemplified by the Avago HCPL-7800A analog opto-isolator: an internal input modulator creates a bitstream that is optically coupled to an internal analog output demodulator, to deliver an accurate analog replica (0.004% nonlinearity, typical) of high stability (3 ppm/°C gain change, typ), kilovolt-level isolation, and 100 kHz bandwidth. Another example of delta–sigma "A-to-A" is the Super Audio CD (SACD), a CD-like audio storage format in which the 2.8 Mbps intermediate (encrypted) bitstream itself is recorded and distributed to the user, with lowpass filtering applied at playback.

[69] The bitstream can be thought of either as *digital* (1s and 0s), which is then filtered by a digital filter (if this is an ADC), or as an *analog wave-*

interesting action (and the *magic*) takes place in the modulator. Figure 13.51 shows a block diagram of a “first-order” oversampled modulator, which accepts analog input voltages between -1 V and +1 V, bandlimited to a maximum frequency of $f_{\rm max}$, and produces a 1-bit output bitstream at a rate OSR time higher than the critical Nyquist sampling rate $2f_{\rm max}$.

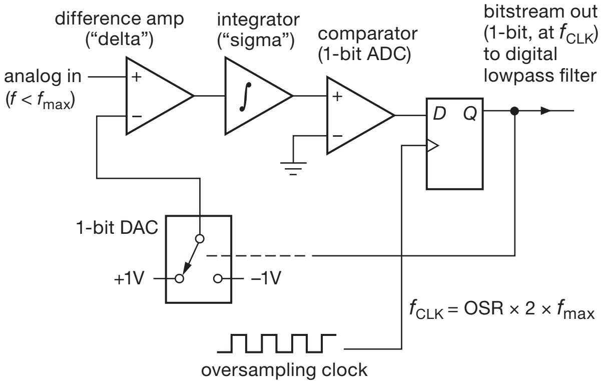


Figure 13.51. A first-order delta–sigma analog modulator.

At each clock cycle the current bitstream value, converted to an analog voltage (in this case ±1 V), is subtracted from the analog input, the difference signal being integrated (in a standard op-amp analog integrator, here assumed to be noninverting) and presented to a latched comparator. The integrator gain is such that a full-scale analog input to the integrator (i.e., +1 V) produces a full-scale (+1 V) change in the integrator’s output after one clock period. That is, you can think of the integrator as an “analog accumulator”: for a (fixed) input voltage V, its output voltage increases by V during one clock period.

The result is a rapid stream of 1s and 0s (at, say, 64 times the usual $2f_{\rm max}$ sampling rate), responding to the relatively slow (64 times slower, say) changes in the input signal. Thinking of these bits as ±1 V, the modulator produces a stream whose *average value* matches the input signal. We can understand this by thinking of the modulator circuit as a negative-feedback loop that strives to minimize the averaged (i.e., integrated) error between the input signal and the output stream (which has been converted back to analog by a “1-bit DAC”). Looking more closely, though, we can see that it is doing a *terrible job*: sample by sample, its output bitstream simply bounces between extremes. As Bob Adams has written,[70] pithily, “Oversampling converters achieve increased resolution not by decreasing the error between analog input and digital output, but by making the error occur more often.”

B. The ADC’s dynamic range (resolution)

The output lowpass digital filter (typically an FIR digital filter – see Figure 13.52 – in this case a 1-bit shift register with the marching 1- and 0-bit values turning on or off a set of fixed digital coefficients[71] that are added digitally to create the multibit output samples) creates the digital n-bit numbers that are the converter’s output. Because they emerge from the filter at the full oversampling rate, they are subjected to a “decimation” operation, most simply by discarding superfluous outputs and outputting only one converted output for every OSR clock cycles.[72]

Naively, then, we achieve increased resolution by having plenty of 1-bit samples to average over, for each half-cycle of the highest frequency in the incoming waveform. Because the average value of the bitstream tracks the input signal, we understand Bob Adams’ statement, and all is well.

Or is it? Consider an example: suppose we are digitizing audio, with an $f_{\rm max}$ of 20 kHz. A conventional ADC (say a successive approximation converter) might sample at 48 ksps, comfortably above the critical minimum of 40 kHz. Imagine instead that we rig up a ΔΣ ADC, with a typical oversampling ratio of 64; that is, we run the modulator at 3.072 Msps (64×48 kHz), creating the (1-bit) bitstream at that rate. Now we filter that bitstream, for example by taking a running average,[73] digitally, that captures 64 successive bits at a time. What does the output look like? Well, when you take the average value of 64 bits, there are only 64 possible values. So we’ve invented a paltry 6-bit ADC.

Following this logic, we’d need to oversample by 2^{16} (i.e., 64K) to achieve 16-bit conversion. That would require a sampling rate of roughly 3 *giga*hertz! Delta–sigma conversion is not looking good.

form that switches between two fixed voltage levels (if this is a DAC). Note also that the phrase “just a filter” does not mean that the filter design is simple, or trivial. In particular, *digital* filter design is a sophisticated art, with issues of window functions, nulls in the response, and so on. See §6.3.7.

[70] “Design and implementation of an audio 18-bit analog-to-digital converter using oversampling techniques,” *J. Audio Eng. Soc.* **34**, 153–166 (1986).

[71] To a first approximation the time-series coefficients are a (signed) sinc function, the Fourier transform of a “brickwall” lowpass function. See §6.3.7, on digital filters.

[72] In practical implementations the filtering and decimation are combined, using a “multirate decimation filter.”

[73] Rather than the more complicated sinc-function weighted average that’s required for implementing an ideal “brick-wall” lowpass; see §6.3.7.

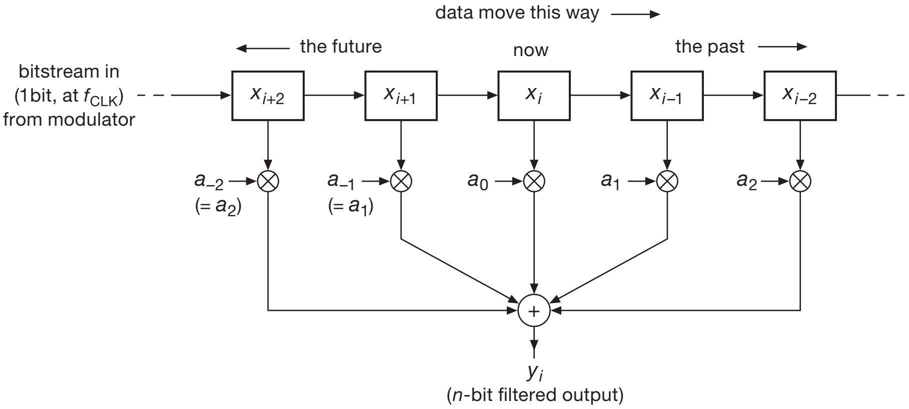


Figure 13.52. A digital filter uses digital memory and arithmetic elements to generate a digital-output sequence that represents a filtered version of a digital input sequence (see §6.3.7). Here a shift register, digital multipliers, and an adder form a symmetric nonrecursive (finite impulse response, FIR) filter, suitable as the output lowpass filter in a 1-bit delta–sigma ADC.

C. So, what's going on? (time-domain intuition)

The answer to this paradox can be stated in different ways. In the literature the usual approach is to say that the operation of the 1-bit digitizer consists of a perfect conversion, but with added broadband "noise" (consisting of the difference between the actual analog waveform and the 1-bit quantized samples). This "injected quantization noise" has a broad spectrum (because of the oversampled clocking frequency), extending to the clock frequency and beyond. Of greater importance, the resulting "output quantization noise" (what remains at the output) is lowest at low frequencies, being "shaped" by the modulation process such that most of the output quantization noise is well above $f_{\max}$. Because of this so-called "noise shaping," the final lowpass filter acts to selectively eliminate most of the quantization noise from the bitstream, while preserving the converted signal. Voilà: resolution and dynamic range far better than our naive estimate above.

This is all quite correct, but, to us, unsatisfying (though we'll take a brief look at that approach in §13.9.5). We wanted to understand the secret of ADC dynamic range *in the time domain*, without recourse to the frequency domain. We struggled with this, reading exposés with titles like "Demystifying Sigma–Delta ADCs" and "Delta–Sigma ADCs in a Nutshell." Not much help – they all get to the critical step, then punt ("...sigma–delta converters overcome this limitation with the technique of noise shaping..." and "...you can see how the modulator shapes the noise to higher frequencies, facilitating the production of a higher resolution result," excerpts from those two articles, respectively).

Here's how to think of this in the time domain:[74] First, the lowpass filter does not simply take a running ("boxcar") average of the bitstream. Rather, it weights the individual 1-bit samples with carefully tailored coefficients to produce a better lowpass filter characteristic (see §6.3.7D). Because the individual bits are weighted differently, there are many more than 64 possible results (taking the example above). Furthermore, a typical FIR digital filter will weight and sum many more bits, and over a spread of time (samples) much longer than the oversampling rate (i.e., extending beyond what we might call a single "Nyquist interval," by which we mean the time interval equal to a half-period of $f_{\max}$). For a 64× oversampling ADC, the digital lowpass filter might use of order a thousand "taps" (samples along the bitstream), each with its multiplying coefficient, and spanning perhaps ten to twenty Nyquist intervals, to generate each final (decimated) output number. So it is at least *plausible* that you could achieve resolution far greater than possible with simple averaging.

Continuing in this vein, it's worth noting that each bit in the bitstream contributes to many final n-bit output numbers (after decimation). So, in a conspiratorial vein, it's plausible that a cleverly contrived modulator could generate a bitstream that, after lowpass filtering, could produce an n-bit digitized output with quite considerably improved dynamic range. Thinking this way, we would not be wrong[75] in concluding that "the magic is in the modulator."

[74] We are indebted to Bob Adams of Analog Devices for helpful discussions, and the extermination of mental cobwebs. He is not responsible, however, for any errors (egregious or otherwise) here committed.

[75] "You're not *wrong*, Walter; you're just ...."

So the question becomes "How does such a simple device (Figure 13.51) behave so cleverly?"

13.9.5 An aside: "noise shaping"

As we remarked, the usual description of delta–sigma converters talks about *noise shaping* in the frequency domain: the flat-spectrum "quantization noise" introduced in the quantizer (the comparator of Figure 13.51) gets "pushed up" to high frequencies, mostly above the output sampling rate. And less in-band noise equates to higher accuracy – end of argument.

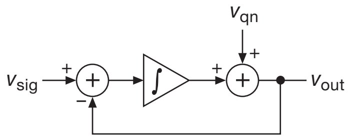


Figure 13.53. Noise shaping in a first-order delta–sigma ADC: an all-analog model, with quantizer replaced by an additive quantization noise source.

To many engineers this is a satisfactory explanation. But even if you aren't particularly convinced by this argument, it's worth understanding it. To see most simply how this works, look at Figure 13.53, in which the modulator is constructed with an *analog* integrator, and converts the (continuous) analog input to a 2-state analog (±1 V) output. In this equivalent analog model we've replaced the 1-bit quantizer (comparator) with an additive quantization noise voltage v_{qn}, whose flat spectrum extends to the oversampling clock frequency (and beyond).[76] You can think of the integrator as being in the forward path of the loop for the signal input (thus lowpass), but in the feedback path for the noise input (thus highpass).[77] From this simple analog loop we can easily figure the frequency responses to both the input signal and to the quantization noise signal. There's only one gain parameter, namely that of the integrator, whose gain we write as $\mathbf{G} = \omega_0/j\omega$; that is, the integrator's gain (proportional to $1/\omega$) is such that its magnitude is unity at ω_0. From our earlier discussion, $\omega_0 \approx 1/2\pi f_{\rm overclock}$, the frequency of the converter's oversampling input clock.

Let's figure the gains, for which we properly have to do the full complex-number business. To get the input signal gain $\mathbf{G}_{\rm sig}$ we set v_{qn}=0; then

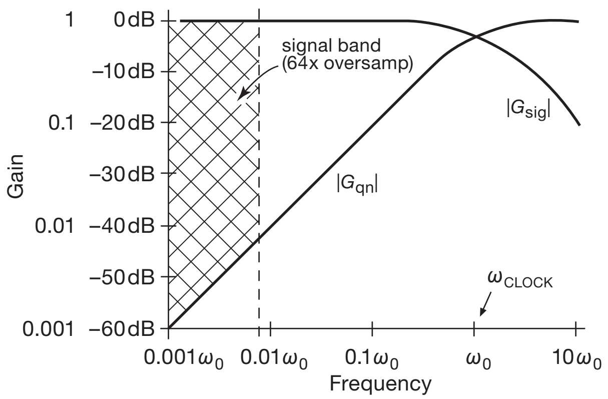


Figure 13.54. Signal gain and quantization noise gain versus frequency for a first-order delta–sigma ADC. The clock frequency $\omega_{\rm clk} = \omega_0 = 2 \cdot {\rm OSR} \cdot \omega_{\rm max}$ here equals 128 times $\omega_{\rm max}$.

$$v_{\rm out} = \frac{\omega_0}{j\omega}(v_{\rm sig} - v_{\rm out}),$$

$$\text{so} \quad \frac{v_{\rm out}}{v_{\rm sig}} = \frac{\omega_0/j\omega}{1+\omega_0/j\omega},$$

$$\text{and} \quad |\mathbf{G}_{\rm sig}| \equiv \left|\frac{v_{\rm out}}{v_{\rm sig}}\right| = \frac{1}{\sqrt{1+(\omega/\omega_0)^2}}.$$

That's a lowpass filter, with breakpoint at $\omega = 2\pi f = \omega_0$ (Figure 13.54).[78]

Similarly, for the quantization noise gain $\mathbf{G}_{\rm qn}$ we set $v_{\rm sig}$=0; then

$$v_{\rm out} = v_{\rm qn} - \frac{\omega_0}{j\omega} v_{\rm out}$$

$$\text{so} \quad \frac{v_{\rm out}}{v_{\rm qn}} = \frac{1}{1+\omega_0/j\omega}$$

$$\text{and} \quad |\mathbf{G}_{\rm qn}| \equiv \left|\frac{v_{\rm out}}{v_{\rm qn}}\right| = \frac{\omega/\omega_0}{\sqrt{1+(\omega/\omega_0)^2}}.$$

And that's a highpass filter, with the same breakpoint.[79]

So, in this lowest-order delta–sigma ADC the quantization noise is attenuated at low frequencies, its spectrum

[76] You usually think of additive "noise" as being small compared with the signal that it is guilty of degrading. Here the quantization noise (the difference between the analog signal and the 2-level output voltage $v_{\rm out}$) is actually larger than the signal itself.

[77] Explained nicely by Ewe Beis, on his website at `http://www.beis.de/Elektronik/Electronics.html`.

[78] A bit of intuition may be helpful here: at *low* frequencies (well below ω_0) the integrator has lots of gain, so the loop is closed with plenty of loop gain, creating a unity-gain output (in spite of the integrator within). In fact, the combination of input adder and integrator is not unlike a standard op-amp with its $1/f$ (compensation) rolloff. But at high frequencies there's no loop gain, so you get the $\sim 1/f$ integrator rolloff.

[79] Intuition, again: this time the "signal" (i.e., the quantization noise $v_{\rm qn}$) acts like an additive output disturbance to the closed-loop unity-gain amplifier. So it is *removed* by the loop gain, which is high at low frequencies (its magnitude is ω_0/ω), but ineffective above ω_0.

sloping linearly up to the oversampling clock frequency. But this is an *over*clocked ADC, so the input signal frequency range of interest sits down at the low end of that spectrum (by the ratio of oversampling). In other words, the quantization noise is mostly out of the signal band. And, for higher-order modulators, the effect is more pronounced: the noise curve is quadratic for a second-order modulator, cubic for third-order, and so on. Thus the conclusion you usually see: a delta–sigma converter achieves its accuracy by "shaping the noise to higher frequencies."[80] You may not be particularly impressed by this little bit of arithmetic. But, hey, at least you've seen it done, simply and explicitly.

13.9.6 The bottom line

Both from plausibility arguments in the time domain, and from explicit calculation in the frequency domain, it appears that the modulator circuit holds the key to the sigma–delta ADC's performance; that is, its ability to quantize an analog input signal with a resolution considerably greater than the oversampling ratio. Furthermore, that figure of merit ($N_{\text{eff}}/\log_2\text{OSR}$, where N_{eff} is the effective number of bits in the quantized digital output) grows with modulator complexity: contemporary ADCs employ "higher-order" modulators, meaning that the single difference amplifier and integrator is replaced with several cascaded stages of difference amplifier plus integrator, each driven from the common bitstream (see Figure 13.55).[81] Higher-order modulators are widely used, because they extend the dynamic range without having to increase the oversampling ratio (see below); they also suppress to a large extent the *idle tones* (see §§13.9.9 and 13.9.10) that afflict first-order modulators.

Although our time-domain musings above may be helpful (if only to make plausible the excellent dynamic ranges that are claimed), any serious analysis must use the frequency-domain approach. The latter shows that a higher-order modulator (constructed with m integrators) modifies the noise shaping such that the in-band quantization noise (i.e., dc to f_{max}) is suppressed as $\text{OSR}^{m+0.5}$, where m is the modulator order (m=1 for Figure 13.51). Put another way, each doubling of the oversampling ratio suppresses quantization noise so as to increase the dynamic range by $m+\frac{1}{2}$ bits; or, to state it in terms of modulator order, the effective number of bits (ENOB) is the $\log_2$ of the oversampling ratio (e.g., 6 for OSR=64) multiplied by $m+\frac{1}{2}$ (thus, for example, ENOB≈15 for a second-order modulator with oversampling ratio of 64). The graph in Figure 13.56 shows the theoretical maximum dynamic range of a delta–sigma ADC as a function of oversampling ratio and modulator order.[82]

Another technique for extending dynamic range, speed or both, is to design a modulator that generates a modulated "wordstream" that is more than one bit wide. In Figure 13.51, for example, the 1-bit ADC, 1-bit DAC, and 1-bit register would be replaced with analogous 2-bit (4-level) components. There are lots of clever tricks to address imperfections in the multibit converters within the modulator (for example, cyclically exchanging bit positions to average out nonlinearities caused by offsets); they are well beyond the scope of this book.

13.9.7 A simulation

We wanted to see for ourselves just how the signals move through a delta–sigma ADC – particularly the bitstream produced by some random-looking analog input signal, and of course the resultant output numbers (plotted as discrete points alongside the analog input waveform). We also wanted to see what things look like in the frequency domain, where the noise shaping should be evident.

The simulation was coded[83] in Mathematica®, with the following recipe:

(a) a spectrally-flat pseudorandom waveform with Gaussian amplitude distribution was generated, evaluated at 8192 successive time steps;

(b) this waveform was filtered with an approximate ideal brick-wall lowpass, with cutoff at 1/8 of the maximum frequency; it was then normalized so that its amplitude was bounded by ±1, generating the "analog input

[80] Sometimes stated as "pushing the noise upward to higher frequencies." In our *linear* model here, nothing gets *pushed*; it just gets attenuated at the low-frequency end, and passes unattenuated at the high end. However, a fully accurate quantization noise model must properly take account of the fact that the 2-state bitstream has unit amplitude (always ±1), with the result that reducing the quantization noise power at the low-frequency end causes it to rise at the high end.

[81] A simple cascade of integrators can be used up to second order, but not beyond (because their accumulated phase shift produces instability); a weighted sum of cascaded integrator outputs is used instead, in a higher-order modulator. Contemporary audio $\Delta\Sigma$ ADCs typically use fifth-order modulators and 64× oversampling to achieve 20-bit effective dynamic range.

[82] Note, however, that practical modulators of order greater than 2 use a modified structure (weighted sum of integrator outputs), for which the formula is not strictly correct.

[83] By the ever-talented Jason Gallicchio, to whom we are indebted.

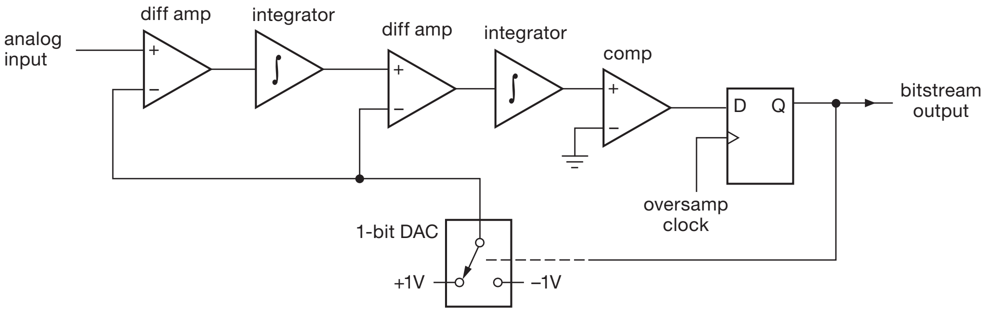


Figure 13.55. A second-order delta–sigma analog modulator. Lowpass filters can be substituted for one or more of the integrators.

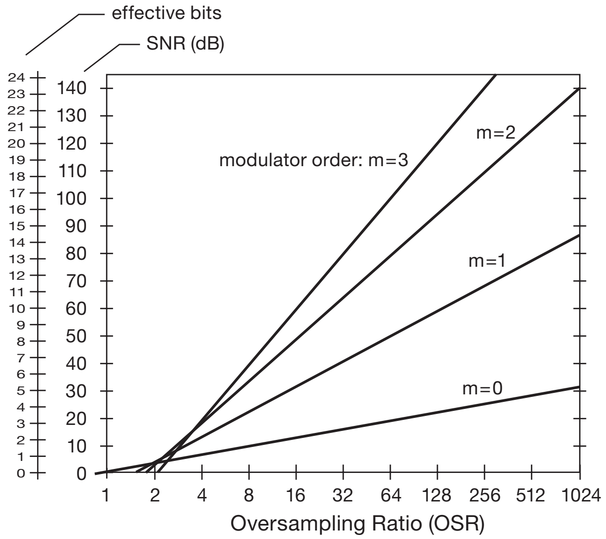


Figure 13.56. Dynamic range (SNR) and effective number of bits (ENOB), as functions of oversampling ratio (OSR) and modulator order (*m*), for a 1-bit oversampling ADC. (For a 2-bit modulator the slopes are doubled.)

signal"; the maximum frequency present in this signal was defined as the Nyquist frequency, f_{nyq};

(c) this signal is used to generate a bitstream with values of ±1, by numerically simulating a first-order oversampling delta–sigma modulator (in which the integrator is realized as a discrete digital accumulator); the oversampling ratio is 8×, and thus the clock frequency $f_{\text{clk}}=16f_{\text{nyq}}$ (recall that critical "1×" sampling requires $f_{\text{clk}}=2f_{\text{nyq}}$); finally,

(d) the bitstream, considered as an analog waveform itself, was lowpass filtered with the same filter function as in step (b), to produce the output samples; these emerge at the full 8× oversampled rate, and would normally be decimated (e.g., by preserving only every eighth point) to produce the ADC's digitized output at the "1×" rate (twice the highest frequency present in the input waveform).

Figure 13.57 plots a typical portion of the (longer) simulation, showing what's happening in the time domain. The tick marks on the time axis correspond to 1× (critical Nyquist) sampling, with individual dots plotted at the 8× oversampling rate. The input signal is the wiggly solid line, closely approximated by the discrete dots that are the digitized output numbers (plotted at their full 8× rate). You can see the bitstream waveform, on the same amplitude scale, as dots at ±1. Finally, the error (i.e., digitized output minus analog input, at each oversampling point) is the dotted waveform of small amplitude. From these plots you can do an eyeball estimate of the converter's accuracy; to us it appears to exhibit something like ±6% peak-to-peak amplitude error, which translates to an amplitude SNR of 16:1 (24 dB), in good agreement with the graph of Figure 13.56.

From this same simulation we plotted also the frequency spectra of the input signal, the output digitized "waveform," and the difference (an "error signal"); see

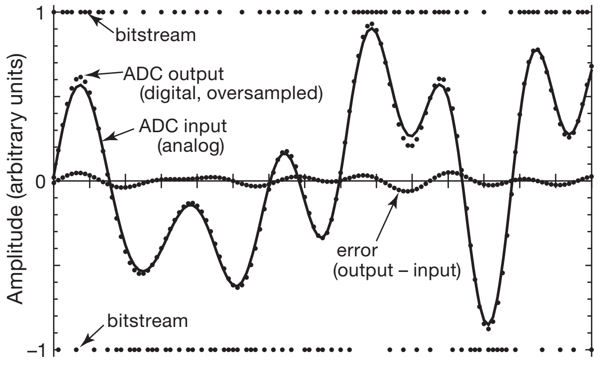


Figure 13.57. Numerical simulation of an 8× oversampling first-order delta–sigma DAC.

Figure 13.58. The spectra[84] extend to half the oversampling clock frequency, which corresponds to 8 times the highest input frequency. The top graph shows the flat input spectrum, cutting off sharply (nearly "brick wall") at unit frequency. The middle graph shows the spectrum of the raw bitstream itself, considered as a "waveform," where we would expect an approximate replica of the input, plus additional "noise" extending up to the oversampling clock frequency. In fact, we see just that – almost the same input spectrum up to unit frequency, with an additional quantization noise that increases roughly proportional to frequency. By taking the difference of these two spectra you can extract approximately the spectrum of the added noise introduced by quantizing (bottom graph), showing an approximately linear growth from zero up to at least half the oversampling clock frequency. The quantization noise above unit frequency is, of course, removed by the digital lowpass filter that accepts the bitstream as input and that completes the converter circuit (Figure 13.50).

The linear shape of the quantization noise spectrum, for this first-order modulator, would be replaced with a quadratic shape for a second-order modulator, and so on for higher orders. That higher-order noise shaping corresponds to improved accuracy (or SNR, or effective number of bits), as shown graphically in Figure 13.56.

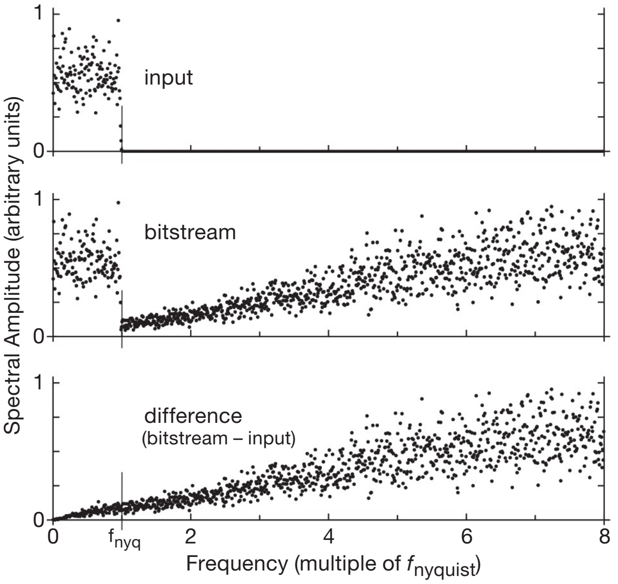


Figure 13.58. Frequency spectra of simulated 8× oversampling first-order delta–sigma ADC.

13.9.8 What about DACs?

As we indicated at the outset, the same scheme of lowpass filtering a bitstream produced by an upstream modulator is used also to make ΔΣ *digital-to-analog* converters. Look at Figure 13.59. Playing this role, the modulator accepts *n*-bit digital input samples representing the input signal. Comparing it with the modulator used for the ADC (Figure 13.51), the difference amplifier is replaced with a digital subtractor, and the integrator is replaced with a clocked digital accumulator. (At each clock the accumulator replaces its current latched value by the sum of that value and the input value.) The analog comparator is replaced with a digital comparator, most simply by just forwarding the sign bit (or the MSB, for unsigned offset binary) to create the 1-bit bitstream according to whether the accumulator's value is above or below the midpoint. And, finally, the 1-bit DAC is replaced with an "*n*-bit ADC" that simply generates one or the other of the full-scale *n*-bit quantities, in response to the 1-bit bitstream output. For 16-bit unsigned (offset) binary, for example, those values would be 0000h and FFFFh (all bits LOW or all bits HIGH).[85]

As with the analog modulator used in ΔΣ ADCs, the digital modulator for a DAC can be of higher order, with several stages of subtractor and accumulator (or digital lowpass filter); likewise, the digital modulator is not restricted to a 1-bit output stream. It could (and often does) generate a several-bit wordstream, in which case the several most significant bits form both the output wordstream and the digital feedback. Taking the example of a 2-bit (4-level) modulator, the 2-bit output stream would be both (a) converted to a 4-level analog voltage with a resistor ladder, then analog lowpass filtered to form the (analog) output signal, and (b) simultaneously mapped to one of four *n*-bit codes spanning the full input range (e.g., 0000h, 5555h, AAAAh, and FFFFh), then used as input to the *n*-bit digital subtractor at the input.

The output stage of a ΔΣ DAC is, as with the ADC, a lowpass filter. Here, however, it is an *analog* filter, which denies you the benefits of sophisticated digital filtering. The result is some compromise in filter characteristics, including clock feedthrough, and (with an analog, or "continuous-time" filter) sensitivity to clock timing jitter.[86]

[84] The spectral amplitudes were binned in groups of four to generate a sparser, and therefore more easily plotted, figure.

[85] For 2's complement 16-bit quantities, the corresponding values are 8000h and 7FFFh (corresponding to −32768 and +32767, the lowest and highest values).

[86] The lowpass filter can alternatively be implemented as a switched-capacitor (or "discrete-time") filter, which, sharing the same clock signal, effectively suppresses clock jitter.

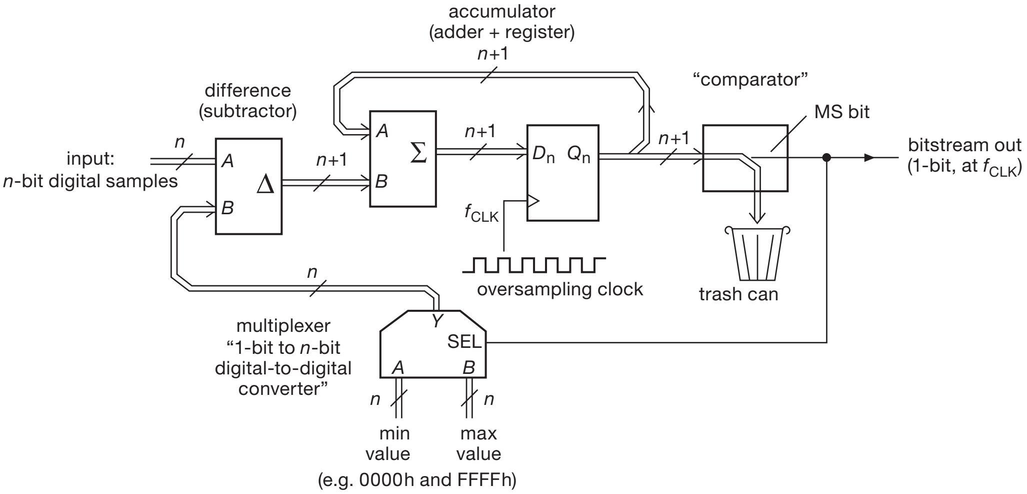


Figure 13.59. A first-order digital modulator in a delta–sigma DAC. The adder output is shifted one bit to prevent word growth.

13.9.9 Pros and Cons of ΔΣ oversampling converters

A. Advantages

Linear, monotonic, accurate Delta–sigma 1-bit converters are guaranteed monotonic; they are inherently linear, and can achieve 24-bit resolution at audio rates and below.

Inexpensive Delta–sigma ADCs use inexpensive (and accurate) *digital* lowpass filtering, and (because of oversampling) require only a low-order analog anti-alias filter at the input (see Figure 13.60)

B. Disadvantages

Limited bandwidth To ∼10–100 Msps at most (limited by gigahertz-scale oversampling clock).

Time delay The built-in ADC post-conversion digital filter achieves nearly ideal "brick wall" cutoff by using many taps, resulting in significant delay (or "latency," typically tens of output sample times, thus ∼millisecond for audio ADCs).[87]

DAC noise Delta–sigma DACs use an *analog* post-conversion lowpass filter, which permits some digital-switching feedthrough (by contrast, an *R*–2*R* DAC is completely "quiet").

Idle tones An ADC with first-order modulator can produce "idle tones" (see below) when presented with a static input that causes the integrator output to cycle with a sufficiently long period to be in-band (and causing apoplexy among audio aficionados); higher-order modulators suppress this artifact, even if present in the quantizer, owing to the higher in-band loop gain.

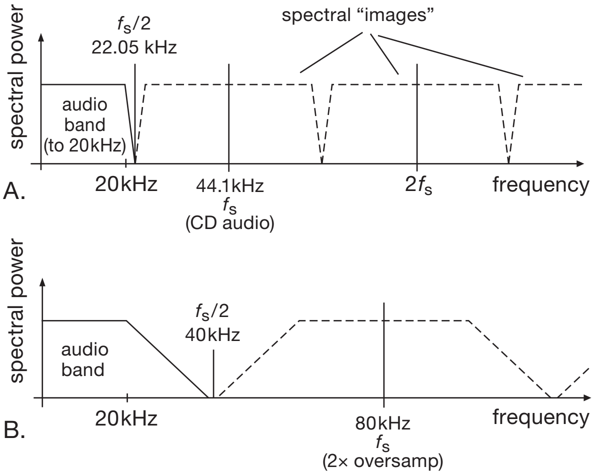


Figure 13.60. The spectrum of an analog signal that has been digitized ("sampled") periodically at a sampling frequency f_s includes mirrored copies ("images") centered at multiples of f_s. The analog signal must be lowpass filtered, before sampling, to eliminate components above $f_s/2$ (the "Nyquist frequency"); otherwise the mirrored bands create in-band "aliases" that cannot be subsequently removed. Oversampling relaxes the required steepness of such an "anti-alias LPF," as seen here for conventional CD-audio sampling (A, ≈10% oversampling) compared with 2× oversampling (B).

[87] However, note that the time delay is the same as would be produced by a sharp input anti-alias filter (of similar cutoff characteristics to that of the delta–sigma's post-conversion digital filter) followed by a conventional (zero-latency) ADC.

13.9.10 Idle tones

A distinguishing idiosyncracy of delta–sigma converters is the possibility of producing an "idle tone" – an unprovoked in-band periodic low-level (you hope) output signal. This undesirable characteristic afflicts particularly first-order modulators, which for this reason are never used for serious audio applications. You can get in-band idle tones (i.e., in the passband of the output lowpass filter) for particular values of static (dc) input; these are generally suppressed when there's signal activity at the input, hence the term "idle."

To see how this goes, consider the first-order modulator of Figure 13.51, which has a full-scale input range of ±1 V. If you apply a fixed dc input of value 0.625 V, and watch what happens with successive clock cycles, you get the states shown in Figure 13.61. The modulator (and therefore the output bitstream) goes periodically through a cycle of length 16 clocks. Such a repeating pattern can be of long enough period to produce a signal in the final lowpass-filtered output: if, in this example, the oversampling ratio were 4×, the idle tone would fall at the midpoint frequency of the output band.

This can be seen in Figure 13.62, where we've[88] calculated and plotted two complete cycles of this idle tone. Note particularly the brick-wall lowpass filtered output, where the sinusoidal idle tone can be clearly seen. Its amplitude (118 mV pp) is about 6% of full scale; that's a meager suppression of just 25 dB, totally unacceptable even for inexpensive consumer audio devices.

Delta–sigma converters intended for audio applications typically use third-order (or higher) modulators, both to suppress[89] idle tones and to increase the dynamic range (effective number of conversion bits, ENOB).

13.9.11 Some delta–sigma application examples

Enough theory! Let's look at a few simple delta–sigma application examples (with some of greater complexity in §§13.11 and 13.12). We urge the reader to have copies of the relevant datasheets in hand while reading through these examples (get them from the manufacturers' websites or, often more quickly, linked through a distributor like DigiKey, Mouser, or Newark).

Input	Delta	Sigma	Bit	Feedbk
5/8	0	0	0	-1
5/8	13/8	13/8	1	1
5/8	-3/8	10/8	1	1
5/8	-3/8	7/8	1	1
5/8	-3/8	4/8	1	1
5/8	-3/8	1/8	1	1
5/8	-3/8	-2/8	0	-1
5/8	13/8	11/8	1	1
5/8	-3/8	8/8	1	1
5/8	-3/8	5/8	1	1
5/8	-3/8	2/8	1	1
5/8	-3/8	-1/8	0	-1
5/8	13/8	12/8	1	1
5/8	-3/8	9/8	1	1
5/8	-3/8	6/8	1	1
5/8	-3/8	3/8	1	1
5/8	-3/8	0/8	0	-1
5/8	13/8	13/8	1	1
5/8	-3/8	10/8	1	1
5/8	-3/8	7/8	1	1

cycle of 16

Figure 13.61. Sequence of states of the first-order delta–sigma modulator (Figure 13.51), with a fixed dc input of 625 mV. "Delta" and "Sigma" are the input and output, respectively, of the integrator, "Bit" is the bitstream output, and "Feedbk" is the feedback DAC's output voltage.

A. An even-simpler delta–sigma ADC

In §13.9.1 we presented "the simplest delta–sigma" in the form of the integrating charge-dispensing suntan monitor. With the sophistication of the preceding sections, we can now recast that example as a first-order delta–sigma modulator with simple accumulation of the 1-bit bitstream. If that's what you need, you can implement it even more simply, with a microcontroller.

These devices (Chapter 15) are the flexible putty of electronics, and one of them can easily replace the logic portion of the previous example, as in Figure 13.63. Here the pin labeled Q is an output-port bit that swings cleanly rail-to-rail, and A_{in} is an input that sets the threshold, analogous to U_2 in Figure 13.49. It could be an internal comparator, provided in many microcontrollers (see, e.g., Figure 13.64); or it could be a low-resolution ADC internal to the microcontroller (also quite common); or, at the crudest level, it could be a logic input, whose (inaccurate) threshold voltage replaces an honest comparator.

A microcontroller implementation like this lets you do

[88] Jason, again.

[89] Higher-order modulators do not *eliminate* idle tones, they merely suppress them enough to render them almost harmless. As delta–sigma guru Bob Adams puts it, "Higher order modulators are less likely to fall into a simple repeating pattern, but it is still possible. The real benefit of higher-order modulators is that they have excellent suppression of quantization noise (because the loop gain at low frequencies is extremely high), so that, even if the quantizer decides to fall into an idle-tone-producing pattern, it will be suppressed by such a large amount that it will be buried in thermal noise."

more: at the simplest level you could write software that implements the counter and other logic needed to complete the suntan monitor (including features such as a display of progress of baking, time remaining, clock time and date, your next appointment, and so on…). At a more sophisticated level you could implement a digital filter to create a sequence of converted values, in the manner of a fully integrated delta–sigma ADC. You could, but you probably shouldn't, because many highly skilled designers are turning out excellent delta–sigma ADCs, a sampling of which we'll visit soon enough. And there is a danger in hijacking a microcontroller to control the integrator charge cycles, namely that the accuracy of sigma-delta integration depends upon a stable switch ON-time; and the stability of a converter depends also upon a stable sampling clock frequency. You must ensure that the controller gives you control over that timing; and, even so, you may find that the firmware coding becomes inconveniently constrained. These examples are intended merely to illustrate the simplest kind of delta–sigma circuit. If what you want is a high-quality converter, you should choose one from the many hundreds that are available, that are easy to use, and that are remarkably inexpensive; several of these are illustrated below.

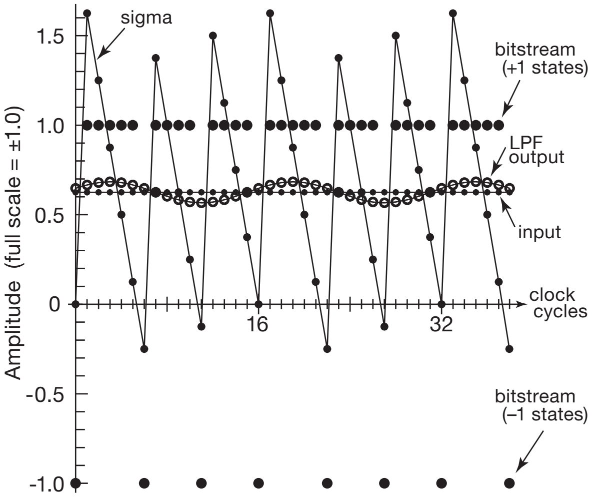


Figure 13.62. Modulator signals, along with time-aligned filtered-output waveform, for the idle-tone example of Figure 13.61. The tone is down ~25 dB, relative to full scale, and would fall at midband (half of Nyquist frequency) for 4× oversampling.

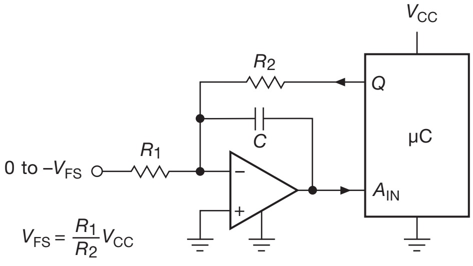


Figure 13.63. A microcontroller can replace the logic portion of a discrete delta–sigma ADC.

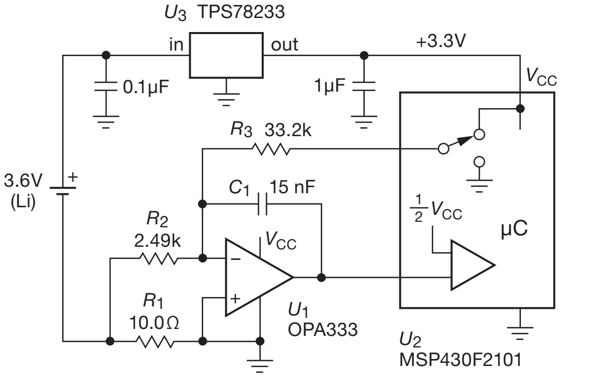


Figure 13.64. Keeping track of battery capacity with a discrete delta–sigma charge integrator.

B. Coulomb counter

Here's an example of a microcontroller-assisted converter with low quiescent current and a wide dynamic range. In all candor, an admission: we designed the converter first, then we looked for a plausible application. Figure 13.64 shows what we came up with – a low-side sensing battery "gas gauge" to keep track of the state of discharge of a lithium cell that powers a portable instrument.

Here's how it works: we used a small (10 Ω) current-sensing resistor, to limit the voltage burden to 0.25 V at the maximum anticipated load current of 25 mA (caused, for example, by switched loads such as a 350 Ω strain gauge). Then we chose a single-supply chopper op-amp (maximum offset voltage of 10 μV) to minimize the error at low currents; here the offset voltage corresponds to a sensed current error of 1 μA (max), or a dynamic range of 25,000:1. The voltage developed across the sense resistor drives the integrator through R_2, with a full-scale input current of 100 μA. (You can think of $R_1 R_2$ as a "current divider," if you like that concept.)

Next we chose R_3 to set the converter's full-scale input current ($I_{FS} = V_{CC}/R_3$), i.e., to source the full-scale current into the summing junction when the switch is on. Finally, taking a clock frequency of 10 kHz, we chose the value of the integrating capacitor C_1 such that the integrator would ramp no more than $V_{CC}/5$ in one clock cycle. It's easy to figure out that this condition makes $C = 5/fR_3$, or 15 nF.

Exercise 13.6. Make sure you understand this design, by running through the calculations of maximum load current and dynamic range, and the design of the integrator components R_2, R_3, and C_1.

The MSP430 is a low-power microcontroller series from TI. This particular variant conveniently includes a comparator whose reference input can be biased at $V_{CC}/2$, which we use in conjunction with a digital output bit that we switch between ground and V_{CC}.[90] With a clock frequency of 1 MHz the microcontroller draws 0.3 mA when active, and 25 μA when running in "low-power mode 2"; think of the latter as a "sleep" mode, during which an internal timer continues to run, and from which the processor can wake up to full alertness in one clock cycle. This is important, because it's common to put portable equipment into a low-power mode to conserve battery charge.

Note that this "Coulomb meter" keeps track of *all* loads, including the regulator's quiescent current, the microcontroller's operating current, additional loads powered from the regulated V_{CC} line, and even the current needed to run the delta–sigma's integrator op-amp itself. Exclusive of additional loads, the power budget is dominated by the processor, followed by the chopper op-amp (17 μA, typ) and the regulator (1.3 μA, max); that equates to a battery life of several months (with processor active) for a typical 1 Ah rechargeable Li-ion cell. The peripheral loads would likely reduce battery life considerably, and would typically be power switched by the processor.[91]

Note that the zero error of 1 μA (from the op-amp's worst-case offset) is completely negligible in comparison with the system current, even in sleep mode. The 25,000:1 dynamic range is extravagant for this application.

C. Three fully integrated delta–sigma ADCs

We conclude these delta–sigma examples with three elegant integrated ADCs from three different manufacturers. For more parts to consider, see Table 13.9.

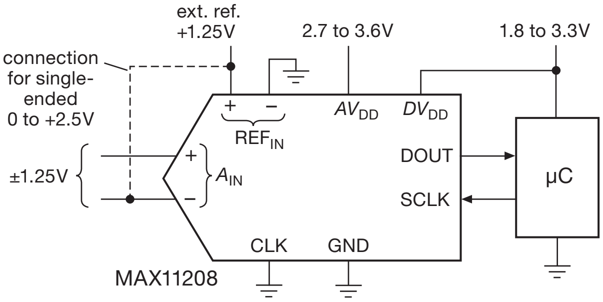


Figure 13.65. Maxim's MAX11208B 20-bit low-noise ADC with internal clock. The differential input can be reconfigured for single-ended 0 to $+2V_{ref}$ by connecting the A_{IN} input to the reference, as shown.

Maxim MAX11208B: inexpensive 20-bit ADC

From Maxim comes this compact (10-pin) converter (Figure 13.65), both inexpensive ($3.75 in single quantities) and quiet (0.7 μVrms noise). It delivers true 20-bit resolution at a leisurely 13.75 sps, and it has 80 dB rejection of both 50 Hz and 60 Hz powerline frequencies with its accurate internal system clock (enabled by grounding the CLK pin).[92] The full-scale conversion range of the differential input is $\pm V_{ref}$, over the common-mode range of 0 to $+AV_{DD}$. It performs a gain and offset calibration at power-on, and also on request via an amusing digital command (two extra SCLKs tacked onto the end of the normal serial readout). In common with most converters, it has a separate digital supply pin for compatibility with low-voltage digital logic.

Analog Devices AD7734: versatile 24-bit ADC

From Analog Devices comes this 4-channel multiplexed 24-bit converter (Figure 13.66), with an unusual input structure of trimmed resistors that provides a full ±10 V input range (while operating from a single +5 V supply), and in fact with an over-range bit that extends the conversion range to 11.6 V, thus permitting accurate gain calibration at full-scale input. It is extremely tolerant of input overdrive: to ±16.5 V without affecting the accuracy of the other inputs, and to ±50 V without damage.

There's lots of flexibility in the conversion process, with programmable selection of parameters like filter length, auto-zero, and "chopping" mode. The latter consists of differential-input reversal on successive conversions, with the outputs averaged to cancel offsets in the buffer and delta–sigma modulator. With chopping mode enabled and with the longest filter setting, the converter delivers an

[90] This processor provides several "capture control registers" by which we can ensure that those pulses are of accurate and stable duration, a necessary condition for delta–sigma integration.

[91] With a much smaller battery, one could put the processor into sleep mode as well, waking up only often enough to ensure that the system current during sleep does not cause the integrator ramp to saturate between naps. For this system the $\sim 45\,\mu$A total sleeping quiescent current would require measurements every 80 ms to limit the unobserved ramp amplitude to 1 V. This microcontroller can do that easily, because it has the nice property of waking up in a single 1 μs clock cycle.

[92] The –A suffix version runs at 120 sps, with poor rejection of powerline frequencies, but deep rejection at 120 Hz and its harmonics.

Table 13.9 Selected Delta-sigma A-to-D Converters^{a}

									DC Errors				Power Supply						Pkgs, Pins						
			Conv						Offset^{b}		Gain														
	ADCs per		Rate max	Channels	Single-		Pwr typ	PGA	typ	drift,typ	typ	drift, typ	min	max	Internal	External								Cost qty25	
Part #	pkg	Bits	(ksps)	per ADC	ended	Interface^{q}	mW	Gains	(µV)	(µV/°C)	(ppm)	(ppm/°C)	(V)	(V)	Ref	Ref	Sequencer	Int osc	DIP	SOIC	TTSOP	SOT-23	Smaller	($US)	Comments
ADS1158	1	16	125	16	f	S	42	-	25	1^{t}	10000	2	2.7	5.25^{o1}	-	•	•	g,h	-	-	-	-	48	11.77	A
ADS1100	1	16	0.128	1	•	I	0.2	1-8	2500	1.5	1000	2	2.7	5.5	-	-	-	•	-	-	-	6	-	4.80	B
ADS1115	1	16	0.86	4	•	I	0.5^{e}	1-16	50	0.4	100	5	2	5.5	•	-	-	•	-	-	10	-	10	5.53	C
AD73360	6	16	4	1	•	S	86^{e}	1-80	20000	-	10000	-	2.7	5.5	•	-	-	-	-	28	44	-	-	9.24	-
MAX11208B	1	20	0.12	1	?	S	0.8	-	3	0.05	3^{c}	0.05	2.7	3.6	-	•	-	•	-	-	10	-	-	2.18	-
CS5513	1	20	0.33	1	?	S	1.9	-	120	0.06	-	1	0	6^{o3}	-	•	-	•	-	8	-	-	-	4.96	D
MCP3551	1	22	0.014	1	?	S	0.3^{e}	-	3	0.1	2	0.028	2.7	5.5	-	•	-	•	-	8	8	-	-	3.31	E
LTC2412	1	24	0.008	2	?	S	0.2	-	0.5	0.01	2.5	0.03	2.7	5.5	-	•	•	•,h	-	-	16	-	-	6.34	F
AD7730	1	24	0.2	2	?	S	65^{m}	four	2^{c}	0.5	100^{c}	2	4.75	5.25	-	•	-	g,h	24	24	24	-	-	16.10	G
AD7794	1	24	0.47	6	•	S	1^{z}	INA	1^{c,k}	0.01^{k}	2^{c}	1	2.7	5.25	•	•	-	•,h	-	-	24	-	-	10.80	H
MAX11210	1	24	0.48	1	•	S	0.25	1-16	30^{m,c}	0.05	20^{m,c}	0.05	2.7	3.6	-	•	-	•,h	-	-	16	-	-	3.32	J
ADS1246	1	24	2	1	?	S	1.4^{e}	1-128	15^{m,c}	0.05^{b}	50	1	2.7	5.25^{o1}	-	•	-	•,h	-	-	16	-	-	8.38	K
CS5532-BS	1	24	3.84	2	?	S	70	1-64	6	0.64^{b}	16	2	0	5.5^{o2}	-	•	-	g,h	-	-	20	-	-	12.80	L
AD7190	1	24	4.8	4	p	S	1^{b}	1-128	0.5^{k}	0.005	50^{m}	1	4.75	5.25	-	•	•	•,g,h	-	-	24	-	-	10.89	M
ADS1259	1	24	14	1	d	S	13	-	40	0.05	500^{y}	0.5	4.75	5.25^{o1}	•	•	-	•,g,h	-	-	20	-	-	12.15	N
AD7734	1	24	15	4	f	S	85	-	13000^{m,x}	2.5	4500^{m,x}	3.2^{m}	4.75	5.25	-	•	•	g,h	-	-	28	-	-	15.84	O
ADS1210	1	24	19.5	1	?	S	26	1-16	0.15^{c}	1	0.06^{c}	0.15	4.75	5.25	•	•	-	g,h	18	18	-	-	-	22.84	P
ADS1258	1	24	23.7	16	w	S	42	-	0.2	0.02	50	2	4.75	5.25^{o1}	•	•	•	g,h	-	-	-	-	48	18.80	Q
ADS1298	8	24	32	1	d	S	10	1-12	500	2	2000	5	2.7	5.25	•	•	-	•	-	-	64	-	64	40.40	R
ADS1278	8	24	144	1	d	S	530	-	250	0.8	1000	1.3	v	v	-	•	-	-	-	-	-	-	64	39.57	S
AD7764	1	24	312	1	-	S	300	-	240	0.6	180	0.5	u	u	-	•	-	-	-	-	28	-	-	16.94	T
ADS1672	1	24	625	1	-	LC	350	-	2000^{m}	2	10000	2	4.75	5.25	-	•	-	-	-	-	-	-	64	22.06	-
AD7760	1	24	2500	1	-	P	960	-	400	0.3	160	2	r	r		•	-	-	-	-	-	-	64	42.49	U
ADS1675	1	24	4000	1	-	LC	575	-	5000^{m}	4	10000^{m}	4	4.75	5.25	-	•	-	-	-	-	-	-	64	35.88	V
ADS1281	1	32	4	1	d	S	12	-	1^{c}	0.06^{t}	2^{c}	0.4	4.75	5.25^{o1}	-	•	-	-	-	-	24	-	-	49.68	W

Notes: (a) sorted by precision and speed; all except AD7734 have differential inputs; all have no missing codes except as noted. (b) at minimum PGA gain. (c) after calibration. (d) unused input biased at mid-supply. (e) at V_S=3.3V. (f) pseudo-diff'l. (g) external bare xtal. (h) opt ext osc input. (k) in chopper mode. (m) min or max. (n) 0.1 to V_-0.1 with buffers enabled. (o1) has neg supply, 0 to –2.6V, with 5.25V max total supply. (o2) has neg supply, 0 to –3.5V, with 5.5V max total supply. (o3) has neg supply, 0 to –6V, with 6V max total supply. (p) 2 differential, 4 pseudo-diff'l with single common return. (q) S=SPI, P=parallel, LC=LVDS or CMOS serial. (r) five supplies: +5 ±5%, +2.5 ±5%, +3.15 to +5.25 (2x), and +1.67 to +2.7. (s) at max decimation. (u) four supplies: +5 ±5% (3x), +2.5 ±5%. (v) 4.75-5.25V and 1.65-2.2V. (w) 8 diff'l, 16 single-ended (pseudo-diff'l); (x) before cal. (y) 2ppm after cal. (z) 0.4mW unbuffered.

Comments: **A:** 16 single-ended (pseudo-diff'l) or 8 diff'l; MUX output and ADC input pins. **B:** internal clk; self-cal. **C:** 4 single-ended (not pseudo-diff'l) or 2 diff'l; auto-shutdown in single-shot mode. **D:** CS5512 has ext osc; CS5510/11 = 16-bit. **E:** low-noise, 2.5µVrms; auto-shutdown in single-conv mode. **F:** no latency; 0.8µVrms noise; a favorite. **G:** bridge subsystem with offset DAC and ac excitation output. **H:** low-noise; opt input buffer for hi-Z; INA PGA; includes 2 curr srcs; chopper mode; auto-shutdown in single-conv mode; int ref 4ppm/°C typ; AD7793 has fewer channels; a favorite. **J:** opt input buffer for hi-Z. **K:** low noise; 20-pin ADS1247 has 2 channels, 28-pin ADS1248 has 4 channels, both have int 10ppm/°C ref. **L:** CS5534 4-ch diffl; industrial workhorse; 6nV/√Hz. **M:** low noise. **N:** off-scale detectors. **O:** has chopper mode; AD7732 has two diff inputs. **P:** ADS1211 has 4 ch diff'l MUX; sample-rate boost to 312kSps; no missing codes to 22 bits. **Q:** low noise; pins for MUX diff'l output and ADC input. **R:** biopotential measurements (EKG, EEG, etc); ADS1294=quad. **S:** ADS1274 = quad. **T:** diff'l input buffer; AD7765 to 156kSps. **U:** diff'l input buffer. **V:** pin-programmed (no registers); LVDS or CMOS serial output. **W:** no missing codes to 31 bits.

effective 21-bit conversion at 372 sps for input signals of ±10 V full scale; this figure comes from a consideration of the converter's noise level, which is 9.6 μVrms under these conditions (21 bits is the ratio of a ±10 V span to ~ 10 μV). The datasheet specifies resolution alternatively as "peak-to-peak resolution in bits," which for these same conditions is listed as 18.1 bits. This turns out to be the more conservative specification, which is explained as "representing [bit] values for which there will be no code flicker within a 6-sigma limit." In other words, you can rely on any single conversion to be accurate to 18 bits, taking into account the fact that the occasional peak excursions of a signal of given rms noise voltage are substantially larger than V_{rms}.[93] If you know the rms noise voltage V_{rms},

[93] The datasheet for Cirrus' CS5532 explains it this way: " 'Noise-free resolution' is not the same as 'effective resolution.' Effective resolution is based on the RMS noise value, while noise-free resolution is based on a peak-to-peak noise value specified as 6.6 times the RMS noise value."

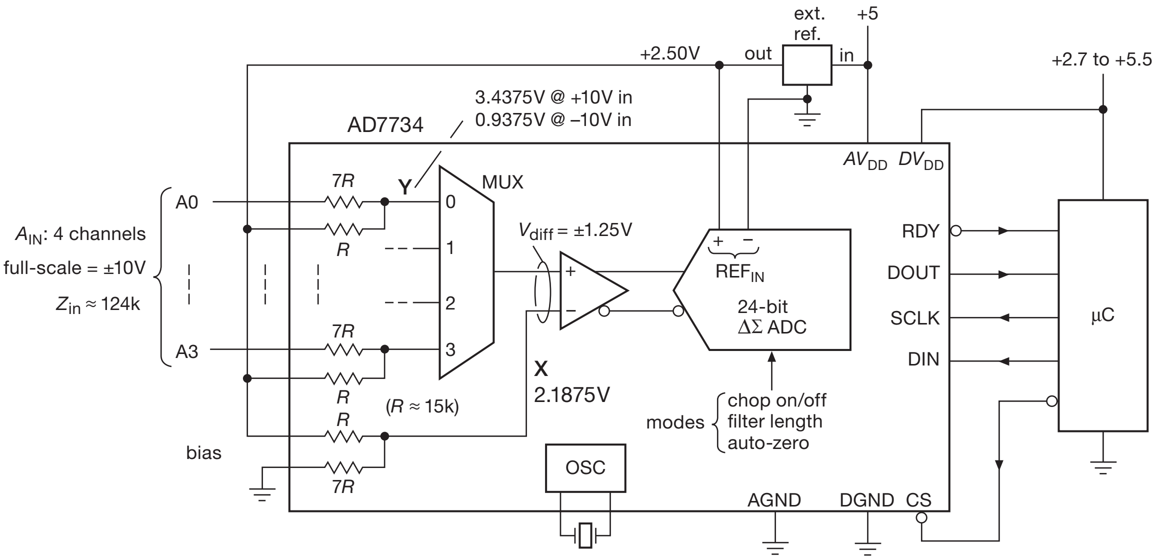


Figure 13.66. Analog Devices AD7734 multiplexed 4-channel 24-bit converter (21-bit ENOB) with wide input-voltage range and robust overdrive tolerance, courtesy of Analog Devices, Inc.

you can calculate the noise-limited effective resolution as $\text{ENOB} = \log_2 \left(V_{\text{span}}/V_{\text{rms}}\right) = 1.44 \log_e \left(V_{\text{span}}/V_{\text{rms}}\right)$; from that you get the peak-to-peak resolution by subtracting 2.7 bits.

This converter can be operated at conversion rates to 12 ksps, with correspondingly degraded resolution. It has maximum offset and gain drifts of ±2.5 μV/°C and ±3.2 ppm/°C, respectively. It comes in a 28-pin package and is currently priced at about $15 in single quantities.

Cirrus CS5532 high-performance “industrial” ADC

From Cirrus Logic (formerly Crystal Semiconductor) comes the long-lived (circa 1999) CS5532-BS 24-bit delta–sigma converter (see Figure 13.67), with a chopper-stabilized PGA (gains of 2, 4, 8, 16, 32, and 64) and with particularly good noise, drift, and linearity characteristics: $e_n = 6.4\,\text{nV}/\sqrt{\text{Hz}}$ (typ) at 0.1 Hz with $G = 64$,[94] $i_n = 1\,\text{pA}/\sqrt{\text{Hz}}$ (typ), $\Delta V_{os} = 15\,\text{nV}/°\text{C}$ (typ) with $G = 64$, full-scale drift of 2 ppm/°C (typ), and ±0.0015% (max) nonlinearity. It can convert at rates from 6.25 sps to 3.8 ksps. At the slowest rates it delivers noise-free resolution ranging from 20 bits (for $G = 64$) to 23 bits ($G \leq 8$); or, if you prefer, corresponding “effective resolutions” (ENOBs) of 23 and 24 bits, respectively.

With its low-noise, high-gain PGA and ability to run from split ±3 V supplies,[95] this converter has what it takes to process the low-level signals from a thermocouple (∼40 μV/°C) or a strain gauge (full scale $\Delta V \approx \pm 2$ mV per volt of bridge excitation). With a PGA gain of 64, the full-scale span is ±2.5 V/64, or ±40 mV, and an LSB (at 20-bit resolution) corresponds to 80 nV which is 500× less than the thermal-voltage change corresponding to a temperature change of 1°C. Likewise, at this gain a 20-bit LSB corresponds to 0.0008% of the strain gauge’s full scale. At this gain, full-scale inputs from these sensors stay within the conversion range. Evidently there’s no need for external front-end gain stages or the like. This converter costs about $16 in unit quantities.

In our circuit we balanced the thermocouple signals about ground to minimize the effects of common-mode pickup in the lead wires, which are often unshielded. And for both sensors we added a simple *RC* filter (time constant 0.1 ms) to suppress spikes and protect the inputs. We chose the ADR441 because we needed a low-dropout voltage reference that would operate with 500 mV of headroom.

D. Pro-audio ADCs

Delta–sigma converters are adored by the professional audio community, owing to their combination of resolution,

[94] Analog Devices’ AD7190 comes close, at $8.5\,\text{nV}/\sqrt{\text{Hz}}$

[95] Other high-resolution delta–sigma converters that allow bipolar input supplies and signals include (a) the 32-bit ADS1281, with ±2.5 V analog supplies and input signals to those same limits (but sadly not to ±3 V, with their total supply limit of 5.25 V), and (b) three 24-bit converters: the 16-channel ADS1258, the ADS1259 with its 2 ppm voltage reference, and the ADS1246 family with internal PGA.

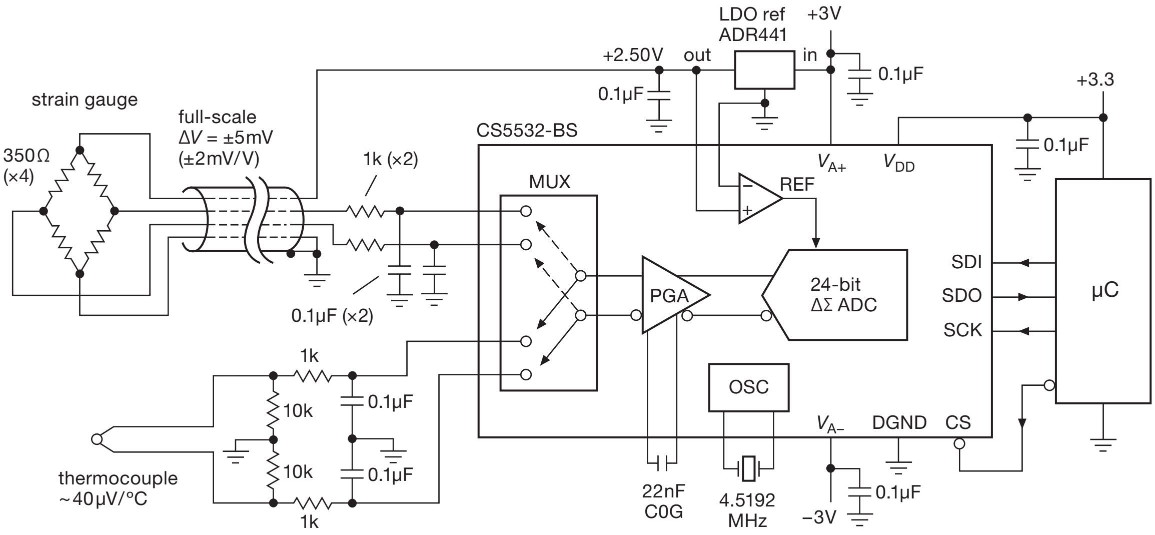


Figure 13.67. Cirrus' CS5532-BS precision low-noise 24-bit converter. The PGA permits gains of 1, 2, 4, ..., 64. This circuit omits the cold-junction compensation. The Maxim MAX31855 includes this important compensation for seven thermocouple types; see item 44 in §15.8.2 and accompanying figure.

Table 13.10 Selected Audio Delta-Sigma ADCs

	Channels	Bits	f_{samp} max (kHz)	SNR[k] (dB[a])	THD+N[k] (dB[b])	Supply V_{CC} (V)	Supply P_{diss} (mW)	inputs, other
PCM1870A	2	16	50	90	-81	3	13	mic preamp
CS43432	4[c]	24	96	105	-98	5	600	diff or SE; codec
PCM2906[h]	2[d]	16	48	89	-80	5	280	SE; codec, USB
AK5384	2	24	96	107[e]	-94	5	275	diff
AK5388	4	24	192	120	-107	5	590	diff
AK5394A	2	24	192	123	-94[f]	5	705	diff; popular
CS5381	2	24	192	120	-110	5	260	diff; preferred
AD1974	4	24	192	105	-96	3.3	430	diff, with PLL
PCM4204	4	24	216	117	-103	4	615	diff
PCM4222[g]	2	24	216	123	-108	4	340	diff; configurable

Notes: (a) A-weighted filter. (b) at max sample rate. (c) codec: 4 in + 6out. (d) codec: 2 in + 6 out. (e) at 48kHz. (f) -110dB at 96kHz. (g) consider also the PCM4420. (h) stereo codec (ADC + DAC) with S/PDIF to USB in/out. (k) SNR and THD+N is with respect to full-scale; SNR is typically measured with a -60dB input signal, processed with an A-weighted filter; THD is typically measured at 1kHz, with a -1dB signal.

inherent anti-aliasing, noise shaping, and monotonicity. (See Table 13.10.) If you open up pretty much anyone's high-quality audio gear, you'll probably find a circuit based around a 24-bit, 192 ksps 128× oversampling delta–sigma ADC; and chances are it will be a part from Cirrus (e.g., the CS5381) or AKM (e.g., the AK5394A). These parts seem to have "long legs" – they have been around for many years and represent good value in terms of price and performance.

Audio ADCs universally use delta–sigma technology, but they differ greatly from their industrial ADC counterparts in the delta–sigma table. They generally have poor gain accuracy (5% to 10%) and dc offset (∼25 mV), in part because these don't matter in the audio field. On the other hand, they do offer 0.1 dB or 1% stereo-channel gain *matching*. They're usually wired with ac-coupling, and they also have an internal digital highpass filter (typically ∼1 Hz).[96] They are meant to operate at specific audio data rates, and they employ specialized audio PCM data-output interfaces (I^2S, TDM, etc.). Compared with industrial ADCs they have high latencies (data-output delays) of 12 to 63 sample intervals, even though they may advertise "low latency" (by which they mean small compared with, say, one millisecond of time delay). They have unique audio specifications, like A-weighted SNR, and spectral-analysis-derived THD+N distortion specs.

Figure 13.68 shows a straightforward signal-conditioning circuit, adapted from AKM's evaluation board, not unlike the innards of many commercial audio digitizers. The 5534 op-amp seems to be the perennial

[96] You'll discover this if you bypass the coupling capacitors. However, many of these converters offer a dc mode, and some (like the CS5381 and AK5394A) provide a logic-triggered dc-auto-zero function, which is convenient for very slow or dc applications.

favorite (it's been around for at least three decades), inexpensive and "good enough." Although you can do better in terms of distortion, what seems to matter most to audiophiles is dynamic range (set by the ADC resolution and the noise level); harmonic distortion at 0.001% is inaudible. However, we prefer full differential signal-conditioning circuits for high-performance audio ADCs, as described expansively in Chapter 5 (see Figures 5.70 and 5.102).[97] See §5.14.2E for more about pro-audio signal levels.

13.10 ADCs: choices and tradeoffs

The good news is that world of ADCs is a world of richness, with many choices. The bad news is, well, that the world of ADCs is a world of richness, with many choices. In the following sections we offer some guidance, to help navigate the welter[98] of choices.

13.10.1 Delta–sigma and the competition

A. Analog-to-digital converters

Delta–sigma converters are one of several ADC conversion technologies, which (as we've seen) include also (a) dual-slope and quad-slope integrating converters, (b) successive-approximation converters, (c) flash converters, and (d) pipelined flash converters.

Low speed For "voltmeter-speed" conversion (e.g., 10/sec), multislope integrating converters have been the perennial favorite, but their dominance is being challenged by excellent ΔΣ's from LTC (e.g., the LTC2412: 24 bit) and ADI (e.g., the AD7732: 24 bit, ±10 V range), among others.

Medium speed (to ∼100s of ksps) Delta–sigma converter ICs dominate at resolutions above 16 bits, with nice products from companies like Cirrus and AKM (e.g., the AK5384: 24 bit, 96 ksps, 4 channels, or the converters in Figure 13.68). There are many good delta–sigma audio ADCs, but their dc specs tend to be poor (several percent) or nonexistent. For resolutions of 16 bits or less, consider the highly usable successive approximation converters.

Medium-high speed (to a few Msps) Here there is a fierce battle between ΔΣ's and successive-approximation ADCs using a switch-capacitor charge-redistribution ADC: comparable accuracy, but the SARs are faster (e.g., the ADI AD7690: 18 bit, 400ksps; a member of the AD76xx/79xx PulSAR™series). See the "shootout" below.

High speed (to 100s of Msps) For these speeds you choose a pipelined flash converter (previously known as "half-flash"), with successive stages of low-resolution flash conversion operating on the analog residue of the previous stages, or with the "folding-amplifier" architecture (Figure 13.27). Pipelining results in high throughput, but with the latency of typically 10 sample intervals. Examples are the ADI AD9626 (12 bit, 250 Msps) and the TI ADS6149 (14 bit, 250 Msps).

Breakneck speed (>250 Msps) Variants on the basic flash (such as folding/interpolating) rule the roost here, but the tradeoff is modest resolution (6–10 bits). National has some nice ones, for example the ADC08D1520 (8 bits, 3000 Msps), ADC10D1500 (ditto, 10 bits), and ADC12D1800 (12 bits, 3600 Msps). These kinds of converters are used widely in 'scope front-ends,[99] and in digital radio. At the extreme (and we're not sure how they do it), Fujitsu is featuring a 56,000 Mbps (!) 8-bit converter.[100]

B. Digital-to-analog converters

Competing DAC technologies are (a) *R*–2*R* ladder, (b) linear resistor ladder with switch array, and (c) current-steering switch array.

Highest linearity Delta–sigma DACs are best, with accuracy and linearity to 20 bits at audio speeds, and sometimes with excellent dc specs as well (e.g., the millisecond-speed 20-bit TI DAC1220); however, watch out for broadband and clock noise (the DAC1220 has $\sim 1000\,\text{nV}/\sqrt{\text{Hz}}$ at 1 kHz compared with $\sim 10\,\text{nV}/\sqrt{\text{Hz}}$ for resistor ladder converters).

Medium speed, high accuracy Many excellent *R*–2*R* and linear ladder DACs compete, for example:

- TI DAC8552 (dual 16 bit, serial in, voltage out, ext

[97] Interestingly, both AKM and Cirrus use this simple scheme, and inexpensive '5534 amplifiers, in their ADC evaluation kits. However, the Cirrus "reference design" kit substitutes the lower-noise LT1128 op-amp. By contrast, TI uses a true differential amplifier (the OPA1632) in the evaluation kit for their comparable PCM4222 audio delta–sigma converter.

[98] Merriam–Webster: "A large number of items in no order; a confused mass. *Synonyms:* confusion, jumble, tangle, mess, hodgepodge, mishmash, mass; *informal* rat's nest.

[99] Currently available digital 'scopes achieve analog bandwidths of 32 GHz with sampling rates of 80 Gsps (e.g., Agilent 90000X series), figures sure to increase with time.

[100] Evidently they're so excited about it that they have neglected, thus far, to assign a part number.

Table 13.11 Selected Audio D-to-A Converters[a]

										Dig Filter		Pwr Supplies						
Part #	Bits	# Channels	f_{samp} max (ksps)	THD+N typ (dB)	SNR typ (dB)	Technology[c]	Output[f]	Volume Control	Pwr Amp?	Ripple max (±dB)	Stop[n] min (dB)	Voltage analog (V)	Voltage digital (V)	Power[o] typ (mW)	Control Interface[b]	Pkg	Price qty 25 ($US)	Comments
AK4386	24	2	96	84[d]	100	mbDS	Vse	-	-	0.01	64	2.2-3.6	-	20	H	TSSOP-16	1.00	A,J
CS4334/5/8/9	24	2	96	88[d]	88[d]	DS	Vse	-	-	0.2	50	5	-	75	-	SOIC-8	1.63	B,J
PCM1753	24	2	192	94	106	mbDS	Vse	-	-	0.5	50	5	-	150	S	SSOP-16	1.65	C
TLV320DAC3100	24	2	192	82[e]	95	mbDS	SP/HP	•	p	-	-	3.3	1.8	-	I	QFN-32	3.03	D
PCM1789	24	2	192	94	113	C-seg	Vdiff	-	-	0.0018	75	5	3.3	168	H,I,S	TSSOP-24	4.38	E
LM49450	24	2	192	64g	88	DS	SP/HP	•	q	-	-	2.7-5.5	1.8-4.5	50[h]	I	WQFN-32	5.15[k]	F
AK4358	24	8	192	92[d]	112	mbDS	Vdiff	•	-	0.02	54	5	5	560	I,S	LQFP-48	5.22	G,J,N
PCM1690	24	8	192	94	113	DS	Vdiff	•	-	0.0018	75	5	3.3	620	H,I,S	HTSSOP-48	5.23	G
PCM4104	24	4	216	100[d]	118	DS	Vdiff	-	-	0.002	75	5	3.3	236	H,S	PQFP-48	7.74	H
CS4398	24	2	192	107	120	mbDS	Vdiff	•	-	0.01	102	5	1.8-5	192	H,I	TSSOP-28	8.86	N
PCM1794A	24	2	192	102[d]	127	AS	Idiff	-	-	0.00001	130	5	3.3	335	H	SSOP-28	10.83	L
ADAU1966	24	16	192	98	118	DS	Vdiff	•	-	0.01	68	5	2.5	521[m]	I,S	LQFP-80	11.71	J,M
AD1955	24	2	192	110	120	DS	Idiff	•	-	0.0002	110	5	5	210	S	SSOP-28	12.35	J,N
AK4399	32	2	192	102[d]	123	DS	Vdiff	•	-	0.005	95	5	5	530	S	LQFP-44	15.00	P
PCM1704K	24	1	96	102[d]	120	R-2R	Ise	-	-	-	-	±5	-	175[d]	-	SOIC-20	68.20	Q

Notes: (a) in order of increasing price. (b) H="hardware," I.e., pin-programmable; I=I^{2}C; S=SPI. (c) AS=TI "advanced segment"; C-seg=current-segment; DS=delta-sigma; mbDS=multibit delta-sigma; R-2R=ladder. (d) at 96ksps. (e) 48ksps audio. (f) Idiff=differential current; Ise=singled-ended current; SP/HP=speaker or headphone; Vdiff=differential voltage; Vse=single-ended voltage. (g) 0.6W into 8Ω, with V_S=5V. (h) analog supply only; class-D amp has 80% effy. (k) qty 100. (m) at 48kHz. (n) stopband attenuation, in sharp rolloff mode, at f/f_S=0.546. (o) at 192ksps. (p) 2.5W mono. (q) 1.9W stereo, 1% THD into 4Ω with V_S=5V. (t) typical.

Comments: A: low power. **B:** 8-pin, "entry level"; choose p/n for data format. **C:** PCM1754=H/W interface. **D:** stereo headphone drivers, and class-D mono spkr amp, 2.7-5.5V. **E:** high performance consumer audio/video. **F:** portable consumer audio; 100dB SNR at headphone output. **G:** octal, multichannel consumer audio/video. **H:** quad, professional and high-end audio, high performance. **J:** low jitter. **L:** premium; DSD1794A=IIC/SPI. **M:** 16 channels; automotive, etc. **N:** supports SACD output. **P:** premium. **Q:** legendary, the best of the legacy R-2R audio DACs; needs external filter.

ref, very low glitch, 10 μs settle; DAC8560/4/5 similar, with int ref)

- ADI AD5544 or TI DAC8814 (quad 16 bit MDAC, serial in, current out, 0.5–2 μs settle with external *I*-to-*V* op-amp)
- LTC1668 (16 bit, parallel in, differential current out, 20 ns settle into 50 Ω as "voltage out")
- TI DAC9881 (18 bit serial in, rail-to-rail voltage out, ext ref, low-noise, 5 μs settle)

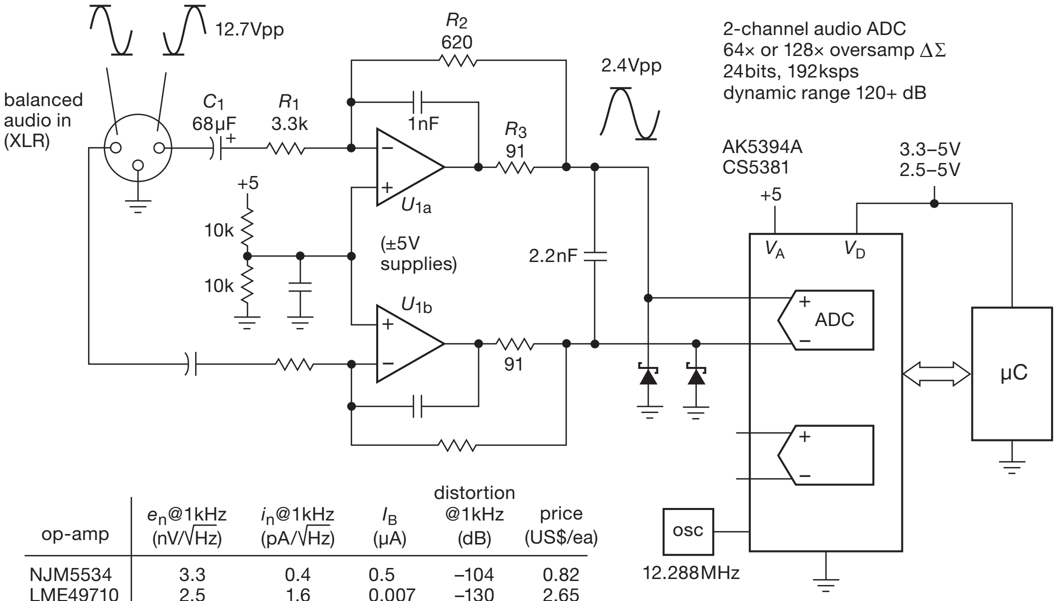
Highest speed Here you can't beat current-steering converters, for example the TI DAC5681/2 (16 bit, 1 Gsps) or the ADI AD9739 (14 bit, 2.5 Gsps).

C. Interlude: shootout at the ADC corral

To illustrate the important performance differences between delta–sigma and successive-approximation ADCs, we invited two able and well-matched contenders from the same training camp (Analog Devices) to slug it out. They came forward with their respective (and respectable) specifications, which look like this:

	SAR AD7641	ΔΣ AD7760	units
Introduced	2006	2006	year CE
Price	$47	$53	US dollars
Conv rate	2.0	2.5	Msps
Samp freq	2	40	MHz
Alias above	1	20	MHz
Resolution	18	24	bits
Zero error	60	200	ppm max
" tempco	0.5	0.1	ppm/°C typ
Gain error	0.25%	0.016%	max/typ
" tempco	1	2	ppm/°C typ
SNR	93	100	dB typ
THD	−101	−103	dB typ
INL	±7.6	±7.6	ppm typ
Data delay	0.5	12	μs
Reference	int	ext	
Supplies	1	3	
Power	75	960	mW

The delta–sigma contender put in a few good jabs, with its superior resolution and the ease with which the user could



op-amp	e_n@1kHz (nV/√Hz)	i_n@1kHz (pA/√Hz)	I_B (μA)	distortion @1kHz (dB)	price (US$/ea)
NJM5534	3.3	0.4	0.5	–104	0.82
LME49710	2.5	1.6	0.007	–130	2.65
AD797A	0.9	2.0	0.25	–130	8.36

Figure 13.68. The AKM and Cirrus delta–sigma ADCs are ubiquitous in professional audio digitizers. They are often implemented with a simple analog frontend like this, though a better approach would exploit a true differential amplifier (§5.17). Reproduced with permission of Asahi Kasei Microdevices, Tokyo, Japan.

design an input bandwidth-limiting (anti-aliasing) lowpass filter (owing to its 8× oversampling ratio). The SAR responded that the number of bits isn't a big deal, it's really *linearity* (for which both are equal) that counts. And, by the way, the SAR produces output bits 25 times faster, with its superior low latency. The sigma–delta counterpunched with its claim to superior SNR, to which the SAR parried with disapproval at the ΔΣ's need for at least two power supplies and an external reference, and its profligate waste of 13× as much power. The sigma–delta, though somewhat chastened, bounced back with a claim of 15-fold smaller gain error, to which the SAR replied that the delta–sigma was cheating because it relied on a gain-correction register to do "smart" calibration. This set up the delta–sigma to deliver the ultimate insult, namely that the SAR wasn't even smart enough to cheat. Both contenders claimed victory (as they staggered back to their respective corners), but onlookers judged it a close match, with good punches on each side.

13.10.2 Sampling versus averaging ADCs: noise

Delta–sigma converters are inherently *integrating*; that is, a measurement takes account of the varying signal throughout the conversion time; you can think of this as simple *averaging*. With a SAR converter, by contrast, the instantaneous voltage of the input signal is captured in a track-and-hold (during the so-called *aperture* time) when the converter is triggered. This distinction has some important consequences, among them the ability of SAR converters to operate at exceptionally low average power consumption when sampling at a leisurely rate (see next section).

Another consequence of importance is the effective bandwidth at which the input signal is sampled. A short aperture corresponds to a wide bandwidth, and vice versa. Intuition serves well, here: high frequencies are washed out during a long averaging interval, whereas a quick sample can record the signal's amplitude as it gyrates rapidly. Put another way, averaging a signal over some time interval τ acts like a lowpass filter, whose bandwidth is very roughly of order $1/\tau$; mathematically speaking, they are related by the Fourier transform.[101]

To make these statements quantitative, look at Figure 13.69, which illustrates the lowpass filter function of an averaging window of time duration T. Low frequencies pass through, but higher frequencies suffer from averaging;

[101] And particularly by the convolution theorem, where the sampling interval is represented by a unit-amplitude rectangular window in time. Its Fourier transform is the sinc function, $(\sin t)/t$, with a first zero at $f=1/\tau$.

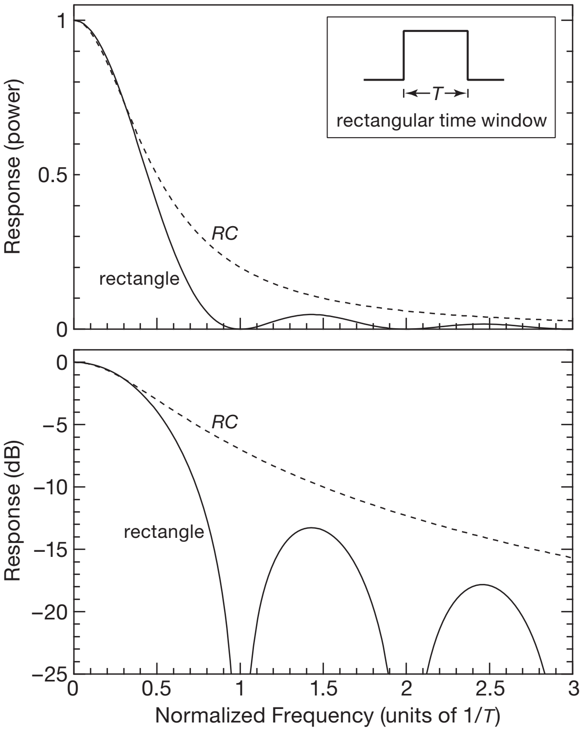


Figure 13.69. Spectrum of a rectangular pulse $\Pi(t)$ of length T. An input signal, gated (windowed, or multiplied) by $\Pi(t)$, is effectively lowpass filtered by the indicated power spectrum. An RC lowpass filter with 3 dB rolloff at $f{=}1/2T$ is shown for comparison.

a signal of frequency $f{=}1/T$ completes one full cycle during the window's time duration T, thus averages to zero. Additional zeros occur at multiples of $1/T$, where a signal of the corresponding frequency completes an integral number of cycles.[102]

So a short window admits wideband noise that may be present, degrading the accuracy of an intrinsically slow signal that would benefit from averaging. Keep this in mind when designing a conversion circuit, particularly one that operates on intermittent samples of a slowly varying signal (for example, a temperature sensor or strain gauge). It's OK to use a fast-sampling successive-approximation ADC, as long as you are willing to add a lowpass filter at the input. With an averaging converter (delta–sigma, dual-slope, or multislope) you get that benefit for free.

13.10.3 Micropower A/D converters

Small battery-powered devices often need some information about the real world, which they can get from a sensor signal and a low-power ADC. Frequently a simple 8- or 10-bit ADC included in the microcontroller IC will do, but for those who need better performance we offer a selection of micropower ADCs in Table 13.6 on page 916. The table lists both SAR and $\Delta\Sigma$ converters, with most of the entries appearing also in either Tables 13.5 or 13.9. Let's take a look at a comparison of these converter types for a typical micropower application.

SAR types are known for fast conversion speeds, but this comes at the expense of higher power consumption. SAR ADCs capture a sample of the signal when they are triggered, allowing you to turn the sensor off immediately; this saves power with something like a hungry strain-gauge bridge. Fast SAR converters draw more current, but they also finish faster and go to sleep. For example the AD7685 consumes 2.7 mW during continuous 16-bit conversions at 200 ksps (its maximum, when running on 3 V); but for a sensor application we can make measurements far less frequently, say at 100 samples per second where the average power dissipation is just 1.4 μW (2000× less). As noted in the table, for most SAR types the power dissipation is proportional to the sample rate, so there are plenty of other low-power candidates to be found in Table 13.5.

As remarked above, delta–sigma converters are integrating by nature, and they take longer to perform a conversion. Furthermore, a 16-bit conversion can take 16 times longer than a 12-bit conversion. But $\Delta\Sigma$ ADCs generally consume less operating power than comparable SARs. The delta–sigma MCP3425 dissipates 0.44 mW when operating continuously at its maximum 15 samples/second (for 16-bit conversions), six times less than the SAR above running at its maximum 200 ksps. At this stage you may conclude that the delta–sigma is the low-power winner. But the comparison is lopsided, because the sampling rates differ by a factor of more than 10,000. Note, by the way, that these power requirements are much larger than the promising figures you may find on the datasheet, where for example the MCP3425 claims the ability to operate at power levels as low as 1.8 μW average; but there's a gotcha, because that rosy figure applies to operation in 12-bit mode, and at just one sample per second.

Low-power ADC shootout. To compare these two converters fairly, then, let's assume we want to make ten measurements per second at 16-bit resolution, and we want to choose the converter that minimizes the power dissipation. At that rate the delta–sigma converter requires 290μW

[102] It is this complete rejection of signal frequencies at $1/T$ and all its harmonics that lets you make benchtop DMM measurements without worrying about powerline pickup: the DMM's integrating interval is chosen to be an integral number of powerline cycles (PLCs), recall §13.8.3, and particularly Figure 13.42.

average power, to be compared with the SAR's required 0.14μW. The SAR beats the pants off the $\Delta\Sigma$ – it's using just 1/2000 of the power! On this figure of merit alone, it's a slam-dunk.[103] But there's more to consider. The delta–sigma integrating conversion takes 66 ms, which yields a quieter measurement than a SAR ADC sampling the signal in a fraction of a microsecond, as described in §13.10.2.[104] By comparison, the AD7685 SAR mentioned above can perform 2000 measurements for the same total energy as one $\Delta\Sigma$ measurement – but you may need to take and average all of them to reduce the noise.

We always suggest a careful study of any candidate part's datasheet. When evaluating micropower ADCs, consider also whether they include an on-chip input amplifier, internal voltage reference, and conversion oscillator. If not, these functions may require additional external power. Some ADCs use the supply as the voltage reference, which is well suited for a ratiometric sensor like a thermistor or strain gauge. For other sensors you may have to operate the entire ADC from an external voltage reference. Some ADCs use the interface data shifting as the conversion clock, which may force the controller to waste time and power creating a slow data-clock rate. Note also that some external-clock converters require quite high frequencies, for example the AD7091R SAR ADC needs a 50 MHz clock to run at its 1 Msps maximum speed; requirements like this can have a severe impact on power consumption.[105] When considering converters for applications where you are planning to power them intermittently, keep in mind also that some converters have a non-trivial startup delay time from sleep mode.

One last consideration is the supply voltage. Most of the parts in Table 13.6 require a moderately high minimum supply voltage, such as 2.7 V. But ADCs capable of operating at lower voltages can save considerable power. For example, the AD7466 consumes 620 μW at 100 ksps when running with a 3.0 V supply, but it uses only 120 μW at 1.6 V.[106] This ADC suffers modest performance penalties at 1.6 V, but the bigger problem may be designing analog circuitry to operate on such a low voltage. Looking at the bright side, you may be rewarded with a simplified battery arrangement.

13.11 Some unusual A/D and D/A converters

Here are some interesting converter ICs, both instructive and useful, that we cannot resist displaying. These all come from Analog Devices, a traditional leader in conversion ICs and other high performance analog products. (See also the low-power ADCs in Table 13.6, and the unusual "specialty converters" in Table 13.12.)

Table 13.12 Specialty A-to-D Converters[a]

Part #	Bits	Conv Rate max (ksps)	Channels	ADCs	SAR	Integrating	Comments
AMC1203	1	10M	1	1	-	•	$\Delta\Sigma$ modulator, AC motor curr
AD7873	12	125	6	1	•	-	resistive X,Y touch screen
AD7490	12	1000	16	1	•	-	flexible sequencer
AFE5401	12	25M	4	4	•	-	automotive-radar AFE[e]
AFE5804	12	40M	8	8	•	-	8-ch ultrasound, 0.9nV/√Hz
AD6620	12	67M	2	1	•	-	FIR filter + prog RAM, to FPGA
AD9869	12	80M	1	1	•	-	transceiver, 200Msps DAC
AD6655	14	150M	2	2	-	•	IF diversity rcvr, 32-bit NCO
LMP90080	16	0.21	8	1	•	-	sensor-AFE[e], many interfaces
ADE7753	16	14[c]	2	2	-	•	AC power mon, single phase
DDC316	16	100	16	16	-	•	16 current inputs, 3 to 12pC
AD7147	16	250	13	1	-	•	13 capacitance inputs, touch
AD7609	18	200[f]	8[f]	1	•	-	simultaneous-sampling, diff'l
DDC232	20	6	32	32	-	•	32 current inputs, 3 to 12pC
78M6631	22	2.5	6	1	-	•	3-phase power, w/ 8051 CPU
AD7746	24	0.09	2	1	-	•	precise capacitance, ±4pF FS
ISL26102	24	4	2	1	-	•	quiet, 7nV/√Hz, 2ppm linearity
LDC1000	24[b]	d	2	2	•	•	inductance, loss resistance
ADS1298	24	32	8	8	-	•	standard 12-lead ECG
TPA5050	24	192	2	2	-	•	audio, lip-sync delay to 120ms

Notes: (a) wherein we find inductance, capacitance, ultrasound, ECG, powerful sensor analog front ends, and RF communication, etc. (b) 24 bits for L(inductance), 16 bits for R_P (equiv parallel resistance). (c) 14kHz RMS bandwidth. (d) fastest response = 192 LC cycles, for example 10 ksps for 2MHz. (e) sensor analog front end. (f) simultaneous sampling, 200kHz all channels.

[103] And, of course, there's also the power saved if the external sensor is power-switched.

[104] Recall that bandwidth is related inversely to pulse width, as $\mathrm{BW}\approx 1/\tau$, and that noise power grows with bandwidth ($P_\mathrm{n}=e_\mathrm{n}\cdot\mathrm{BW}$ for white noise). So a slower measurement corresponds to reduced bandwidth; it averages out the high-frequency noise.

[105] Taking the AD7091R as an example, the SCLK driver consumes $P=CV^2f$; taking C=5 pF, that amounts to 2.25 mW for a 3 V logic swing at 50 MHz, which is significantly greater than the converter's own specified power dissipation of 1 mW. Happily, the 50 MHz clocking on the SCLK pin is needed only during the 12 or 13 shifts of output data, thus reducing the average SCLK power to 0.6 mW (i.e., a factor of 13/50); much better, but still a significant contributor to the total power.

[106] The 5.2× power ratio is more than the square of the 1.9× supply-voltage ratio, revealing that the operating current drops faster than linearly versus supply voltage. This should not be a surprise, considering the effect of class-A "shoot-through" current in CMOS logic, see Figure 10.101.

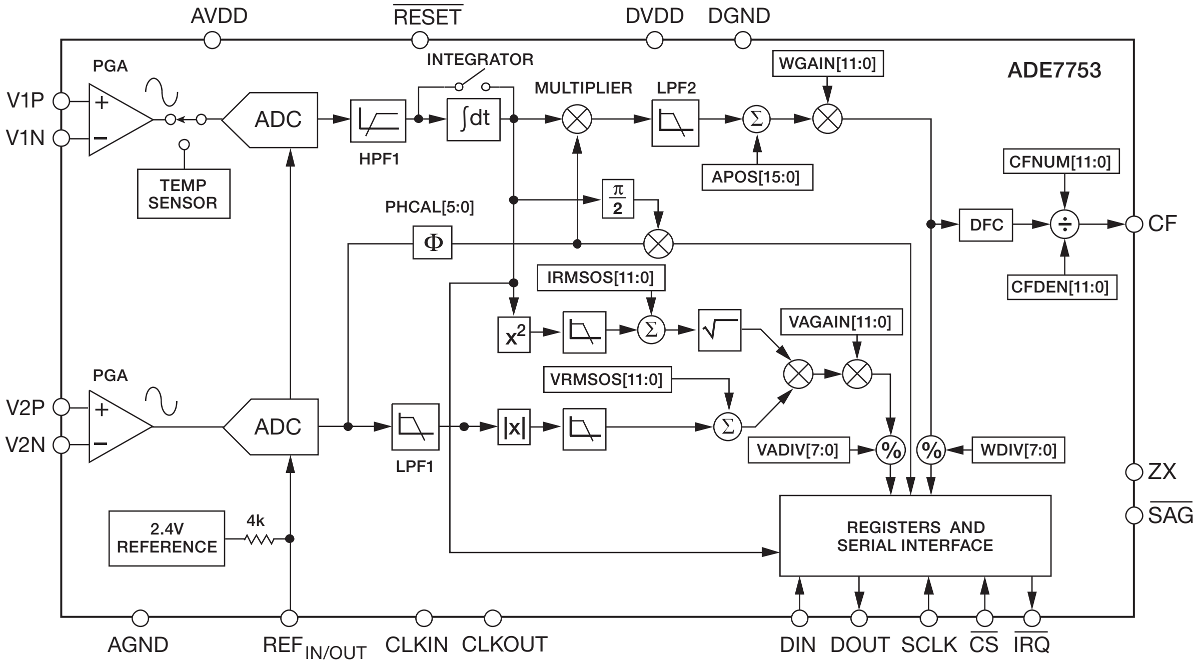


Figure 13.70. The Analog Devices ADE7753: an elegant ac power-monitor IC, courtesy of Analog Devices, Inc.

13.11.1 ADE7753 multifunction ac power metering IC

In industrial settings (and, increasingly, in the energy-aware residential context as well) it's important to keep track of electrical energy usage, in the manner of the traditional power meter with the rotating disk and the accumulated watt-hour dials. As important, or more so, is the need to monitor and minimize *reactive* power; that is, to compensate for reactive loads (such a motors) in order to keep the power factor (§1.7.6) close to unity. The power company cares about reactive power, and indeed conveys that care in the form of surcharges to industrial users, because it produces I^2R heating losses in their lines and transformers, even though it delivers no useful power to the load. It's nice also to be able to monitor the instantaneous power (both real and reactive), and, while we're at it, the presence of voltage dips (brownouts) or peaks (surges).

The elegant ADE7753 from Analog Devices[107] (Figure 13.70) is a fine example of A/D conversion, tailored specifically to this application. It normally would be paired with a microcontroller, as shown in Figure 13.71 (and in Figure 15.21). Here we'll simply admire its many thoughtful design features.

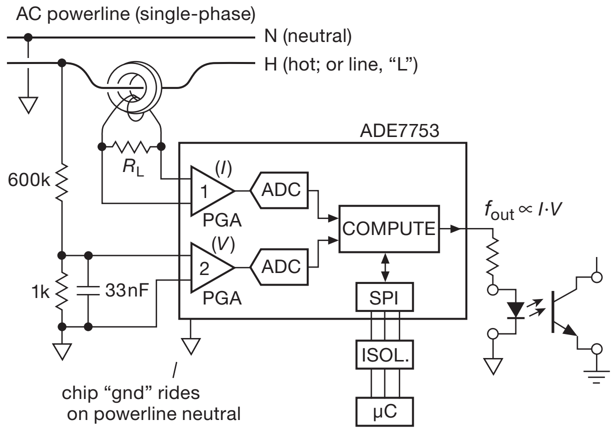
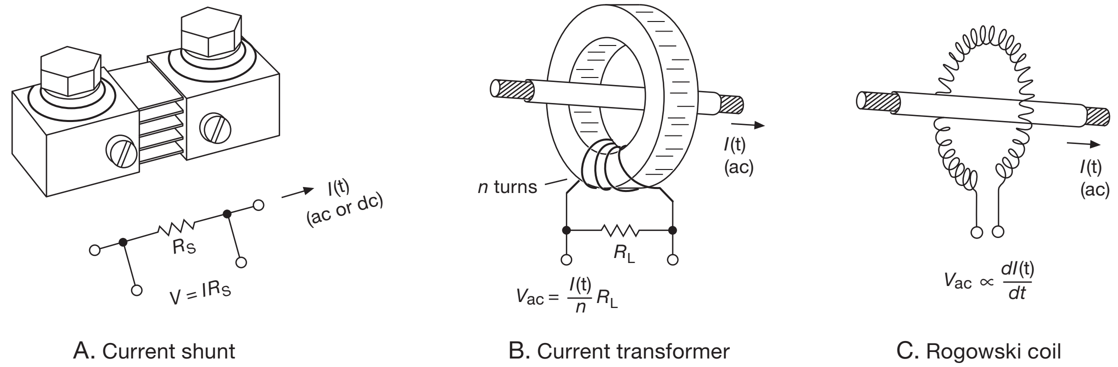


Figure 13.71. Basic powerline connection, with current transformer sensing of ac line current. The f_{out} pulse train provides a running count of energy usage, in the manner of the traditional rotating-disk power meters.

Overview From an analog input signal pair that provides a sample of line voltage and line current, this chip uses purely digital techniques (after the front-end amplifiers) to calculate continually the values of real ("active") power, reactive power, and the volt–amp product ("apparent power"). It also accumulates active and apparent power, and detects voltage sags and peaks. It is highly configurable

[107] We've chosen the single-phase part, for simplicity; the similar ADE7758 handles 3-phase power.


I(t)
(ac or dc)
R_S
$V = IR_S$
n turns
I(t)
(ac)
R_L
$V_{ac} = \frac{I(t)}{n} R_L$
I(t)
(ac)
$V_{ac} \propto \frac{dI(t)}{dt}$

A. Current shunt

B. Current transformer

C. Rogowski coil

Figure 13.72. Current-sensing techniques. The 4-wire resistive shunt (A) works with ac or dc, but provides no isolation. The current-transformer (B) and Rogowski coil (C) methods work only with ac.

by way of the simple 3-wire SPI bus (§14.7); that's how an embedded microcontroller communicates with the chip's 64 internal registers, which are used both to set up operating modes and to report measured values. It also provides a pulse-train output whose rate is proportional to active power (the rotating disk in a mechanical power meter); so, once calibrated, it can be used in stand-alone mode (no microcontroller) if all you want is a running count of accumulated energy usage.

Details, details A pair of programmable gain difference amplifiers accept the low-level (±0.5 V) signal inputs from the line voltage and current samples. There are three ways to derive the latter (Figure 13.72): (a) a small-value calibrated 4-wire in-line series resistor (a "shunt"); (b) a toroidal current transformer (as in Figure 13.71) with resistive load; or (c) a "Rogowski coil." The latter produces a voltage signal proportional to dI/dt, compared with $\propto I$ for the other two, thus requiring an additional integration. In exchange for that hassle it has the advantages of linearity (no magnetic core) and easy installation[108] (no need to disrupt power). The amplifiers have digitally trimmable offsets; their outputs are digitized with a pair of second-order 16-bit $\Delta\Sigma$ ADCs, producing the ∼28 ksps digitized stream of V and I samples.

Now comes the fun. The top path through the block diagram of Figure 13.70 is the active (real) power computation: channel 1 is the line current signal, with dc removed, feeding an optional integrator (for Rogowski); that is multiplied by the channel 2 voltage waveform, with a 0.05° phase trim (PHCAL) to ensure accurate in-phase multiplication. The result – instantaneous active power – has offset and gain trims, and then generates a proportional output frequency (at the CF pin) by way of a digital frequency converter (DFC); it also goes to the register bank, where it (and its accumulated value) can be read out.

The middle path is the reactive power computation: it's similar, but with a 90° phase shift in the current path. Finally, the bottom path is the volt–amp product (apparent power) computation, done as the product of the magnitude of voltage times the rms current, with the usual trimmable offsets and settable gain.

The block labeled "registers and serial interface" shyly conceals its considerable intelligence. It's really in charge here, with all those trims and gain settings, mode settings for things like the sag and peak detectors, the optional integrator, and the frequency scaling of the CF output. It also houses the 49-bit energy (power × time) accumulators (both real and apparent), and registers that hold data about sags and peaks. It can be configured to make an *interrupt* (§14.3.7) to the processor when bad things happen.

Overall, a pretty impressive performance for a part that costs about $4 in small quantities.

13.11.2 AD7873 touchscreen digitizer

A "touchscreen" is the familiar combination of a display device (usually a backlit color LCD), on top of which is an overlay that is sensitive to contact pressure (by fingertip or stylus). These things are used in smartphones, PDAs, tablet PCs, teller machines, point-of-sale terminals, and the like, to allow simple digital (that's a pun, get it?) manipulation of displayed objects. A simple and effective type, called a *resistive touchscreen*, consists of two thin sheets of

[108] It's even better than the figure suggests, because in practice one lead snakes back through the coil so that both leads come out at the same end. See datasheets for the RoCoil from DENT Instruments or the RopeCT from Magnelab.

transparent material, each with a conductive coating, which are pressed together by the finger's contact force.

How can you figure out where that happens? Easy: there is a metal-strip electrode along two opposing edges of each sheet; so if you apply a dc voltage across one such sheet, it acts like a resistive voltage divider, with a linear increase of voltage from one edge to the other. The touchscreen sandwich is a stacked pair, one oriented in the *x* direction, the other in the *y* direction. To read out the contact position, you first energize one sheet (say the *x* sheet) with a dc voltage, and read the voltage that the contacting point transfers to the other (*y*) undriven sheet; that gives you the *x* coordinate of the pressed location. Then you reverse the roles, energizing the *y* sheet and reading the voltage transferred to the *x* sheet.

The AD7873 (Figure 13.73) does everything you need, and more. It talks to the usual embedded processor (Chapter 15) via a 3-wire SPI serial port (§14.7 and §15.8.2), for both setup and readout. It includes internal MOSFET switches for energizing alternately the *x* and *y* sheets; an internal voltage reference; an internal temperature sensor; and a 12-bit SAR ADC with input multiplexer to select and digitize among (a) the undriven sheet, (b) the driving voltage, to make the measurement ratiometric, (c) the battery voltage, (d) the temperature, and (e) an uncommitted analog input of your choosing. This thing runs on a single supply (+2.2 V to +5.25 V), with power consumption of a few milliwatts and a price tag of about $2 in modest manufacturing quantities (1000 pcs).

An alternative touchscreen technique substitutes *capacitance* for resistance, with various schemes for determining the point of proximity. You can get complete capacitance converters, with on-chip reference, excitation, delta–sigma converter, and serial interface, for example the AD7140/50 and AD7740-series from Analog Devices. These come in single- and multiple-channel varieties, with resolutions from 16 to 24 bits. They're not fast (∼100 sps), but they're quite inexpensive (a dual 12-bit 200 sps or an 8-channel 16-bit 45 sps converter costs about $2 in quantities of 25).

13.11.3 AD7927 ADC with sequencer

Many ADCs include an on-chip analog multiplexer, so you can sample and convert a succession of analog inputs. The AD7927 (Figure 13.74) lets you do this; but it adds a programmable sequencer mode, so that you can designate a subset (actually, two subsets) of input channels to be converted in sequence, over and over. Sampling and conversion are triggered by the chip select pin, without the pipeline delays characteristic of delta–sigma converters. It uses an SPI serial port for both control/programming and for data readout; it's described in this context in §14.7.1.

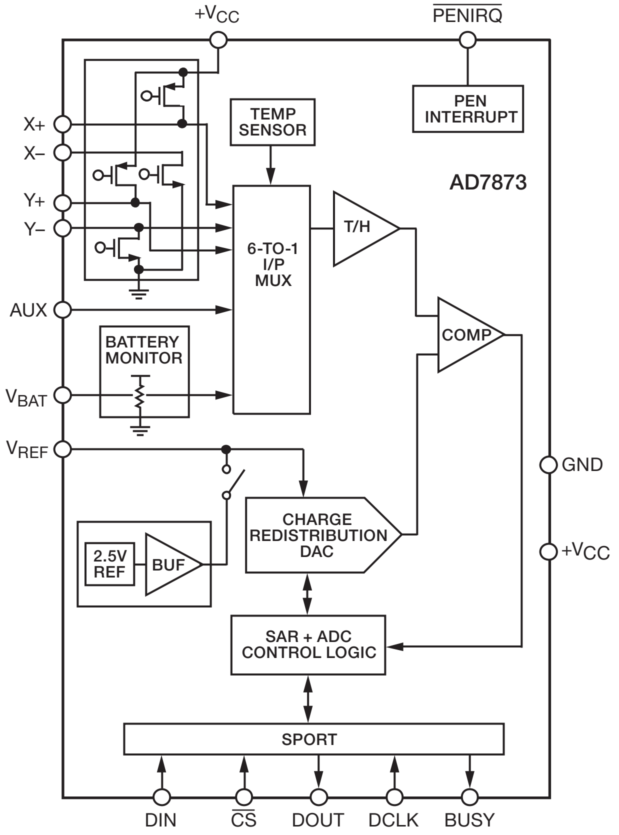


Figure 13.73. The AD7873 Resistive touchscreen digitizer. The fingertip's *x* and *y* position is determined in two phases, by energizing each plane in succession and reading the voltage-divided output from the other, courtesy of Analog Devices, Inc.

13.11.4 AD7730 precision bridge-measurement subsystem

Here's a nice chip (Figure 13.75), which takes aim at the weigh-scale market where resistive bridge transducers (strain gauges) are used. Its nicely organized datasheet makes it easy to navigate its host of clever features. It has programmable gain frontend differential amplifiers with adequate gain for 10 mV full-scale inputs, and a differential reference input for fully ratiometric measurements. It can be operated in a chopping mode, to minimize offset voltage errors and drift; and it has internal calibration modes to correct for scale errors. So you can connect the strain-gauge terminals directly to this chip, without any external preamps.

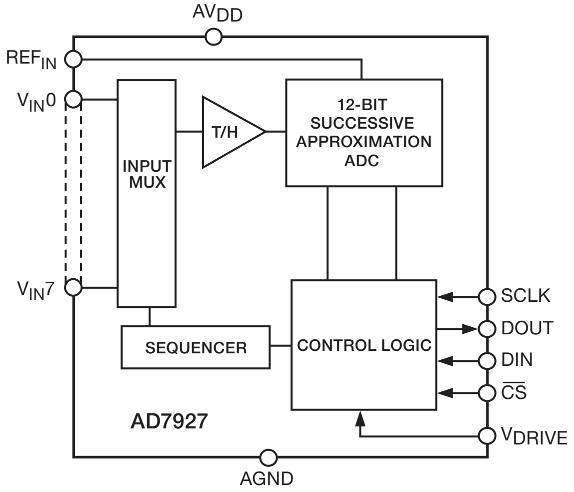


Figure 13.74. The AD7927 multiplexed successive-approximation ADC with flexibly programmed "sequencer" modes, courtesy of Analog Devices, Inc.

Of particular note is the built-in "ac excitation" signal, which you can use to reverse the polarity of bridge drive on successive measurements, thus cancelling residual offsets – including *external* offsets caused, for example, by thermoelectric voltages at the junctions of dissimilar metals. The front-page banner proclaims "Offset Drift: 5 nV/°C, Gain Drift: 2 ppm/°C." On the digital side of the fence, there's lots of programming flexibility in the control of its 24-bit delta–sigma converter and digital filter, by way of the SPI serial port (§14.7). This puppy runs on a single +5 V supply; it's available in both DIP and SMT package styles, and it costs about $15 in small quantities.

13.12 Some A/D conversion system examples

In this section we look at a few examples of complete conversion *systems*. These illustrate the kinds of design tradeoffs and subcircuit interplay that you've got to worry about when you incorporate a converter within a larger system, one that includes front-end amplifiers, voltage references, and digital interfaces.

13.12.1 Multiplexed 16-channel data-acquisition system

This application example uses a successive-approximation ADC to create a 16-channel multiplexed A/D data-acquisition system (DAQ). Figure 13.76 shows the circuit configuration, which lets you digitize any combination of 16 differential analog inputs (or 32 single-ended inputs), under flexible control of an embedded microcontroller (the latter is the subject of Chapter 15). For example, the various input channels can be configured on-the-fly as single ended or differential and with different front-end gains (set by the PGA programmable-gain amplifier), along with timestamps for each measurement; and the channels can be sampled (or skipped over) in any order, with programmable sampling intervals. A subsystem like this could form the "front end" of a microprocessor-controlled data-taking experiment, in which a quick scan of a dozen voltages is programmed to occur, with successive scans taken at intervals of 100 ms or so.

Although it looks pretty straightforward on its face, a finished circuit like this is usually the result of lots of juggling of features and compromises, as you struggle to find components with the right properties. This example was no exception. Let's take a "design walkthrough" to visit the various choices we made, and the resulting performance.

Input multiplexer Analog multiplexers are plentiful (DigiKey lists nearly a thousand); but less so in varieties that will handle a full ±10 V analog input range. And even less so in devices that permit swings beyond the ±15 V supply rails, or that do not clamp the input when unpowered. The long-lived MPC506[109] from TI (originally Burr–Brown, a leader in analog ICs) is outstanding in this regard, though at the expense of relatively high on-resistance (1.5 kΩ). Its dielectrically isolated CMOS process permits input swings 20 V beyond the rails, without "SCR latchup" or crosstalk between inputs; and its switches are "break-before-make," which means that the various input channels don't find themselves shorted together during address changes in the MUX. Watch out for considerations of this type when shopping for linear switches. They sometimes involve a compromise. For example, "break-before-make" results in a slower switching-time specification (here 0.3 μs typ) because the "make" must be delayed (by 80 ns here) to allow the switch to open.

A side note: what happens if an analog input swings more than 20 V beyond the MUX's supply rails? There will be some input current, beginning at about 15 V beyond the rails, and increasing to about 20 mA when you are 40 V beyond the rails. Beyond that you risk damaging the part. If you are expecting serious input overvoltages and want *real* protection, you can include an

[109] Or Intersil's original HI-506A, which is often listed with a break-the-search-engine part number like HI3-0506A-5Z. Ordinary ±15 V switches without beyond-the-rail capability have part numbers like DG506, HI-506, or ADG1206.

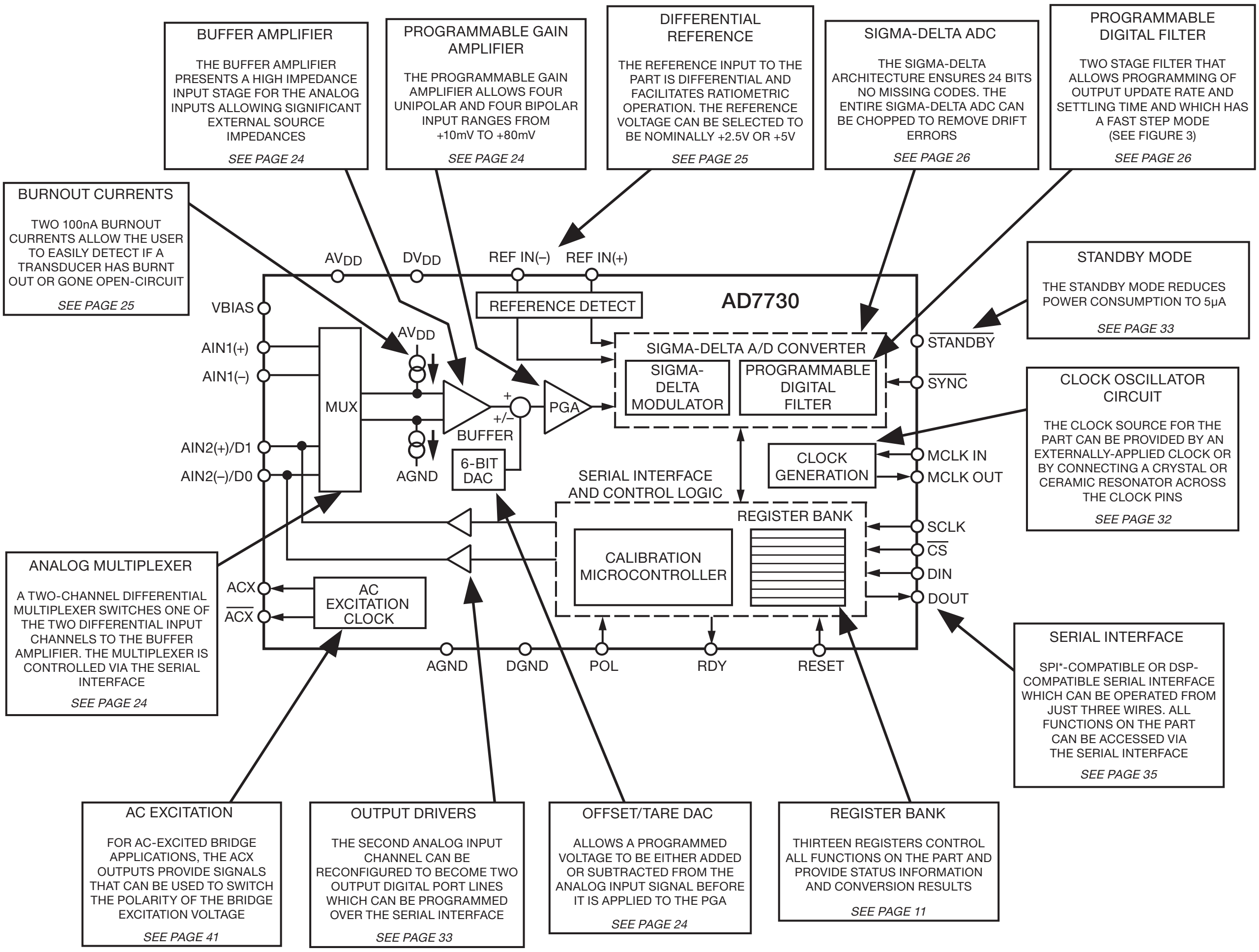

BUFFER AMPLIFIER
THE BUFFER AMPLIFIER PRESENTS A HIGH IMPEDANCE INPUT STAGE FOR THE ANALOG INPUTS ALLOWING SIGNIFICANT EXTERNAL SOURCE IMPEDANCES
SEE PAGE 24
PROGRAMMABLE GAIN AMPLIFIER
THE PROGRAMMABLE GAIN AMPLIFIER ALLOWS FOUR UNIPOLAR AND FOUR BIPOLAR INPUT RANGES FROM +10mV TO +80mV
SEE PAGE 24
DIFFERENTIAL REFERENCE
THE REFERENCE INPUT TO THE PART IS DIFFERENTIAL AND FACILITATES RATIOMETRIC OPERATION. THE REFERENCE VOLTAGE CAN BE SELECTED TO BE NOMINALLY +2.5V OR +5V
SEE PAGE 25
SIGMA-DELTA ADC
THE SIGMA-DELTA ARCHITECTURE ENSURES 24 BITS NO MISSING CODES. THE ENTIRE SIGMA-DELTA ADC CAN BE CHOPPED TO REMOVE DRIFT ERRORS
SEE PAGE 26
PROGRAMMABLE DIGITAL FILTER
TWO STAGE FILTER THAT ALLOWS PROGRAMMING OF OUTPUT UPDATE RATE AND SETTLING TIME AND WHICH HAS A FAST STEP MODE (SEE FIGURE 3)
SEE PAGE 26
BURNOUT CURRENTS
TWO 100nA BURNOUT CURRENTS ALLOW THE USER TO EASILY DETECT IF A TRANSDUCER HAS BURNT OUT OR GONE OPEN-CIRCUIT
SEE PAGE 25
STANDBY MODE
THE STANDBY MODE REDUCES POWER CONSUMPTION TO 5µA
SEE PAGE 33
AVDD
DVDD
REF IN(–)
REF IN(+)
AD7730
VBIAS
REFERENCE DETECT
AIN1(+)
AIN1(–)
AVDD
SIGMA-DELTA A/D CONVERTER
STANDBY
SIGMA-DELTA MODULATOR
PROGRAMMABLE DIGITAL FILTER
SYNC
MUX
+
+/–
PGA
BUFFER
AIN2(+)/D1
AIN2(–)/D0
AGND
6-BIT DAC
SERIAL INTERFACE AND CONTROL LOGIC
CLOCK GENERATION
MCLK IN
MCLK OUT
CLOCK OSCILLATOR CIRCUIT
THE CLOCK SOURCE FOR THE PART CAN BE PROVIDED BY AN EXTERNALLY-APPLIED CLOCK OR BY CONNECTING A CRYSTAL OR CERAMIC RESONATOR ACROSS THE CLOCK PINS
SEE PAGE 32
REGISTER BANK
SCLK
CS
DIN
DOUT
CALIBRATION MICROCONTROLLER
ANALOG MULTIPLEXER
A TWO-CHANNEL DIFFERENTIAL MULTIPLEXER SWITCHES ONE OF THE TWO DIFFERENTIAL INPUT CHANNELS TO THE BUFFER AMPLIFIER. THE MULTIPLEXER IS CONTROLLED VIA THE SERIAL INTERFACE
SEE PAGE 24
ACX
ACX
AC EXCITATION CLOCK
AGND
DGND
POL
RDY
RESET
SERIAL INTERFACE
SPI*-COMPATIBLE OR DSP-COMPATIBLE SERIAL INTERFACE WHICH CAN BE OPERATED FROM JUST THREE WIRES. ALL FUNCTIONS ON THE PART CAN BE ACCESSED VIA THE SERIAL INTERFACE
SEE PAGE 35
AC EXCITATION
FOR AC-EXCITED BRIDGE APPLICATIONS, THE ACX OUTPUTS PROVIDE SIGNALS THAT CAN BE USED TO SWITCH THE POLARITY OF THE BRIDGE EXCITATION VOLTAGE
SEE PAGE 41
OUTPUT DRIVERS
THE SECOND ANALOG INPUT CHANNEL CAN BE RECONFIGURED TO BECOME TWO OUTPUT DIGITAL PORT LINES WHICH CAN BE PROGRAMMED OVER THE SERIAL INTERFACE
SEE PAGE 33
OFFSET/TARE DAC
ALLOWS A PROGRAMMED VOLTAGE TO BE EITHER ADDED OR SUBTRACTED FROM THE ANALOG INPUT SIGNAL BEFORE IT IS APPLIED TO THE PGA
SEE PAGE 24
REGISTER BANK
THIRTEEN REGISTERS CONTROL ALL FUNCTIONS ON THE PART AND PROVIDE STATUS INFORMATION AND CONVERSION RESULTS
SEE PAGE 11

Figure 13.75. Frontend analog subsystem tailored for bridge-type transducers such as strain gauges, weigh scales, and pressure transducers. The AD7730 datasheet, from which this figure is taken, is a model of clarity, and a delight to read, courtesy of Analog Devices, Inc.

input current-limit circuit like the one in Figure 13.77: the back-to-back depletion-mode MOSFETs, in convenient TO-92 or SOT-23 packages, can hold off 500 V, and limit the current to ∼1 mA.[110] See §5.15.5 and Figure 5.81 and associated discussion for more detail.

Differential/single-ended selector switches Dual SPDT analog switches, under digital control, route the multiplexers' outputs: in 16-channel differential mode, S_1 and S_2 always look at U_1 and U_2; in 32-channel single-ended (SE) mode S_2 looks at signal common, while S_1 looks at U_1 or U_2 for channels 1–16 or 17–32, respectively. (This is often called a *pseudo-differential* input, because the common terminal is shared by all the SE channels.) Mode and channel switching can be done on a channel-by-channel basis (explained below). Note the absence of an anti-alias filter, which would limit the multiplexing speed: we assume the input signals are appropriately band limited upstream of the multiplexers.

The IH5043[111] SPDT analog switches S_1 and S_2 were chosen for their low leakage, low capacitance, and low charge injection. These come at the expense of relatively high R_{on} (80 Ω max). (You can get plenty of analog

[110] Under normal conditions the series resistance is $2R_{\text{on}}+R_{\text{s}}$, or about 2.7kΩ. The added 1 kΩ of series resistance is not much of a compromise; you could omit the resistor, and the saturation $I_{\text{DSS}} \approx 2$ mA would still protect the MUX, but then the transistors' dissipation limit means that you'd need to limit sustained input overvoltages to 100 V or so.

[111] Alternative part numbers are HI5043 and DG403.

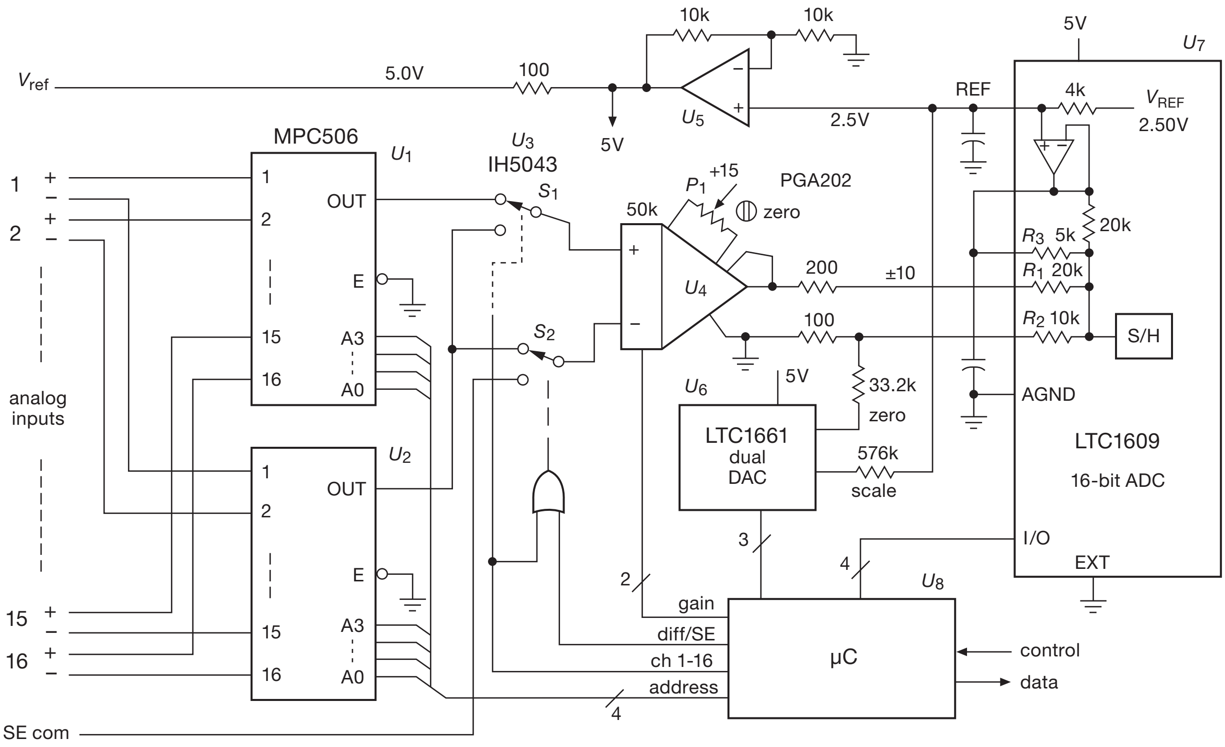


Figure 13.76. A 16-bit 16-channel (differential) or 32-channel (SE) successive-approximation ADC system. The analog ICs in the signal path (U_1–U_5) are powered from ±15 V; the other ICs run from a single +5 V supply.

switches with very low R_{on}, down to 0.5 Ω or less; but you don't need that here, and you'd pay the price in leakage, capacitance, and charge injection.[112])

Instrumentation amplifier Ideally we want an instrumentation amplifier (see §5.15 and Table 5.9) with digitally programmable gain (a PGA), that can accommodate the full ±10 V analog signal range, and that has fast settling time, stable gain, low offset voltage, low noise, and low bias current. It's fine to tick off a laundry list of desirable characteristics – but just how good does each of these need to be? What matters, ultimately, is that the amplifier's limitations do not degrade the overall system's performance.

Given the surrounding system, the PGA202 from TI (Burr–Brown) is a good choice here: it has programmable gains of 1, 10, 100, and 1000 (set by a pair of logic-level input pins), with a settling time (to 0.01%) of 2 *μ*s (for all but G_V=1000), adequate for the downstream ADC's 200 ksps conversion rate. The three lowest gain settings correspond to full-scale input ranges of ±10 V, ±1 V, and ±0.1 V. Its high input impedance and low input current (10 GΩ, 50 pA) do not degrade the combined characteristics of the upstream multiplexer and switch (the MUX specifies a typical leakage current of 2 nA).

Finally, what about amplifier offset voltage and noise? We chose a JFET-input instrumentation amplifier for its high-*Z* input; but, looking at comparable amplifiers in Table 5.9, we see that we've paid a price, in terms of offset voltage and noise, compared with the bipolar-input PGA204. We need to compare these effects with the resolution (LSB step size) of the downstream ADC, which looks at the amplifier's output. But amplifier offset and noise is always specified at the *input* ("RTI," referred to the input); so we need to figure out the converter's RTI step size, which depends on the amplifier's gain. That's easy enough: the ±10 V converter input range, divided by its 2^{16} steps, amounts to an LSB

[112] For example, the ADG884 analog switch has a very low $R_{\text{on}} = 0.4\,\Omega$ (max); but its shunt capacitance C_S(on) is 295 pF, compared with 22 pF for our chosen IH5043. It's also a low-voltage part, with a maximum analog swing of 5 Vpp. The ADG1413 operates over the full ±15 V range, with a low R_{on} of just 1.5 Ω, but its charge injection (±300 pC) is five to ten times greater than that of the '5043/DG403.

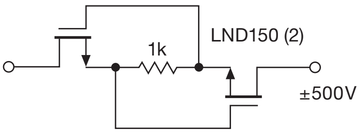


Figure 13.77. Input current-limiter circuit, good to ±500 V overdrive.

step size of 0.3 mV. So the step size, referred to the amplifier's input, is 300 μV, 30 μV, and 3 μV, for gains of 1, 10, and 100, respectively.

We're seriously challenged here, because the amplifier specifies a typical RTI offset voltage of (0.5+5/G) mV; that is, 0.5 mV in its frontend amplifier, combined with 5 mV in its internal output stage. Without some additional tricks we're facing typical offset errors of 5.5 mV, 1 mV, and 0.55 mV at gains of 1, 10, and 100; that's 18, 33, and 180 times the converter's RTI step sizes, respectively. Clearly we need to manually trim its offset, and also include some electronic nulling circuitry;[113] and, even with that, we'll be limited ultimately by the amplifier's tempco and drift.

In our design we've got the recommended manual trim (set once, at maximum gain), and a 10-bit DAC to zero the offset; the latter has a 0–5 V output, for a trim range of ±7.5 mV at the amplifier's output ("RTO"). Its 10-bit resolution gives a 15 μV step size, far better than needed, given the converter's 300 μV LSB step size.

So, to use this thing for full-accuracy measurements, we zero the offset error (via the ADC) at the beginning of a suite of measurements, and we hope that short-term offset drifts are small. Are we on solid ground? Well, the amplifier's specified typical offset drift is (3+50/G) μV/°C, 50 μV/month, and (10+250/G) μV/V(supply). We use regulated supplies, and we are concerned with short-term drifts only; so we're generally OK, with thermal drifts a potential problem only at the highest gain (G=100), where a 1°C change causes a 1-LSB error. There's also drift of amplifier *gain* with temperature to worry about. Here that's 3 ppm/°C (typ) for G=1 or 10, and 40 ppm/°C for G=100. An LSB corresponds to 30 ppm at ±full scale, so once again we're in good shape except at the highest gain (G=100), where the amplifier's gain tempco causes a 1-LSB error for a 1°C temperature change.

Finally, what about the amplifier's voltage noise? The RTI specified values are 1.7 μVpp (typ) for the $1/f$-plagued low-end band of 0.1–10 Hz, and a density e_n=12 nV/$\sqrt{\text{Hz}}$ (typ) at 10 kHz (the $1/f$ corner is roughly 100 Hz). So a band extending from 0.1 Hz to ~10 kHz has an RTI noise voltage of about 3 μV, comparable to the RTI step size at G=100; it's negligible at lower gains.

Analog-to-digital converter We need a converter to handle the full ±10 V signal range, with enough speed for scanning at rates of 100 kHz or more. (There are scanning ADCs, with input multiplexers and sequencer logic, such as the 8-channel 16-bit 500 ksps AD7699, but they force you to live with their limited input range, e.g., 0–5 V or ±2.5 V, rather than the ±10 V as we have here.) The LTC1609 does the job, and runs on a single +5 V supply, which we can regulate and filter individually to keep it quiet. It converts up to 200 ksps, with 2 μs acquisition (input-settling) time, and a serial interface with multiple modes (internally or externally clocked serial data, etc.; see §15.9.2 for interfacing and programming details). Its worst-case zero-offset (±10 mV) and gain (±1.5%) errors have to be calibrated out (done here with the dual DAC, U_6), after which it has acceptably low offset and gain drifts of ±2 ppm/°C and ±7 ppm/°C, typ, respectively. (Recall that an LSB at full-scale is 30 ppm.)

Voltage reference Much of the gain error and drift is due to the internal +2.5 V reference, with its ±1% (worst case) accuracy and ±5 ppm/°C typical tempco. This is in fact a very good drift specification for an internal-reference ADC. But if you want better performance, use a precision external reference (see Tables 9.7 and 9.8), the best of which have ±0.02% (worst case) accuracy and ±1 ppm/°C or better typical tempcos. The LTC1609's REF pin invites an external reference, which simply overdrives the 4 kΩ internal source. With such an external voltage reference the untrimmed gain error is reduced to ±0.5% (which we can trim out with the DAC), and the gain drift is reduced to ±2 ppm/°C (typ). These gain drifts are comparable to those of the upstream amplifier, except at the latter's highest gain (G=100), where the amplifier's drift is an order of magnitude larger and becomes problematical.

Programming and operation With systems that include microcontrollers, there's still plenty of work to do! Here you'd run a calibration setup, storing the ADC trims in non-volatile microcontroller memory. Then at startup you would program the DAC as well as ADC ranges and communication modes. Look first at §15.9.2 and the

[113] Or software offset subtraction: we could devote one (shorted) channel to zero-error measurement. And we could use another to measure the voltage reference for full-scale calibration. Zero-offset error corrections are often stored as parameters that have been measured during calibration.

timing diagram in Figure 15.23 to see how a single ADC conversion is handled. But there's much more involved in controlling a full data-acquisition system like this: you need to set up in advance details such as the active input channels and their sequence, and for each channel whether it's single ended or differential, its gain, etc. This would typically be done with a lookup table, accessed at run time. You also need to specify scan rate, interrupt modes, what data are to be stored at run time (channel number, mode, gain, time stamp, and so on), along with header information that goes into the data file (such as machine ID, experiment type, date, operator, sensor configuration, and so on). We could ramble… but you get the picture: there's plenty of follow-on organization and programming to make it all fly. For these tasks we've found that it's nice to have dedicated graduate students, who have lots of time, lots of skills, and whose degree depends on making it all work.

13.12.2 Parallel multichannel successive-approximation data-acquisition system

The previous example is *multiplexed*, with a single ADC digitizing the input channels successively in some programmed sequence. That's OK in many applications; but sometimes it's important to capture *simultaneously* the input levels on multiple analog inputs. One way to do this is to capture each analog input on its own sample-and-hold (or track-and-hold), then multiplex these stable analog voltages into a single ADC. But ADCs are inexpensive, so it's often better (and always faster) to use a set of them to digitize the inputs simultaneously. In this example we illustrate this with successive approximation ADCs, and in §13.12.3 we'll do it with delta–sigma converters.

Figure 13.78 shows one implementation, using a set of chips from Analog Devices that play well together in this application. Let's walk through it.

Input amplifier Problem: you want to accept bipolarity input signals, say over a ±10 V range – but you've got an ADC running from a single positive supply, which accepts positive signals only. Solution: use an op-amp input stage to offset the signal and reduce its swing. You can do this, but you'll need matched resistors of high precision in order not to compromise the accuracy of the ADCs.

Here we've taken advantage of the elegant AD8275 level-translating ADC driver, which does what you want: it's basically a single-supply difference amplifier with $G = 0.2$ and with an input offset terminal. As here configured it converts a bipolarity input range (±10.24 V) to a positive-only output centered on half the reference voltage, i.e., 0–4.096 V. It's got plenty of bandwidth (0.45 μs settling to 0.001%, which is less than an LSB), rail-to-rail output (so it can run from the same +5 V as the converter, protecting against ADC overdrive), accurate and stable gain ($G = 0.2 \pm 0.024\%$, tempco 1 ppm/°C max), and acceptably low offset and drift ($V_{\text{os}} < 0.5$ mV max, tempco 7 μV/°C max).

Two important points. (a) The offset and drift specifications are *referred to the output* (RTO), not to the input. In other words, the input offset is five times larger, or ±2.5 mV (max), and likewise for the drift. This amount of offset is not insignificant: the 16-bit converter's LSB, referred to the amplifier's input (RTI), corresponds to $2 \times 10.24\,\text{V}/2^{16}$, or 0.31 mV; so the amplifier's worst-case offset is about 8 LSBs. The drift, by comparison, *is* insignificant: it would take a 45°C change of temperature to move by 1 LSB. So the channels should be calibrated at zero input (and also at full-scale input). (b) The unround reference voltage (4.096 V) is popular because it produces a round-number conversion gain, namely precise 10.0 mV steps at the 11-bit level (which you can think of as further subdivided by 2^5 at the full 16-bit resolution). Also, because the full-scale range extends to ±10.24 V, you can calibrate the system using a 10.0 V reference without driving it over-range.

ADC The AD7685 is a single-supply 16-bit successive-approximation converter with an SPI serial output fast enough to handle its maximum 250 ksps conversion rate. The SPI interface permits daisy chaining with additional converters, as shown; and the serial interface allows packaging in a small 10-pin package. It has good linearity and accuracy properties: no missing codes, ±3 LSB (max) integral nonlinearity, ±0.3 ppm/°C (typ) gain drift.[114]

Reference The ADC requires an external reference, which sets the positive full-scale range. The ADR440 series of "XFET" voltage references (§9.10.3) exploit the pinch-off voltage of a pair of JFETs, in a clever configuration that achieves low noise and low drift: 1.8 μVpp (typ) and 3 ppm/°C (max).

Serial port isolator Quiet systems get noisy if you allow digital hash and ground currents into the analog

[114] This is a charge-redistribution ADC, for which it is sometimes advisable to put a capacitor across the analog input, isolated from the driving amplifier with a small resistor, as in Figure 13.37; a suitable choice here would be 2.7 nF and 33 Ω.

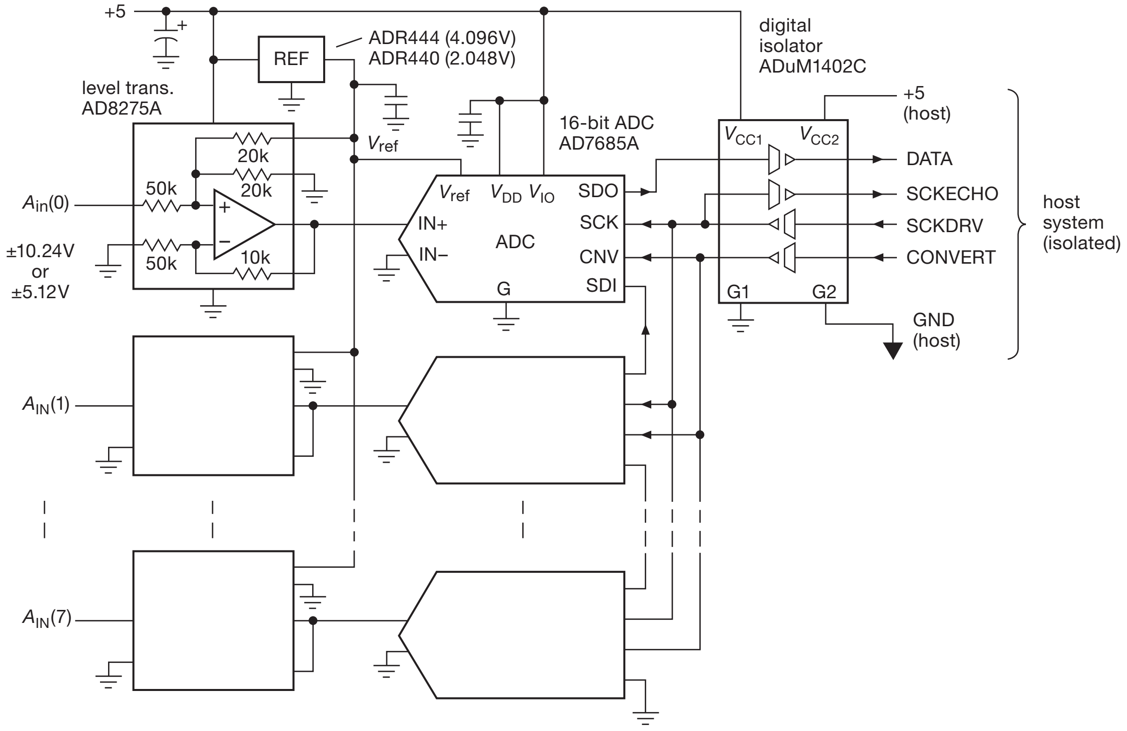


Figure 13.78. Multichannel parallel successive-approximation ADC with isolated SPI serial output port.

portion. With a 3-wire interface it's easy to add full galvanic isolation, in this case with the ADuM1402C 4-channel isolator. This puppy uses on-chip transformer coupling, and is good to 90 Mbps. Note the readback of the SPI clocking signal: that's needed because the signal delay through the isolator (27 ns typ) is comparable to the period of the maximum readout clock rate (∼50 Mbps); by echoing back the SCK received by the ADC, the host system sees the output data accurately synchronized to that echoed clock. This is inaccurate only to the extent that the isolator delay time varies between channels (skew), which for this isolator is an impressive 2 ns (max).

Component cost The amplifier and ADC blocks, which are replicated for each channel, are not expensive: about $3 and $10, respectively (in quantities of 25). Adding $5 for the reference and $9 for the isolator, we have about $118 for an 8-channel system.

A. An integrated parallel multichannel SAR solution

Why build it, when you can buy it? As it happens, the clever folks at Maxim have integrated a simultaneous 8-channel 16-bit successive-approximation ADC system on a chip, with performance comparable to that in the previous section: 250 ksps, single +5 V supply, bipolarity conversion range (±5 V), and good accuracy and linearity (±0.01% maximum offset, ±2.4 μV/°C typical offset tempco, no missing codes, ±2 LSB maximum integral nonlinearity). The MAX11046 uses a parallel data interface, with a separate digital supply pin for compatibility with a low-voltage microcontroller. Figure 13.79 shows the scheme.

This part is unusual in accommodating a bipolarity input voltage range while running from a single positive supply.[115] A single CONV pulse starts the conversion, simultaneous on all channels; the signals are sampled at the rising edge of CONV, with a typical timing skew of 0.1 ns (!). The conversions are complete 3 μs later, and are read out as 16-bit parallel words, one channel per RD′ pulse, as shown.

Figure 13.80 shows in more detail what's included on-chip. The analog inputs are clamped at about 0.3 V beyond the conversion range (i.e., ±5.3 V), but you have to include current-limiting series resistors to limit the clamp current to 20 mA. A set of fast (BW=4 MHz) track-and-hold

[115] The datasheet does not say how this is done, but it's most likely an on-chip charge-pump negative supply generator. The very high input impedance rules out something like the previous voltage-translation scheme.

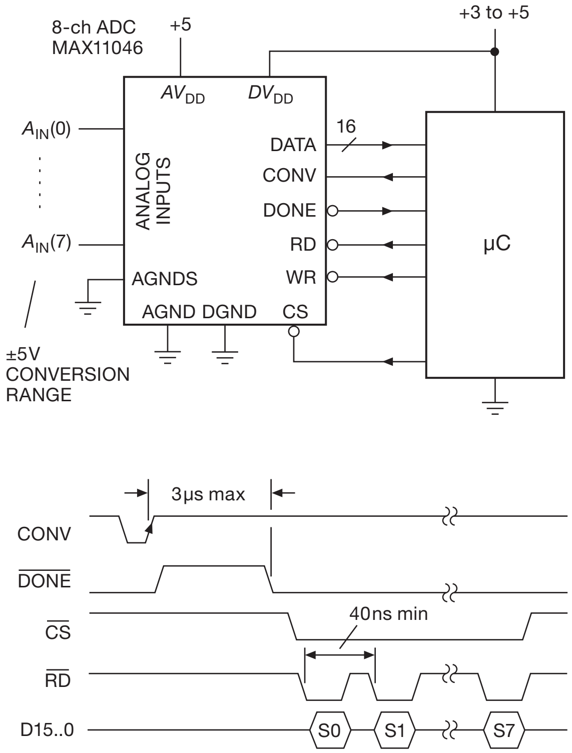


Figure 13.79. The MAX11046 integrates an 8-channel parallel successive-approximation ADC onto a single chip.

circuits capture the input signals, followed by the array of latched ADCs and the output multiplexer. The digital port allows some configuration, via the four low-order bidirectional data lines: internal or external reference, offset binary or 2's complement, and single or continuous conversion mode. You could think about adding galvanic isolation to the digital lines – but you'd have to use 21 channels of isolation, and you'd have to arrange for bidirectional signaling on the four low-order data lines D3..D0 (with the direction set by WR′). Ah, the elegance of serial communication!

Compared with our à la carte design of the previous section, this system is a bargain: about $42 (single-piece price) for the complete 8-channel converter system. As one might expect in the competitive world of silicon, Maxim is not alone in integrating a simultaneous-sampling multichannel ADC. The AD7608 from Analog Devices provides eight channels of T/H (spanning a full ±10 V range) multiplexed into an 18-bit ADC capable of 200 ksps on all channels, with an on-chip digital filter and both parallel and serial output formats. Note the different approach: this latter part uses a single fast ADC to convert the levels captured on the eight T/Hs, whereas the Maxim part uses eight ADCs.

13.12.3 Parallel multichannel delta–sigma data-acquisition system

Here's another example of a multichannel simultaneous-sampling system, this time exploiting the advantages of delta–sigma conversion: high accuracy, low cost, and greatly relaxed requirements for the anti-alias filter (whose rolloff is set by the much higher *over*sampling frequency). Delta–sigma ADCs with integral PGA and serial output (I^2C or SPI) are available in low-pincount packages like the SOT23-6, with prices of just a few dollars, and with conversion resolutions of 16 to 22 bits. A non-multiplexed multichannel data acquisition system is easily assembled by corralling a bunch of these elegant ICs (Figures 13.81 and 13.82), in this case taking aim at a relatively slow but accurate "voltmeter"-like system.

Circuit design involves the fine art of compromise, trading off the various benefits and drawbacks involved in component selection, in the choice of circuit topology and complexity, and in their impact on system cost. The choices we faced, as we worked out this example, illustrate the process nicely. Let's take it in stages.

A. First try

We began with the notion of a bused I^2C array of ADCs (Figure 13.81), which minimizes the number of microcontroller pins needed (compared with an SPI interface), because there is no need for individual chip-select (CS′) lines. TI's ADS1100 converter ($4.50 in unit quantity) looked good: it describes itself as a "self-calibrating, 16-bit ADC" that is a "complete data acquisition system in a tiny SOT23-6 package." It contains a differential input PGA (gains of 1, 2, 4, or 8), runs on a single supply from +2.7 V to +5.5 V (at 90 μA), includes an internal clock, and guarantees 16-bit conversions with no missing codes at its lowest conversion rate of 8 samples/s. It has a maximum integral nonlinearity (INL) of 0.013%, and a typical gain error of 0.01% (the positive supply V_{DD} is the reference, with full scale of $\pm V_{DD}/G$).

That's the good news. Here's the bad news: the I^2C protocol (§14.7.2) requires each bused device to have a unique address (of 128 possible); that's the price you pay for a 2-wire bus with no individual chip-select lines. This is usually accomplished by dedicating a few device pins to set the address (e.g., three pins to select 1 of 8 addresses within a subset of the full 128 possible addresses). But a part with only six pins cannot afford this luxury: count them – differential input (2 pins), power and ground (2 pins), I^2C bus (2 pins) – there are no pins left!

The ADS1100 solves this problem by pre-assigning the addresses, with a different part number for each of eight

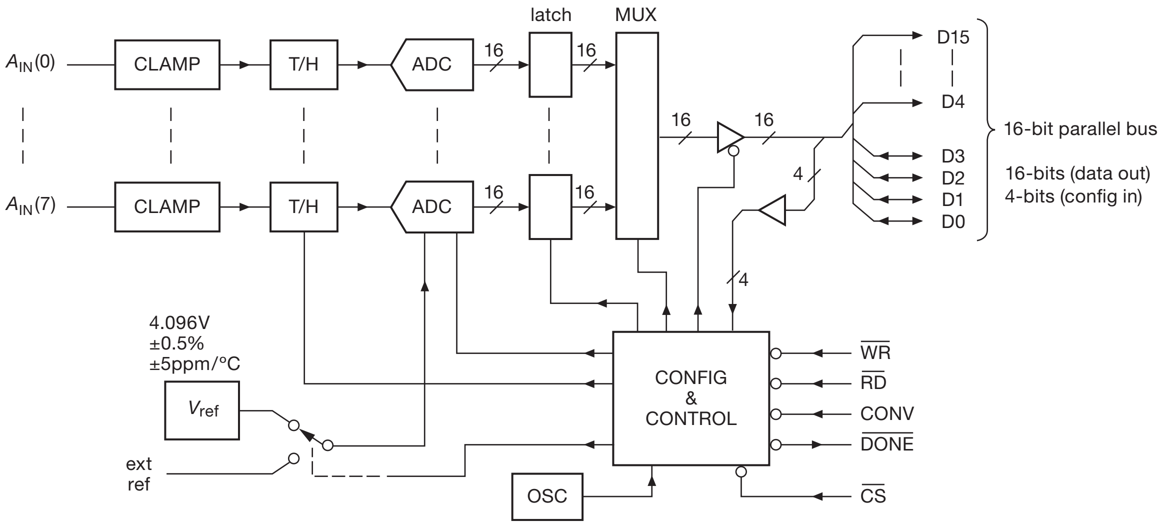


Figure 13.80. Block diagram of the MAX11046 innards.

possible addresses (decimal 72 through 79). For an eight-channel system, then, you've got to order eight different parts (if you can find them in stock: at the moment of writing, DigiKey has addresses 72 and 74; Mouser has addresses 75 through 79; and Newark has addresses 72 through 74), and with less opportunity for quantity pricing.

There's one more piece of bad news: the ADS1100 has an internal clock of coarse accuracy (±20%), with no option for an external clock (no pins remaining!). So you cannot achieve the high rejection of powerline frequencies (60 Hz or 50 Hz) that comes with a sampling rate that is an integral number of powerline cycles (e.g., exactly 10 conversions/s). You wind up with a mediocre ∼30 dB rejection of normal-mode signal at 50 Hz or 60 Hz.

B. Second try

The ADS1115 is a related part from the same manufacturer, priced around \$5 (in quantities of 25), this time housed in a 10-pin package, with one of the additional pins allowing some degree of address selection. This is done cleverly: a single pin sets one of *four* addresses (decimal 72–75), according to whether it is tied HIGH, tied LOW, or connected to one of the two I^2C serial-interface lines (Figure 13.81). This is an improvement over the 6-pin ADS1100, but with only four possible I^2C addresses you'd have to use a second I^2C channel to get eight input channels.

There's a second input channel, which could be used for multiplexed conversion. But we want simultaneous conversions on all channels, so we've used the second channel for zero calibration, as shown, which you'd do between active conversions. Some other nice features of this chip are its operation from +2.0–5.5 V, a wide range of PGA gains (0.66×, and 1× to 16×, by factors of two), a wide range of conversion rates (8–860 sps at full 16-bit resolution), and an internal clock and voltage reference.

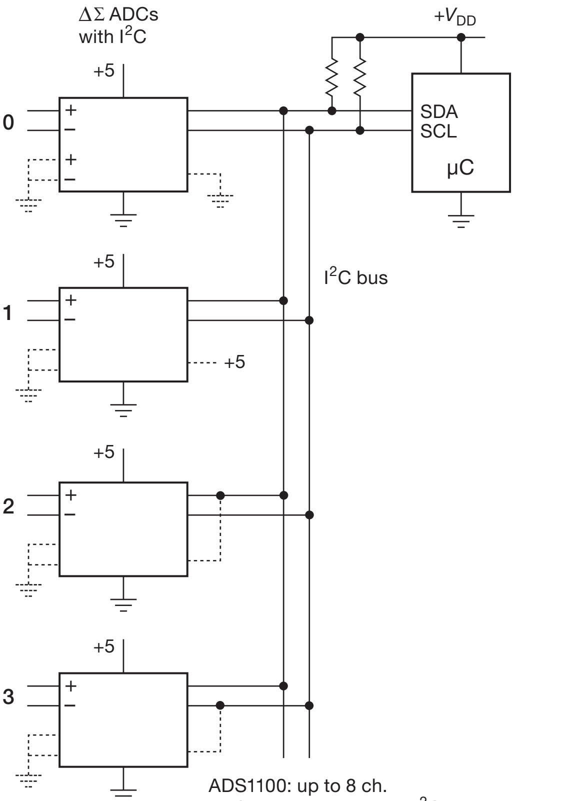
So far, so good. But there's trouble still in paradise. As with the ADS1100, the internal clock is of only coarse accuracy (±10%), so you get only some 30 dB of powerline rejection.[116] Likewise, the internal voltage reference is not of great precision, and there is no option for an external reference: this is specified as *gain error*, which for this part is 0.01% (typ), 0.15% (max). To give it perspective, an LSB (at 16-bit resolution) is 0.0015% (15ppm); so the worst-case full-scale error corresponds to 100 LSB steps. Furthermore the gain drift (tempco) is specified as 40 ppm/°C (max), which corresponds to 3LSBs/°C.

C. Third try

Not satisfied with these choices, we next explored converters with an SPI digital interface. Instead of I^2C's bus addresses, you need to provide a separate chip-select line to each converter (Figure 13.82). That's the bad news. The good news is that there are some terrific converters out there.

We looked at many candidates; the first to pass muster

[116] You need to read the datasheet carefully, here: it lists 105 dB typical *common-mode* rejection at 50 Hz and 60 Hz, but no listed value for *normal-mode* (i.e., differential) powerline signals. Later there's a graph that shows the ∼30 dB value.



ADS1100: up to 8 ch.
ADS1115: need another I²C pair for 8 ch.

Figure 13.81. Multichannel parallel ΔΣ ADC with I²C output port. The dashed lines show connections for the ADS1115 converter only.

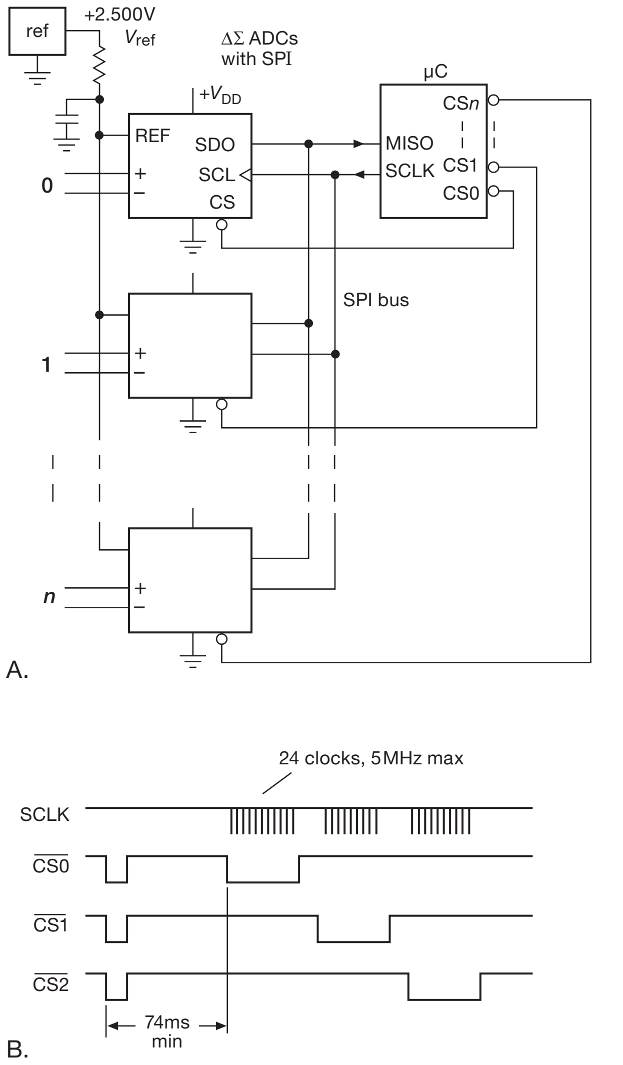


Figure 13.82. Multichannel parallel ΔΣ ADC with SPI output port (e.g., a CS5512 or an MPC3551). The timing diagram applies to the MCP3551 family, for which the first assertion of CS′ initiates conversion (simultaneous across all channels), with subsequent serial data readout (of the individual channels) clocked by SCLK during reassertion of CS′.

was the CS5512 from Cirrus Logic. This is a single-supply (+5 V) 20-bit delta–sigma converter in an 8-pin SOIC, priced around $4.25 (in quantities of 25). An external clock source (32.768 kHz nominal) times the conversions, and also the SPI clocking. The accurate clock frequency is exploited for true *normal-mode* powerline rejection: the chip's digital filter is configured with a broad 80 dB (minimum) notch extending from 47 Hz to 63 Hz, with ∼90 dB simultaneous rejection at both 50 Hz and 60 Hz.

This chip also excels in linearity (±0.0015% of full scale, maximum), and delivers 20-bit resolution with no missing codes. And its typical offset voltage and gain drifts are 0.06 μV/°C and 1 ppm/°C, respectively. This is one classy converter chip!

Now for the thunderclap: this chip requires an external voltage reference, which is normally a good feature (because you can use a high-quality reference, whose voltage also sets the analog scale). Imagine our surprise, then, when we read the datasheet's specification of the full-scale analog input range: V_{FS}=0.8V_{REF}, ±10%. In other words, this is a highly linear and stable converter, but with a conversion gain that is uncertain to ±10%.

D. Fourth try

From Microchip (a company traditionally known for its microcontrollers) comes the winning candidate. Their MCP3551 is a 22-bit delta–sigma converter in an 8-pin SOIC (or smaller MSOP), priced around $3.25 (in quantities of 25), with an accurate (±0.5%) internal clock that delivers excellent powerline rejection. It runs from a single +2.7 V to +5.5 V supply, drawing about 0.1 mA. It requires an external voltage reference, which (unlike the previous converter) accurately sets the analog scale factor; and it allows 12% over- and under-range inputs, signaled with two additional data bits. And it performs single-cycle conversions with no digital-filter settling time, during which it also carries out an offset and gain calibration.[117] Figure 13.82's timing diagram shows the scheme for simultaneous multichannel conversion, followed by sequential data readout.

The specifications are impressive: 22-bit resolution with no missing codes; V_{os} of $\pm 12\,\mu$V (max), full-scale error of ±10 ppm (max), INL of 6 ppm (max), and normal-mode powerline rejection of 85 dB (typ) at both 50 Hz and 60 Hz (Figure 13.83).[118] The typical offset and gain drifts are 0.04 ppm/°C and 0.028 ppm/°C, respectively. What's not to like in this converter? The only thing we can really complain about is the absence of a PGA: lacking that, it takes the converter's full 22-bit capability to achieve 1.2 μV resolution, the same that you would get with a 16-bit converter having a built-in PGA with 64× gain.

Totaling up the cost for an 8-channel delta–sigma simultaneous-conversion system, we have $34 for the ADCs and about $4 for an ADR441A reference, or a very economical $38.

E. An integrated parallel multichannel $\Delta\Sigma$ solution

Never underestimate what can be done on a single piece of silicon. Here again the wizards of Silicon Valley have come through, with several interesting multichannel ADCs featuring simultaneous conversion on all channels.

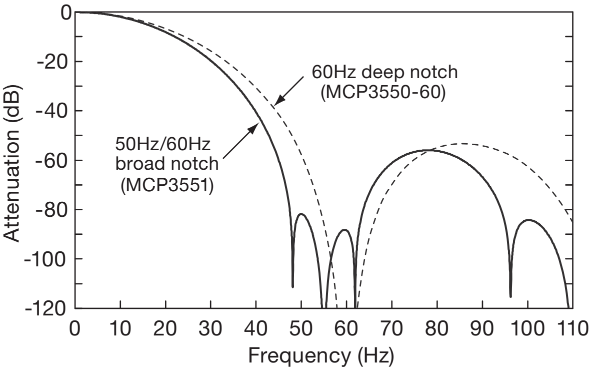


Figure 13.83. The MCP3551's digital filter is configured to create a broad notch covering 50 Hz and 60 Hz powerline frequencies, and a bit more. By contrast, the MCP3550-60 (or -50) is configured for a single deep notch.

For relatively low-speed conversion there's the AD73360, which houses six 16-bit 64 ksps delta–sigma converters, each with its own programmable-gain amplifier (0–38 dB). It comes in a convenient 28-pin SOIC and costs about $8 (in quantities of 25). It has a programmable sampling clock and a serial-output port optimized for transferring data automatically into a downstream DSP chip (from as many as eight cascaded converters). Its six channels are ideal for voltage and current measurements in a 3-phase motor drive, or for industrial power monitoring. It requires in-system calibration (±10% gain accuracy), and is usually used with ac coupling (worst-case dc offset ~10% of full scale).

What about *really* fast multichannel delta–sigma conversion? Our design runs at a leisurely conversion rate of just 15 samples per second (the AD73360 betters that by three orders of magnitude, but with lower resolution and degraded accuracy). The analogous task becomes extraordinarily difficult if you were to increase this by a factor of a million or so. Difficult, but evidently not impossible: the impressive ADC12EU050 from National Semiconductor (Figure 13.84) packs eight simultaneous 12-bit differential-input delta–sigma ADCs that run at 50 Msps (!). That creates a firehose of digital output, for which it devotes an LVDS pair to each output channel; it consumes about 0.4 W at full tilt, and costs about $100.

13.13 Phase-locked loops

13.13.1 Introduction to phase-locked loops

The phase-locked loop (PLL) is an interesting and useful building block, available both as a single stand-alone

[117] In the words of the datasheet, "A self-calibration of offset and gain occurs at the onset of every conversion. The conversion data available at the output of the device is always calibrated for offset and gain through this process. This offset and gain auto-calibration is performed internally and has no impact on the speed of the converter since the offset and gain errors are calibrated in real-time during the conversion. The real-time offset and gain calibration schemes do not affect the conversion process."

[118] How do they achieve simultaneous rejection at both frequencies? According to the datasheet, "The digital decimation SINC filter has been modified in order to offer staggered zeros in its transfer function. This modification is intended to widen the main notch in order to be less sensitive to oscillator deviation or line frequency drift. The MCP3551 filter has staggered zeros spread in order to reject both 50 Hz and 60 Hz line frequencies simultaneously." For maximum rejection at a *single* powerline frequency, use the MCP3550-50 or -60, which offer 120 dB (typ) normal-mode rejection at the corresponding frequency, once again with the internal clock.

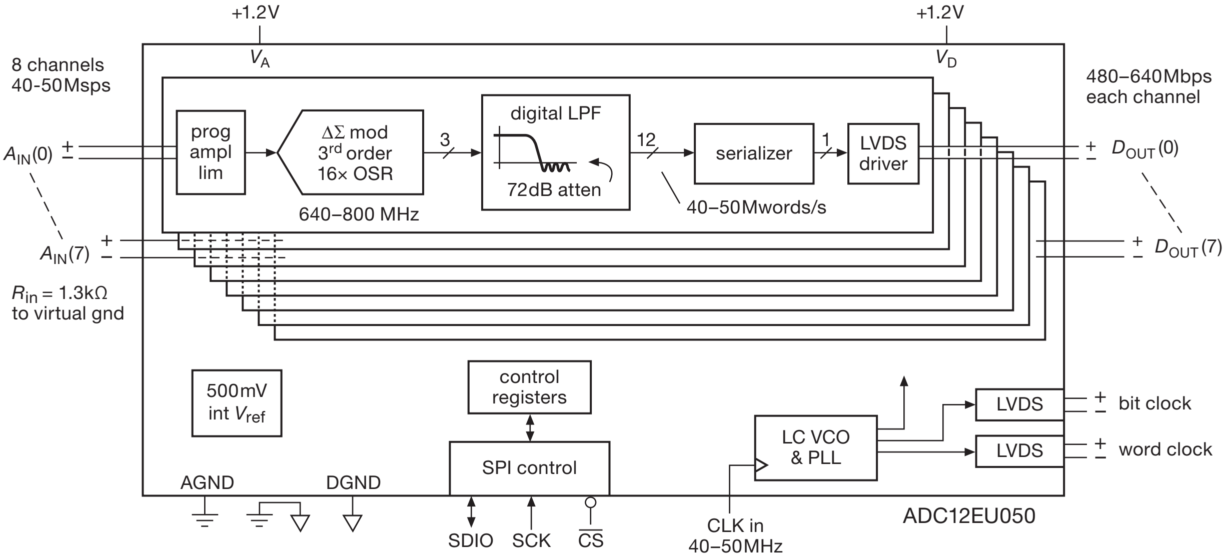


Figure 13.84. The ADC12EU050 is a fast 8-channel delta–sigma converter with 3-bit wordstreams generated by third-order modulators. Its PLL includes an on-chip *LC* VCO to generate the 16× oversampling clock from the 40–50 MHz sample-rate clock input.

integrated circuit, and also often incorporated within more complex ICs. A PLL contains a phase detector, amplifier, and voltage-controlled oscillator (VCO), and represents a blend (sometimes called "mixed signal") of digital and analog techniques. A few of its applications, which we will discuss shortly, are frequency multiplication and frequency synthesis, clock generation and clock recovery, tone decoding, and demodulation of AM, FM, and digitally modulated signals.

In the past there had been some reluctance to use PLLs, partly because of the complexity of discrete PLL circuits and partly because of a feeling that they cannot be counted on to work reliably. With inexpensive and easy-to-use PLLs now widely available, the first barrier to their acceptance has vanished. And with proper design and conservative application, the PLL is as reliable a circuit element as an op-amp or flip-flop.

Figure 13.85 shows the classic PLL configuration. The phase detector is a device that compares two input frequencies, generating an output that is a measure of their phase difference (if, for example, they differ in frequency, it gives a periodic output at the difference frequency). If f_{in} doesn't equal f_{VCO}, the phase-error signal, after being filtered and amplified, causes the VCO frequency to deviate in the direction of f_{in}. If conditions are right (lots more on that soon), the VCO will quickly "lock" to f_{in}, maintaining a fixed phase relationship with the input signal.

At that point the filtered output of the phase detector is a dc signal, and the control input to the VCO is a measure of the input frequency, with obvious applications to tone decoding (sometimes used over telephone lines) and FM demodulation. The VCO output is a locally generated frequency equal to f_{in}, thus providing a clean replica of f_{in}, which may itself be noisy. Since the VCO output can be a triangle wave, sinewave, or whatever, this provides a nice method of generating a sinewave, say, locked to a train of input pulses.

In one of the most common applications of PLLs, a modulo-*n* counter is hooked between the VCO output and the phase detector, thus generating a multiple of the input reference frequency f_{in}. This is an ideal method for generating clocking pulses at a multiple of the powerline frequency for integrating ADCs (dual-slope, charge-balancing) in order to have infinite rejection of interference at the powerline frequency and its harmonics. It also provides the basic technique of frequency synthesizers.

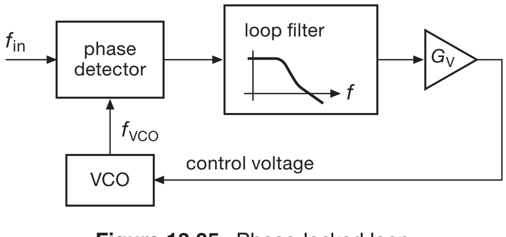


Figure 13.85. Phase-locked loop.

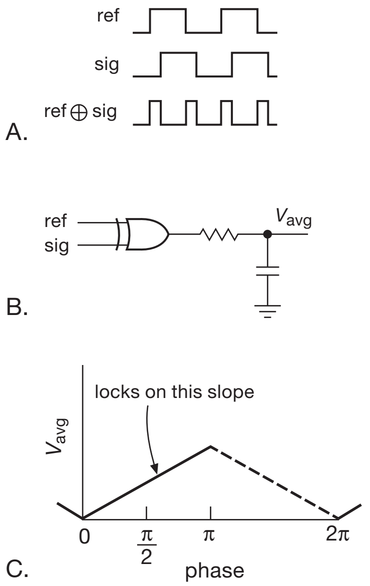


Figure 13.86. Exclusive-OR-gate phase detector (type I).

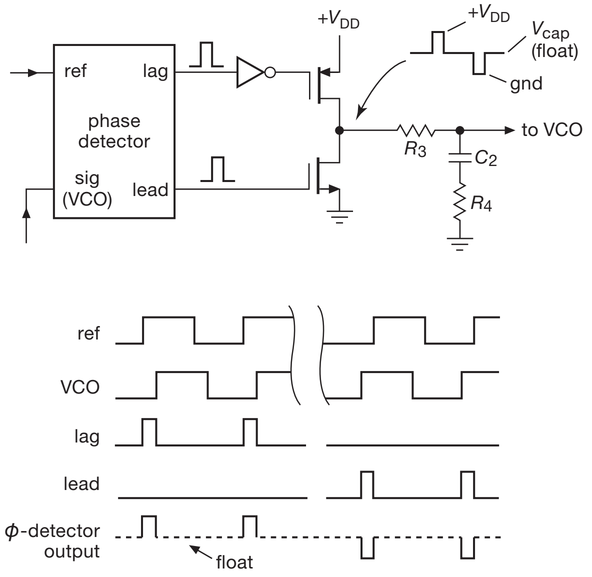


Figure 13.87. Edge-sensitive lead–lag phase detector (type II).

13.13.2 PLL components

A. The phase detector

Let's begin with a look at the phase detector (PD). There are two basic types, sometimes referred to as type I and type II.

The **type I phase detector** is applicable to either analog- or digital-input signals, and performs a simple multiplication of the inputs. For *digital* signals this is just an exclusive-OR gate (Figure 13.86). With lowpass filtering, the graph of the output voltage versus phase difference is a ramp, as shown, for input square waves of 50% duty cycle. For *analog* signals the type I "linear" phase detector is a true analog multiplier (called a "four-quadrant multiplier" or a "balanced mixer"), with similar output-voltage-versus-phase characteristics as with the digital XOR phase detector. Highly linear phase detectors of this type are essential for synchronous detection (also known as *lock-in detection*).

The **type II phase detector**, on the other hand, is a purely digital beast, driven by digital transitions (edges). It is sensitive only to the relative timing of *edges* between the signal and VCO input, as shown in Figure 13.87. The phase-comparator circuitry generates either *lead* or *lag* output pulses, depending on whether the transitions of the VCO output occur before or after the transitions of the reference signal, respectively. The width of these pulses is equal to the time between the respective edges, as shown. The output circuitry then either sinks or sources current (respectively) during those pulses and is otherwise open-circuited, generating an average output-voltage-versus-phase difference like that in Figure 13.88. This is completely independent of the duty cycle of the input signals, unlike the situation with the type I phase detector.

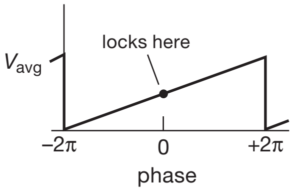


Figure 13.88. Type II phase detector output.

Another nice feature of this phase detector is the fact that the output pulses disappear entirely when the two signals are in lock. This means that there is no "ripple" present at the output to generate periodic phase modulation in the loop, as there is with the type I phase detector. And while we're lavishing praise on the type II, let's point out that it has the pleasant property of producing an average dc output that is indicative of the *sign* of the frequency error (Figures 13.89–13.91). For this reason it's sometimes called a "phase-frequency detector" (PFD). We'll see how that ensures prompt lockup in a PLL.

The classic 74HC4046 PLL (which includes both oscillator and phase detector) gives you a choice (it contains both types of phase detector). Here is a comparison of the properties of the two basic types of phase detector:

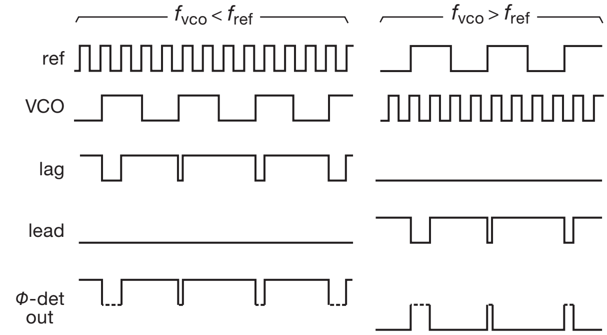


Figure 13.89. The type II phase-detector produces an average dc output that indicates the *sign* of the frequency error.

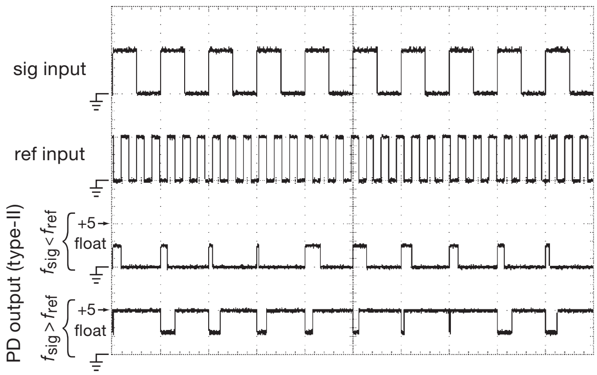


Figure 13.90. Measured waveforms from a type II phase detector driven with greatly mismatched frequencies. The 1 kHz signal and π-kHz reference inputs shown produce the phase-detector output shown in the third trace when driving a 10k–10k resistive divider that floats to +2.5 V. The bottom trace shows what happens when the inputs are interchanged. Horizontal: 1 ms/div.

		Type II
	Type I	*edge triggered*
Parameter	*exclusive*-OR	*("charge pump")*
Input duty cycle	50% optimum	Irrelevant
Lock on harmonic?	Yes	No
Rejection of noise	Good	Poor
Residual ripple at $2f_{\rm IN}$	High	Low
Lock range (L)	Full VCO range	Full VCO range
Capture range	$fL(f<1)$	L
Output frequency when out of lock	$f_{\rm center}$	$f_{\rm min}$

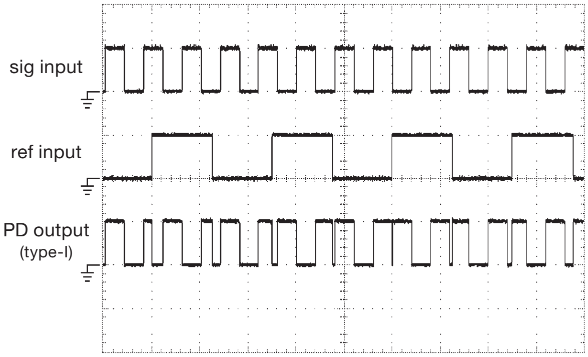


Figure 13.91. By contrast, the type I (XOR) phase detector, presented with the signal and reference frequencies of Figure 13.90, produces a frenetic rail-to-rail output whose dc average is $V_{\rm DD}/2$. Horizontal: 0.4 ms/div.

There is one additional point of difference between the two kinds of phase detectors. The type I detector is always generating an output wave, which must then be filtered by the loop filter (much more on this later). Thus, in a PLL with type I phase detector, the loop filter acts as a lowpass filter, smoothing this full-swing logic-output signal. There will always be residual ripple, and consequent periodic phase variations, in such a loop. In circuits where PLLs are used for frequency multiplication or synthesis, this adds "phase-modulation sidebands" to the output signal.

By contrast, the type II phase detector generates output pulses only when there is a phase error between the reference and the VCO signal. Since the phase-detector output otherwise looks like an open circuit, the loop filter capacitor then acts as a voltage-storage device, holding the voltage that gives the right VCO frequency. If the reference signal moves away in frequency, the phase detector generates a train of short pulses, charging (or discharging) the capacitor to the new voltage needed to put the VCO back into lock. It is a phase-error integrator.

"Dead zone" and "backlash"

A persistent problem with early PLLs using type II phase detectors was the presence of a *dead zone*: the phase pulses became vanishingly small at nearly zero phase error, so the loop tended to "hunt" (bounce back and forth), producing phase modulation and jitter. And this was exacerbated by the effects of capacitive loading at the phase-detector output. For applications that need a clean signal (for example, the synthesized oscillator in a cellphone, communications receiver, or RF frequency synthesizer) this was (and is) a serious problem. The solution, now nearly universally adopted, is to introduce some intentional overlap of the sourcing and sinking output pulses; to make this work you need to reconfigure the phase detector to produce *current* pulses (rather than voltage pulses).

Figure 13.92 shows how this is done: the current source or sink is activated by the first rising edge of the signals being compared (reference or signal, respectively), but it is not turned off until a short interval after the complementary

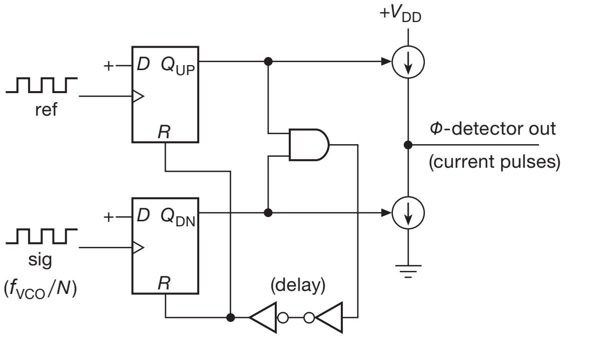


Figure 13.92. Improved type II phase detector (the '9046 version is shown here) replaces the switches with current sources and prevents a dead zone and backlash by creating intentional overlap of phase pulses.

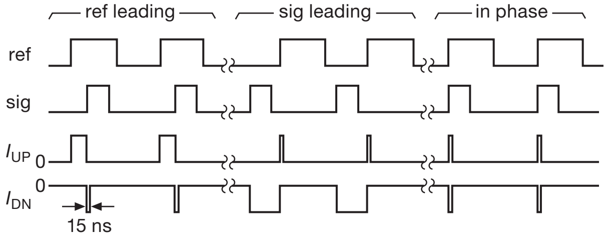


Figure 13.93. Current pulses (both sourcing and sinking) for the phase detector of Figure 13.92. The 15 ns pulses are created by the anti-backlash circuit.

current source is switched on. This "anti-backlash" circuit guarantees that the output pulses never vanish. When the two signals are exactly in phase (the loop is *locked*), the current pulses are short (∼15 ns for the 74HCT9046, an improved version of the classic '4046) and of opposite sign, thus cancelling (Figure 13.93). Moving away from lock, a small phase difference produces an unbalanced pair of current pulses. This linear behavior around zero phase solves the problem; and capacitive loading does not cause trouble, because it behaves like a perfect integrator.

A cheap cure for backlash, if you need it, is to put a large resistor across the loop filter capacitor (C_2 in Figure 13.87), which biases the loop away from the dead zone. The tradeoff is that you introduce a nonzero phase shift, which is not well defined; but at least you've gotten rid of the jitter.

B. The VCO

An essential component of a PLL is an oscillator whose frequency can be controlled by the phase-detector output. We discussed VCOs back in Chapter 7 (§§7.1.4D and 7.1.5D), and we'll see them again presently, in a PLL design example. For now, let's just look at the simple *RC* voltage-controlled oscillator used in the '4046 and its successors (Figure 13.94).

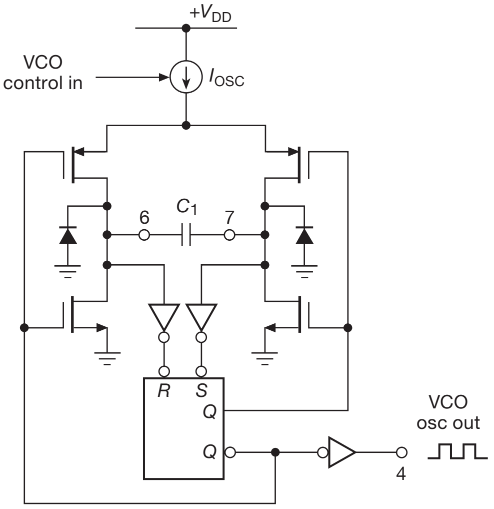


Figure 13.94. Voltage-controlled *RC* oscillator used in the classic '4046 PLL. Output frequency is approximately proportional to controlled current I_{osc}, which charges external capacitor C_1 alternately through the pMOS switches.

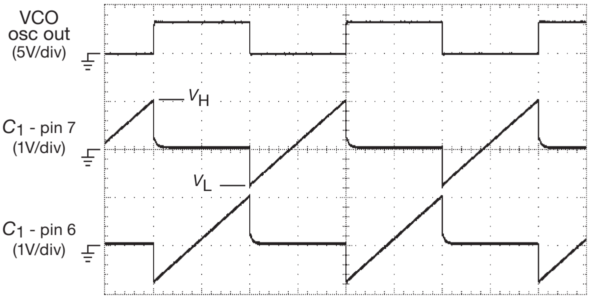


Figure 13.95. Observed waveforms of a 74HC4046 oscillator, with V_{CC}=3.3 V; with a 5 V supply the ramp starts at the same voltage but ends 0.2 V higher. Horizontal: 10μs/div.

The operation is simple: the flip-flop's output holds one side of the external timing capacitor C_1 at ground (via an nMOS switch) while coupling charging current I_{osc} to the other side (via a pMOS switch). The cycle reverses when the rising voltage reaches the inverter's threshold, approximately +1.1 V. Figure 13.95 shows measured waveforms for an 'HC4046 powered from +3.3 V, with C_1=10 nF and I_{osc}=0.85 mA. Note that each cycle begins at approximately −0.7 V, clamped at a diode drop below ground when the high side is switched to ground.

In a PLL you often want to restrict the oscillator's tuning range, to span a modest range of frequency centered on the

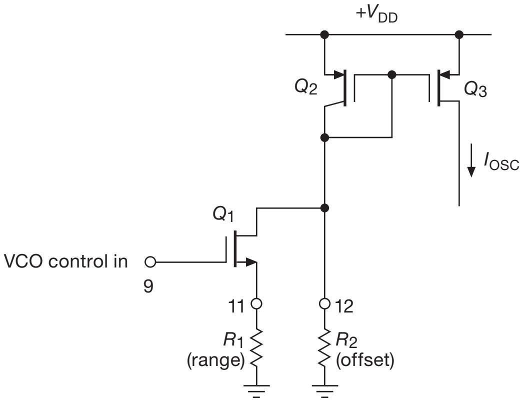


Figure 13.96. External resistors set the range and offset of the ground-referenced programming voltage in the classic high-voltage CD4046 VCO. The 74HC4046 types use op-amps to more tightly control the R_1 and R_2 currents.

desired output frequency. For example, the PLL in an FM radio needs to span ±10 MHz about a central frequency of ∼100 MHz; and we'll see examples later in which this range can be as narrow as ±0.01% (a "voltage-controlled crystal oscillator," VCXO). The oscillator in the '4046 accommodates this rather simply (Figure 13.96), by letting you use a pair of resistors: R_1 sets the span ($f_{\text{max}}-f_{\text{min}}$), and R_2 sets the minimum frequency. In this circuit Q_1 is a programmable current sink, bounced off a pMOS current mirror to create the charging current I_{osc}.

Soon enough we'll see other PLLs, both with and without integrated on-chip oscillators. First, though, we want to have some fun with a PLL design, using our new friend the '4046. Keep in mind, though, that PLLs (and their VCOs) don't have to be restricted to maximum speeds in the tens of megahertz. In fact, it's probably accurate to say that most PLLs in the world earn their livings at frequencies in the hundreds to thousands of megahertz. At those frequencies you don't use *RC* timing – instead you use *LC* circuits (tuned with a voltage-variable capacitor, known as a *varactor*), or a *ring oscillator* (a chain of inverters) tuned by adjusting the operating current (a "starved inverter chain"), or more exotic techniques such as a surface acoustic wave (SAW) delay-line oscillator or a resonator made from a silicon microelectromechanical system (MEMS). A VCO for use in a phase-locked loop doesn't have to be particularly linear in its frequency-versus-control-voltage characteristic, but if it is highly nonlinear, the loop gain (see below) will vary according to the signal frequency, compromising loop stability.

13.13.3 PLL design

A. Closing the loop

The phase detector gives us an error signal related to the phase difference between the signal and reference inputs. The VCO allows us to control its frequency with a voltage input. It would seem straightforward to treat this like any other feedback amplifier, closing the loop with some gain, just as we did with op-amp circuits.

However, there is one essential difference. Previously, the quantity adjusted by feedback was the same quantity measured to generate the error signal, or at least a proportional quantity. For example, in a voltage amplifier we measured output voltage and adjusted input voltage accordingly. But in a PLL there's an integration; we measure *phase*, but adjust *frequency*, and phase is the integral of frequency. This introduces a 90° lagging phase shift in the loop.

This integrator included within the feedback loop has important consequences, since an additional 90° of lagging phase shift at a frequency where the loop gain is unity can produce oscillations. A simple solution is to avoid any further lagging components within the loop, at least at frequencies where the loop gain is close to unity. After all, op-amps have a 90° lagging phase shift over most of their frequency range, and they work quite nicely. This is one approach, and it produces what is known as a "first-order loop." It looks just like the PLL block diagram shown earlier, with the lowpass filter omitted.

Although they are useful in many circumstances, first-order loops don't have the desirable property of acting as a "flywheel," allowing the VCO to smooth out noise or fluctuations in the input signal. Furthermore, a first-order loop will not maintain a fixed phase relationship between the reference and VCO signals, because the phase detector output drives the VCO directly. A "second-order loop" has additional lowpass filtering within the feedback loop (as drawn earlier), carefully designed to prevent instabilities. This provides flywheel action and also reduces the "capture range" and increases the capture time. Furthermore, with type II phase detectors, a second-order loop guarantees phase lock with zero phase difference between reference and VCO, as will be explained soon. Second-order loops are used almost universally, because the applications of PLLs usually demand an output frequency with low phase noise and some "memory," or flywheel action. Second-order loops permit high loop gain at low frequencies, resulting in high stability (in analogy with the virtues of high loop gain in feedback amplifiers). Let's get right down to business, illustrating the use of phase-locked loops with a design example.

13.13.4 Design example: frequency multiplier

Generating a fixed multiple of an input frequency is one of the most common applications of PLLs. This is done in frequency synthesizers, in which an integer multiple *n* of a stable low-frequency reference signal (1 Hz, say) is generated as an output; *n* is settable digitally, giving you a flexible signal source, easily controlled through a digital interface. In more mundane applications, you might use a PLL to generate a clock frequency locked to some other reference frequency already available in the instrument. For example, suppose we want to generate a 61.440 kHz clock signal for a dual-slope ADC. That particular choice of frequency permits 7.5 measurement cycles per second, allowing 4096 clock periods for the ramp-up (remember that dual-slope conversion uses a constant time interval) and 4096 counts full scale for the constant-current ramp-down. The unique virtue of a PLL scheme is that the 61.440 kHz clock can be locked to the 60 Hz powerline (61,440=60×1024), giving infinite rejection of 60 Hz pickup present on any signal input to the converter, as we discussed in §13.8.4.

We begin with the standard PLL scheme, with a divide-by-*n* counter added between the VCO output and the phase detector (Figure 13.97). In this diagram we indicate the units of gain for each function in the loop. That will be important in our stability calculations. Note particularly that the phase detector converts phase to voltage and that the VCO converts voltage to the time derivative of phase (i.e., frequency). This has the important consequence that the VCO is actually an integrator, with phase representing the variable in the lower part of the diagram; a fixed input-voltage error produces a linearly rising phase error at the VCO output. The lowpass filter and the divide-by-*n* counter both have unitless gain.

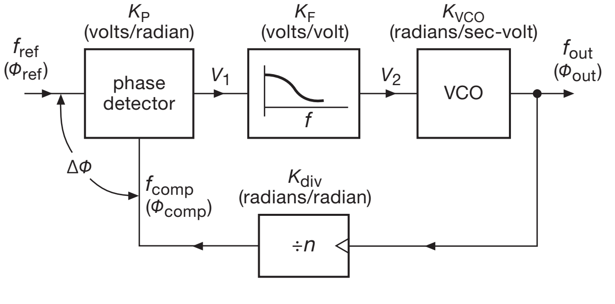


Figure 13.97. Frequency-multiplier block diagram.

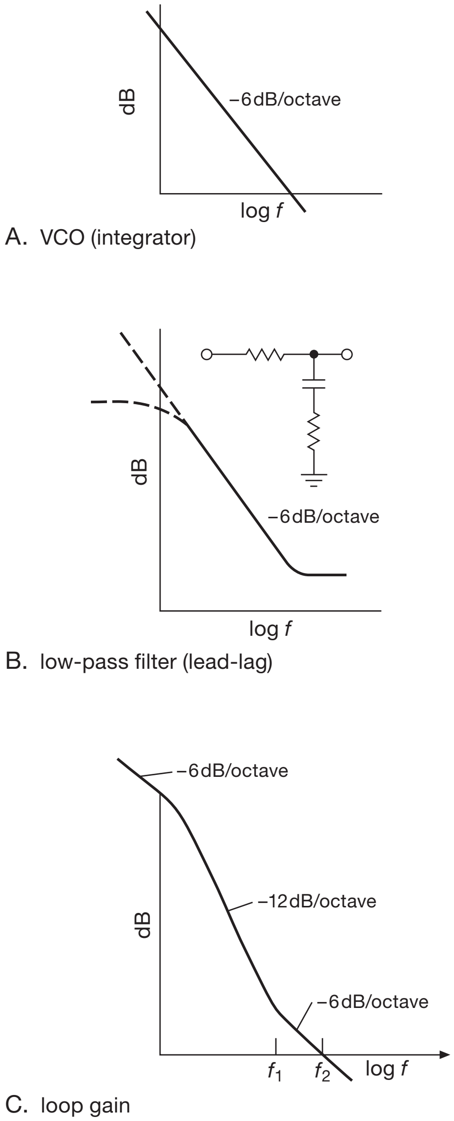


Figure 13.98. PLL Bode plots.

A. Stability and phase shifts

The trick to a stable second-order PLL is shown in the Bode plots of loop gain in Figure 13.98. The VCO acts as an integrator, with $1/f$ response and 90° lagging phase shift (i.e., its response is proportional to $1/j\omega$, a current source driving a capacitor). In order to have a respectable phase margin (the difference between 180° and the phase shift around the loop at the frequency of unity loop gain), the lowpass filter has an additional resistor in series with the capacitor to stop the rolloff at some frequency (fancy name: a "zero"). The combination of these two responses produces the loop gain shown. As long as the loop gain

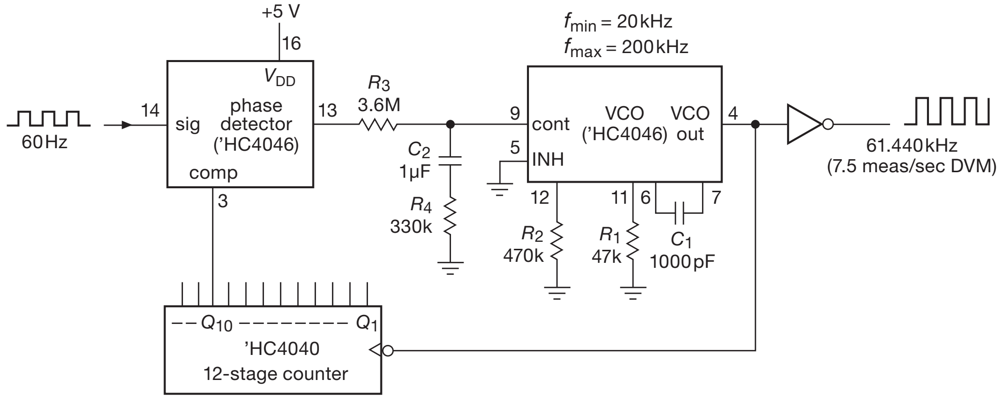


Figure 13.99. Using a PLL multiplier to generate a clock locked to the 60 Hz ac line. Parts values are for TI's CD74HC4046A.

rolls off at 6 dB/octave in the neighborhood of unity loop gain, the loop will be stable. The "lead–lag" lowpass filter does the trick, if you choose its properties correctly (this is the same as lead–lag compensation in op-amps). Next we'll see how it is done.

B. Loop-gain calculations

Figure 13.99 shows the schematic of the 61.440 kHz PLL synthesizer. Both the phase detector and the VCO are parts of an 'HC4046 CMOS PLL. We used the edge-triggered (type II) phase detector in this circuit (recall the 4046 contains both kinds). Its output comes from a pair of CMOS transistors generating saturated pulses to V_{DD} or ground. It is really a three-state output, as explained earlier, since it is in the high-impedance state except during actual phase-error pulses.

The VCO allows you to set the minimum and maximum frequencies corresponding to control voltages of zero and V_{DD}, respectively, by choosing R_1, R_2, and C_1 according to some design graphs. We have made the choices shown, based upon an initial design from the datasheet, validated (and adjusted!) by some bench measurements – see the "real-world" commentary that follows, in §13.13.4E. *Note*: the 4046 has chronic severe supply-sensitivity disease; check the graphs on the datasheet. The rest of the loop is standard PLL procedure.

Having rigged up the VCO range, the remaining task is the lowpass-filter design. This part is crucial. We begin by writing down the loop gain, as in the "PLL gain calculation" box, considering each component (refer to Figure 13.97). Take special pains here to keep your units consistent; don't switch from f to ω or (worse) from hertz to kilohertz. Having chosen the oscillator components, divider ratio, and supply voltages, we need to determine the one remaining gain term (that of the loop filter), K_{F}. We do this by writing down the overall loop gain, remembering that the VCO is an integrator:

$$\phi_{\mathrm{out}} = \int V_2 K_{\mathrm{VCO}} dt.$$

The loop gain is therefore given by

$$\begin{aligned}\text{Loop gain} &= K_{\mathrm{P}} K_{\mathrm{F}} \frac{K_{\mathrm{VCO}}}{j\omega} K_{\mathrm{div}} \\ &= 0.40 \times \frac{1 + j\omega R_4 C_2}{1 + j\omega(R_3 C_2 + R_4 C_2)} \times \frac{3.77 \times 10^5}{j\omega} \times \frac{1}{1024}.\end{aligned}$$

Now comes the choice of frequency at which the loop gain should pass through unity. The idea is to pick a unity-gain frequency high enough so that the loop can follow input-frequency variations you want to follow, but low enough to provide flywheel action to smooth over noise and jumps in the input frequency. For example, a PLL designed to demodulate an FM input signal, or to decode a rapid sequence of input tones, needs to have rapid response (for the FM input signal, the loop should have as much bandwidth as the input signal, i.e., response up to the maximum modulating frequency; while to decode input tones, its response time must be short compared with the time duration of the tones). On the other hand, a loop such as this one, designed to generate a fixed multiple of a stable and slowly varying input frequency, should have a low unity-gain frequency. That will reduce phase noise at the output and make the PLL insensitive to noise and glitches on the input. It will hardly even notice a short dropout of input signal, because the voltage held on the filter capacitor will instruct the VCO to continue producing the same output frequency.

In this case, we choose the unity-gain frequency f_2 to be 2 Hz, or 12.6 radians per second. This is well below

PLL GAIN CALCULATION

Component	*Function*	*Gain*	*Gain calculation* $(V_{\rm DD}=+10{\rm V})$
Phase detector	$V_i = K_{\rm P}\Delta\phi$	$K_{\rm P}$	$K_{\rm P}=\dfrac{V_{\rm DD}}{4\pi}$ (0 to $V_{\rm DD} \leftrightarrow -360°$ to +360°)
Lowpass filter	$V_2 = K_{\rm F}V_1$	$K_{\rm F}$	$K_{\rm F}=\dfrac{1+j\omega R_4C_2}{1+j\omega(R_3C_2+R_4C_2)}$ volts/volt
VCO	$\dfrac{d\phi_{\rm out}}{dt}=K_{\rm VCO}V_2$	$K_{\rm VCO}$	20kHz $(V_2=1{\rm V})$ to 200kHz $(V_2=4{\rm V})$ $\rightarrow K_{\rm VCO}=$ 60kHz/volt $= 3.77\times 10^5$ radians/second-volt
Divide-by-*n*	$\phi_{\rm comp}=\dfrac{1}{n}\phi_{\rm out}$	$K_{\rm div}$	$K_{\rm div}=\dfrac{1}{n}=\dfrac{1}{1024}$

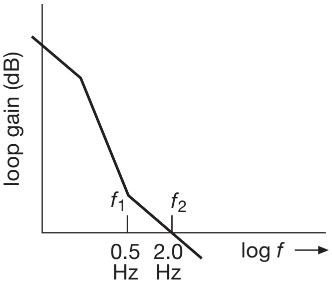


Figure 13.100. Stabilizing a second-order PLL: −6 dB/octave rolloff of loop gain around the frequency of unity gain.

the reference frequency, and you wouldn't expect genuine powerline frequency variations on a scale shorter than this (remember that the 60 Hz power is generated by enormous generators with lots of mechanical inertia). As a rule of thumb, the breakpoint of the lowpass filter (its "zero") should be lower by a factor of at least 3 to 5 for a comfortable phase margin. Remember that the phase shift of a simple *RC* goes from 0° to 90° over a frequency range of roughly 0.1 to 10 times the −3 dB frequency (its "pole"), with a 45° phase shift at the −3 dB frequency. In this case we put the frequency of the zero, f_1, at 0.5 Hz, or 3.1 radians per second (Figure 13.100). The breakpoint f_1 determines the R_4C_2 time constant: $R_4C_2 = 1/2\pi f_1$. Tentatively, take $C_2 = 1\mu$F and $R_4 = 330$k. Now all we do is choose R_3 so that the magnitude of the loop gain equals 1 at f_2. In this case, that works out to $R_3 = 3.6\,\mathrm{M}\Omega$.

Exercise 13.7. Show that these choices of filter components actually give a loop gain of magnitude 1.0 at f_2=2.0 Hz.

Sometimes the filter values are inconvenient, so you have to readjust them, or move the unity-gain frequency somewhat. With a CMOS phase-locked loop these values are acceptable (the VCO input terminal has a typical input impedance of $10^{12}\,\Omega$). For VCOs with a low input impedance you might want to use an external op-amp buffer.

We used an edge-triggered (type II) phase detector in this circuit example because of its simplified loop filter. In practice, that might not be the best choice for a PLL locked to the 60 Hz powerline because of the relatively high noise level present on the 60 Hz signal: many engineers have stumbled on this point, with a noisy reference signal causing false type-II triggering. With careful design of the analog input circuit (e.g., a lowpass filter followed by a Schmitt trigger) the type-II phase detector would likely perform satisfactorily; otherwise an exclusive-OR (type I) phase detector should be used.

C. "Cut and try"

For some people, the art of electronics consists of fiddling with filter component values until the loop "works." If you are one of those,[119] we will oblige you by looking the other way. We presented these loop calculations in detail because we suspect that much of the PLL's bad reputation is the result of too many people "looking the other way." Nevertheless, we can't resist supplying a hot tip for cut-and-try addicts: R_3C_2 sets the smoothing (response) time of the loop,

[119] You can derive some comfort (and perhaps even hold your head high) from some remarks in the TI app note (referring to the loop-filter design) "Optimizing by trial and error should be considered in all cases."

and R_4/R_3 determines the damping, i.e., absence of overshoot for step changes in frequency. You might begin with a value of R_4 somewhere in the range of 10% to 20% of R_3.

D. Loop damping and jitter

A side effect of the nonzero "damping" resistor R_4 is the creation of some jitter in the PLL output. An easy way to see this is to realize that even at high frequencies the loop filter permits a fraction $R_4/(R_3+R_4)$ of the raw phase-detector output to reach the VCO. For typical ratios, $R_3 \approx 10R_4$, this can add substantial jitter to the VCO output. The usual solution is to add a small capacitor ($\sim C_2/20$) from the VCO control input to ground, preferably close to the VCO pin to filter any other high-frequency noise as well.

E. PLL real-world design

We sailed through this design example, having faith that the information in the datasheets for the IC we chose (the popular 'HC4046) was reliable. That faith was, perhaps, a bit misplaced. To give this cautionary section some context, here's the story (the short version) of our oscillator component choices in Figure 13.99:

We wanted a 3× safety margin for our 61 kHz center frequency, so we set f_{min}=20 kHz and f_{max}=200 kHz. We chose TI's CD74HC4046A because it was RCA's original time-proven design. Based on the datasheet's graphs, a good value for the timing capacitor C_1 was 1000 pF. For the timing resistors one of us started with the R_1 plots and came up with 30k and 300k for R_1 and R_2, and we edited the figure accordingly. The other author started with the R_2 plot (as suggested by two manufacturers, but it should not have mattered) and came up with 40 k and 410 k. Worried about these and other inconsistencies (e.g., one spec in the TI datasheet gives f_{osc}=400 kHz, typ, for C_1=1 nF and R_2=220k, while their Figure 27 suggests it should be more like 33 kHz), we went to the bench and found actual values of 45k and 482k for the desired frequency range. The values we chose initially would have worked, despite the factor of 1.5 deviation from reality, but they would have used up half of our 3× safety margin.[120]

So, what's going on here? Each of the 'HC4046 manufacturers uses a different circuit for their VCO design.[121] Although intended to be predictable and linear, in practice the VCO control is nonlinear, and its parameters vary with control current, supply voltage, and operating frequency, especially above 10 MHz. Although you can find analytical expressions for the VCO's frequency (ON Semi app note AN1410), the recommended method is still to start with the timing component values (R_1, R_2, and C_1) from datasheet graphs; then the designer is sternly admonished to adjust and validate those values with careful bench measurements before committing to manufacturing.

This sort of variability and lack of confident predictability leads us to render this advice:

(1) choose one manufacturer for your production design, and do not allow substitutes;
(2) choose a wide safety margin for f_{min} and f_{max}, such as the 3× factor in our Figure 13.99;
(3) replace your initial paper calculations with measured bench values for production.

Rule (1) applies to any linear functionality in a logic IC, e.g., mixed-signal functions such as phase comparators, oscillators, VCOs, mixers, Schmitt triggers, monostables, or comparators.

13.13.5 PLL capture and lock

Once locked, it is clear that a PLL will stay locked as long as the input frequency doesn't wander outside the range of the feedback signal and doesn't change more rapidly than the loop's bandwidth can track. An interesting question to ask is how PLLs get locked in the first place. After all, an initial frequency error results in a periodic output from the phase detector at the difference frequency. After filtering by the lowpass filter, it is reduced to small-amplitude wiggles rather than to a nice clean dc error signal.

A. Capture transient

The answer is a little complicated. First-order loops will always lock, because there is no lowpass attenuation of the error signal. Second-order loops may or may not lock, depending on the type of phase detector and the bandpass of the lowpass filter. In addition, the exclusive-OR (type I) phase detector has a limited *capture* range that depends on the filter time constant (this fact can be used to advantage

[120] We made additional bench measurements on both old and new 74HC4046A parts from TI, NXP, ON Semi, and Fairchild. We found good self-consistency within any single manufacturers parts; the variations in span and zero frequency within a batch and over a 15-year period were generally under 5%. Measured frequencies for the NXP, ON Semi, and Fairchild parts deviated by +5%, +160%, and −60%, respectively, from that of the TI parts, when configured with the timing component values shown in Figure 13.99.

[121] For example, TI biases R_2 from $V_{\text{DD}}-0.7$ V, whereas ON Semi biases it from $V_{\text{DD}}/3$. Their current mirrors have nominal gains of 7.5 and 25 respectively. We think NXP's 74HCT9046 is arguably the best mannered '4046 part available, and the NXP datasheets have better graphs.

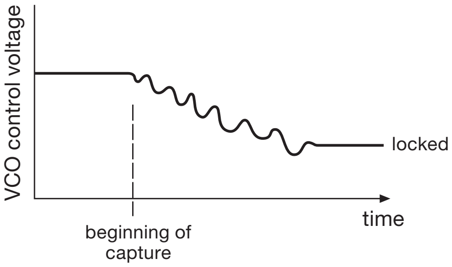


Figure 13.101. PLL capture transient.

if you want a PLL that will lock to signals only within a certain frequency range).

For a type I phase detector you might wonder how the loop can lock up at all, because, with the phase detector's output being periodic at the difference frequency, the VCO's frequency should just wiggle back and forth forever. But looking more closely, the capture transient goes like this: as the (phase) error signal brings the VCO frequency closer to the reference frequency, the error-signal waveform varies more slowly, and vice versa. So the error signal is asymmetric, varying more slowly over that part of the cycle during which f_{VCO} is closer to f_{ref}. The net result is a nonzero average, i.e., a dc component that brings the PLL into lock.[122] If you look carefully at the VCO control voltage during this *capture transient*, you'll see something like what is shown in Figure 13.101. That final overshoot has an interesting cause. Even when the VCO *frequency* reaches its correct value (as indicated by correct VCO control voltage), the loop isn't necessarily in lock, because the *phase* may be wrong. So it may overshoot. As with snowflakes, each capture transient is an individual – it looks a bit different each time.

For a PLL with a type II phase detector, the situation is considerably simpler: because this kind of detector produces a dc component indicating the direction of frequency error (recall it's a "phase-frequency detector"), the VCO's frequency is steered rapidly in the right direction.

B. Capture and lock range

For the exclusive-OR (type I) phase detector, the capture range is limited by the lowpass-filter time constant. This makes sense, because if you begin sufficiently far away in frequency, the error signal will be attenuated so much by the filter that the loop will never lock. It should be evident that a longer filter time constant results in narrower capture range, as does reduced loop gain. The edge-triggered phase detector does not have this limitation, because it acts like a true integrator of the phase-error charge pulses. Both types have a lock range extending to the limits of the VCO, given the available control input voltage.

C. Capture time

PLLs with type II (integrating) phase detectors will always lock (assuming the VCO has sufficient tuning range, of course), with a time constant characteristic of the loop bandwidth.

A type I (multiplier or mixer) phase detector, if followed by an integrating loop filter, will also lock – but it can take a very long time if the loop bandwidth is narrow. It can be shown that the lockup time is roughly $(\Delta f)^2/\text{BW}^3$, where Δf is the initial frequency error and BW is the loop bandwidth. So a PLL with 100 Hz loop bandwidth and 100 kHz comparison frequency might take a minute to lock up if the VCO's initial frequency is 10% away from lock.

In such cases you sometimes see a cute trick: a slow full-range sawtooth is applied to the VCO control voltage of the unlocked loop, until lock takes place. For example, I have in my hand (I'm typing one-handed) an Efratom model FRS rubidium frequency standard, which uses the weak but extremely stable atomic resonance in an optically pumped vapor cell as a reference to which a high-quality crystal oscillator is phase locked. The 20 MHz ovenized crystal oscillator (XO) is voltage controlled (a VCXO), with a narrow tuning range (of order ±1 kHz); it is the flywheel in a low-bandwidth PLL (the loop integrator has R=2M, C=1μF).

Without some help, this thing would take forever to lock. The helpful "Operation Manual" explains how they do it: "When no 'lock' signal is present...[there is] a slow down sweep of the crystal control voltage of about 250 mV per second. Sweeping continues until 'Lock' occurs. 'Lock' then disables the sweep circuit by disconnecting [analog switch] U3 pins 13 and 14, and connecting U3 pins 12 and 14. This switching places the integrator under fundamental loop control."

Such gimmicks aren't necessary with type II phase detectors, as we remarked above, thanks to their indication of both phase and (sign of) frequency difference. But passive mixer-type phase detectors (therefore type I) are prevalent in communications systems at very high radio frequencies, where the digital phase-frequency detector is impractical.

[122] Another way of looking at it is to realize that the error signal weakly modulates the VCO at the difference frequency $\Delta f = |f_{\text{ref}} - f_{\text{VCO}}|$. That frequency modulation puts weak symmetrical sidebands onto f_{VCO}, spaced apart by Δf. One of these is exactly at f_{VCO}, producing an average dc output component from the phase detector, which the loop filter integrates (type II) to bring the system into lock.

13.13.6 Some PLL applications

We have spoken already of the common use of phase-locked loops in frequency multiplication. The latter application, as in the preceding example, is so straightforward that there should be no hesitation about using these mysterious PLLs. In simple frequency-multiplication applications (e.g., the generation of higher clock frequencies in a digital system) there isn't even any problem of noise on the reference signal, and a first-order loop may suffice.

As will become evident, what you care about in a PLL depends on the application: there's a tradeoff among wide tuning range versus high quality (low phase noise, low jitter, low spurious frequency components) versus frequency step size versus loop bandwidth (and switching speed) versus low external component count. For example, for a microprocessor or memory clocking application you don't need high-quality waveforms, and you need only coarse tuning steps; for a cellphone synthesizer you want low phase noise and spurs, with tuning range and step size matched to the cell band and channelization; for a general-purpose sinewave synthesizer you want low phase noise and spurs, small tuning steps, and wide tuning range; for high-speed serial links you care about jitter, as you do when clocking high-quality ADCs (where jitter translates to distortion); and for a motherboard clock generator you'd like a single-chip solution that generates a suite of standard clocks (for the processor, memory, video, internal buses like PCIe and SATA, external serial ports like USB and Ethernet, and so on) without great concern about signal quality.

We would like now to describe two important variations (known as "n/m" and "fractional-n" synthesis) on this basic frequency-multiplication scheme; we'll continue with some other interesting applications of phase-locked techniques, to give an idea of the diversity of PLL uses. Finally, we'll conclude the discussion of phase-locked loops with examples of contemporary PLL ICs, which use a variety of clever tricks to create on-chip oscillators with admirable performance.

A. Fractional-*n* synthesis

The frequency multiplication scheme of Figure 13.97 generates an output frequency that is restricted to an integral multiple of the reference input: $f_{\text{out}}=nf_{\text{ref}}$. That's fine for an application like that of Figure 13.99, but it's not of much use for something like a general-purpose sinewave synthesizer, where you want to generate any old output frequency, perhaps settable down to 1 Hz, or even 0.001 Hz.

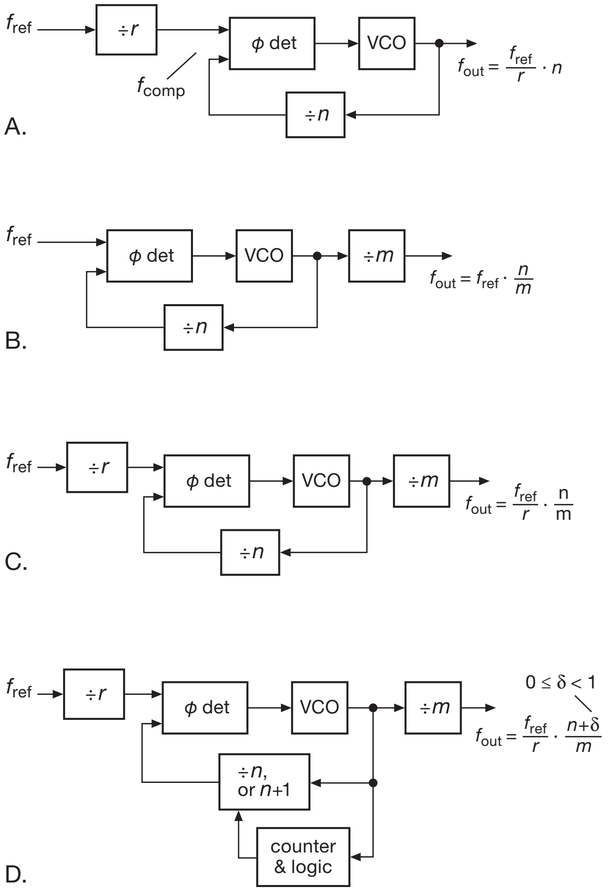

A.
f_{ref}
÷r
f_{comp}
φ det
VCO
÷n
$f_{\text{out}} = \frac{f_{\text{ref}}}{r} \cdot n$
B.
f_{ref}
φ det
VCO
÷m
÷n
$f_{\text{out}} = f_{\text{ref}} \cdot \frac{n}{m}$
C.
f_{ref}
÷r
φ det
VCO
÷m
÷n
$f_{\text{out}} = \frac{f_{\text{ref}}}{r} \cdot \frac{n}{m}$
D.
f_{ref}
÷r
φ det
VCO
÷m
÷n, or n+1
counter & logic
$0 \le \delta < 1$
$f_{\text{out}} = \frac{f_{\text{ref}}}{r} \cdot \frac{n+\delta}{m}$

Figure 13.102. Getting to fractional-n. A–C. Integer-n with input prescaler, output scaler, and both. D. "Fractional-n" allows the feedback divider to effectively take on non-integer values. For simplicity the loop filter between phase detector and VCO has been omitted.

Input prescaler

Taking it in several steps (see Figure 13.102A), the first thing you might do is to reduce the reference frequency to the resolution step size, say 1 Hz. This can be done by "prescaling" the reference input frequency with a modulo-r counter: r is an integer, chosen so that $f_{\text{comp}}=f_{\text{ref}}/r$ equals the desired step size; for example, if we have a 10 MHz input reference (a common standard) and we want 1 Hz settability, we would choose $r=10^7$. The output frequency is then $f_{\text{out}}=n \cdot f_{\text{ref}}/r$.

OK, that would work. But the phase detector is now working on a pair of 1 Hz signals, which requires a very long loop time constant (many seconds). That's not good: it takes a long time to lock to a new frequency setting; and there's more phase noise, because the VCO's intrinsic instabilities are not corrected on short time scales (no loop gain at those frequencies). And (if you need more convincing), the phase detector's correction pulses to the VCO are at low frequency, producing spurious modulation sidebands ("spurs") that are close in frequency to the desired output frequency (to be precise, they are spaced apart by f_{comp}, going both up and down from f_{out}).

Output scaler

The next thing you might try is to keep a high reference frequency, but divide the *output* frequency instead (Figure 13.102B). The output frequency is now $f_{\text{out}}=f_{\text{ref}} \cdot n/m$. This looks pretty good: we can keep plenty of loop bandwidth (because the phase detector is operating at the high-frequency f_{ref}), and we get as small a step size as we want by choosing a large output divider modulus m.

And this works fine, as long as you're happy with low output frequencies. The problem is that the VCO now has to run at m times higher frequency to generate a given f_{out}. For example, with a 10 MHz reference input and with $m=10^7$ (for 1 Hz step size), the VCO would have to run at 1 GHz just to generate a 100 Hz output frequency (n=100). This is clearly a loser.

Input and output scalers

A compromise is in order: use integer frequency dividers at both input *and* output (Figure 13.102C). This way we can set the phase detector's comparison frequency somewhere between the (too small) output step size and the (larger than necessary) input reference frequency. The output frequency is now $f_{\text{out}}=(f_{\text{ref}}/r) \cdot (n/m)$. This is the standard configuration of an "integer-n" phase locked loop (because all three frequency dividers operate with an integer division ratio).

Taking our standard example with 10 MHz reference, we might choose $r=10^4$ (so f_{comp}=1 kHz) and $m=10^3$. The output step size is 1 Hz, the output frequency is n Hz, and we can generate output frequencies to 100 kHz (with 1 Hz resolution) with a VCO that goes to 100 MHz.

The whole enchilada: "fractional-n"

We've got a compromise among competing factors of step size, loop bandwidth, maximum output frequency, and maximum VCO frequency. In the example above we can get to a higher output frequency with the same 1 Hz step size and 10 MHz input frequency, (i.e., keeping the product $m \cdot r$ constant), but only at the expense of either smaller loop bandwidth (smaller m, larger r) or reduced maximum output frequency (larger m, smaller r).

Can we do better? The answer is yes, if we can somehow trick one of the dividers (say the n divider) into a non-integer "in-between" division ratio. We can do that *on the average* with an integer-n divider, if we arrange things to change the modulus so that it spends some of its time as n, and the rest of its time as n+1.[123] This is *fractional-n* synthesis (Figure 13.102D). The output frequency is still $f_{\text{out}}=(f_{\text{ref}}/r) \cdot (n/m)$, but with n now permitted to take on a fractional value. With fractional-n synthesis you have (mostly) the best of both worlds: wide output-frequency range with high resolution (small step size), while retaining high f_{comp} (which permits plenty of loop bandwidth, and therefore fast lock and fast tracking, along with spurs widely offset from the synthesized frequency).

Fractional-n requires some additional counters and logic, to figure out how often to alternate between n and n+1. There's an analogous everyday situation: it's beneficial to keep the annual calendar (the kind you hang on a wall) synchronized to Earth's motion around the sun. Trouble is, there are not an integral number of days in a year. The Gregorian calendar's solution (leap years) is fractional-n: alternate the modulus between 365 and 366, with a 3:1 ratio, to achieve the (approximately) correct value of 365.25.[124,125]

Details, details...

Fractional-n synthesis[126] is a nice technique, but it's not without its own problems. For example, the phase detector is periodically presented with a phase discontinuity (i.e., each time the modulus is alternated), which creates periodic phase modulation at the output if not corrected or otherwise filtered. There are several tricks to fix this problem, involving either injecting compensating charge pulses at the phase detector's charge-pump output, or (probably better) a precomputed correction to the output waveform to create equally spaced output cycles (see later). Perhaps the best technique, though, is the use of delta–sigma

[123] You sometimes hear the visually graphic term "pulse-swallowing" used for this subcircuit.

[124] Astronomers, and other fussy readers, will complain that the number of Earth rotations in a year is actually one greater (366.25), and that Earth rotates once *not* in 24 hours, but in 23h 56m 4s (approximately). Of course they are right. But everyone hates a smarty-pants.

[125] That gets the calendar close, but not perfect: there are 365.242374 solar days in a year. Hence the next-order fix: leap-years are *omitted* in years that are divisible by 100 (i.e., the turn of each century, which, being divisible by 4, would ordinarily be a leap year) unless they are also divisible by 400; we had a leap year in 2000, but we (more properly, our descendants; by then this book, and its authors, will be out of print) will not enjoy one in 2100, or 2200, or 2300. This will keep the calendar on track for about 8000 years.

[126] Some PLLs do the fractional-n business on the input reference (r) divider; they're still called "fractional-n."

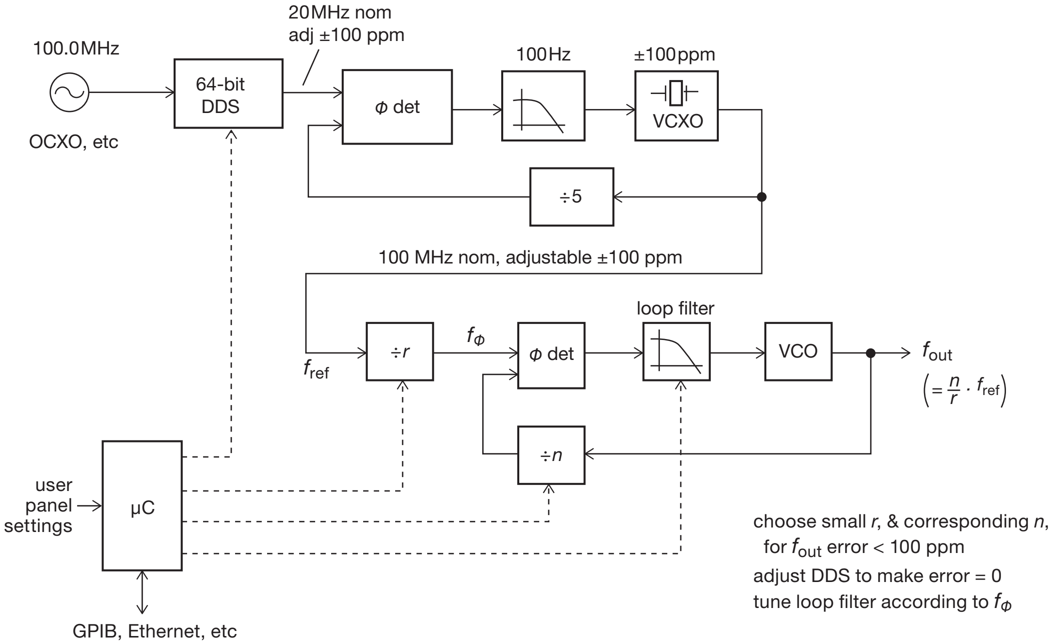

100.0MHz
OCXO, etc
64-bit DDS
20MHz nom
adj ±100 ppm
Φ det
100Hz
±100ppm
VCXO
÷5
100 MHz nom, adjustable ±100 ppm
f_{ref}
÷r
$f_Φ$
Φ det
loop filter
VCO
f_{out}
$\left(= \frac{n}{r} \cdot f_{ref}\right)$
÷n
user panel settings
μC
GPIB, Ethernet, etc
choose small r, & corresponding n, for f_{out} error < 100 ppm
adjust DDS to make error = 0
tune loop filter according to $f_Φ$

Figure 13.103. By making small frequency adjustments to its clean crystal-oscillator reference clock, the method of "rational approximation frequency synthesis" used in the SRS synthesizers achieves microhertz resolution while operating its PLL phase-detector at megahertz frequencies. The resulting output has excellent purity, with very low phase noise and absence of spectral "spurs."

modulation of the modulus: instead of simply alternating between the two moduli that surround the (fractional) desired modulus, the divider's modulus is distributed among a larger set, in such a way that the production of modulation sidebands is shaped to higher frequencies, and the production of discrete spurs is minimized. As with the delta–sigma modulators we saw earlier in the chapter, higher-order loops along with some randomization ("dithering") can be employed to reduce close-in spurs (analogous to its use there to suppress idle tones). This is a complicated business, and best left to the real professionals.[127] Bottom line: leave the converter design to others; but be aware of both the benefits and pitfalls, and examine the datasheets closely for the things you care about in your application.

B. Rational approximation synthesis

With an ingenious variation on integer-*n* synthesis, the ever-creative John Willison at Stanford Research Systems has devised a synthesizer that combines the best of both worlds: it operates with a small integer *r* value (thus a relatively high reference frequency input to the phase detector, for wide VCO loop bandwidth and thus low noise and jitter sidebands), along with integer *n* (avoiding VCO phase modulation); but, with a bit of magic, it permits essentially infinite frequency resolution (*micro*hertz frequency setting) even though the phase detector's reference input is typically in the megahertz region.

How can this be? The trick is to choose a small integer *r* (and corresponding *n*) such that the synthesized frequency is *close to* (say within ±100 ppm of) the target frequency; you then adjust the master reference oscillator correspondingly, to bring the integer-synthesized output frequency on target. This technique, which they call "rational approximation frequency synthesis" (RAFS), was introduced in SRS's SG380 series of RF signal generators, which currently include models that provide outputs from dc to 6 GHz, settable with microhertz resolution. The use of integer PLL synthesis with a wide-bandwidth loop produces excellent output purity, seen for example in the phase noise specification of −116 dBc (relative to the "carrier," i.e., the signal amplitude) at an offset of 20 kHz from a 1 GHz output signal; and the use of a low-noise reference oscillator (the OCXO) keeps the close-in phase noise down to an impressive −80 dBc at an offset of just 10 Hz from a 1 GHz output.

Figure 13.103 shows the scheme, stripped down to its

[127] Take a look at National Semiconductor's App Note 1879, if you want to dip your toe into the turbulent waters.

basics. A microcontroller rides herd over the system, beginning with the choice[128] of r and n to get close to the desired $f_{\rm out}$. The microcontroller also tunes the loop filter according to the resulting phase detector input frequency (here called f_ϕ). Finally, it fine-tunes the master clock, over the needed ±100 ppm range, by way of a 64-bit (thus effectively infinite resolution) direct digital synthesizer running from the clean fixed-frequency input clock. The DDS output is not as pure as its reference input (owing to irregular phase jumps inherent in the DDS process), so its output is cleaned up by phase locking a high quality crystal oscillator, whose frequency can be electrically "pulled" over a ±100 ppm range via a varactor (thus a "VCXO" – voltage-controlled crystal oscillator). With hindsight we can see that it's the VCXO's tuning range that constrains the initial selection of r and n.

To illustrate with an example, suppose we wish to synthesize an output at 1234.56789 MHz. You can bang away on a pocket calculator (does anyone under 50 still use those?), working your way up successive integer r values, until you find that r=26 gives you a "fractional-n" (320.9876514) that is within 100 ppm of an integer (n=321). So we go with the choice $[r,n]$=[26, 321], and we offset the master clock by −38.469 ppm (to 99.9961531 MHz) to get what we want. With this choice the phase detector's reference frequency f_ϕ is pleasantly high (∼3.85 MHz), allowing plenty of loop bandwidth (thus low sideband noise, and absence of close-in spurs that would be caused by a low f_ϕ) in the synthesizing PLL.

In practice there are a host of details (as with any sophisticated and well engineered system) not seen in this simplified description. For example, (a) the output synthesizer tunes only over an octave (2:1 frequency range), driving a set of binary dividers and lowpass filters to generate the final output; (b) the actual production instruments use several staggered fine-tuned VCXOs, greatly relaxing the constraints on r and n (and resulting in values of f_ϕ typically greater than 10 MHz, with a worst case of 2.4 MHz); (c) there are additional DDSs and PLLs in the system, used among other things to create favorable clocking frequencies (which generally are chosen not to be the kind of "round-number" frequencies shown here, to prevent clock-collision artifacts); (d) the 64-bit DDS is dithered to reduce fixed-frequency spurious sidebands ("spurs"); and (e) there are additional subcircuits to provide for modulation, amplitude control, and the like. These are the sort of real-world issues that challenge the instrument designer, to whom great satisfaction can come when a nice solution is found.

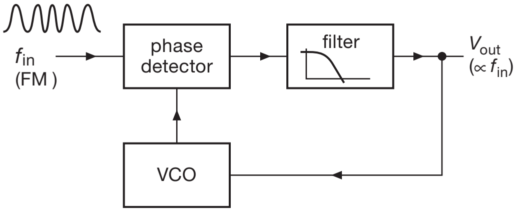


Figure 13.104. PLL FM discriminator.

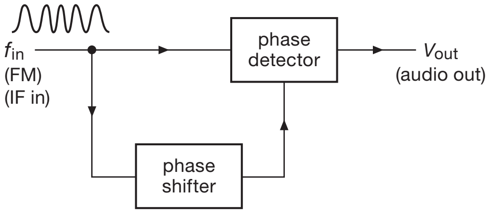


Figure 13.105. Quadrature FM detection.

C. FM detection

In frequency modulation, information is encoded onto a "carrier" signal by varying its frequency proportional to the information waveform. There are two methods of recovering the modulating information with phase detectors or PLLs. The word *detection* is used to mean a technique of demodulation.

In the simplest method, a PLL is locked to the incoming signal. The voltage controlling the VCO frequency is proportional to the input frequency and is therefore the desired modulating signal (Figure 13.104). In such a system you would choose the filter bandwidth to be wide enough to pass the modulating signal, i.e., the response time of the PLL must be short compared with the time scale of variations in the signal being recovered.[129] A high degree of linearity in the VCO is desirable in this method of FM detection, to minimize distortion in the audio output.

The second method of FM detection involves a phase detector, although not in a phase-locked loop. Figure 13.105 shows the idea. Both the input signal and a phase-shifted version of the signal are applied to a

[128] SRS uses an iterative approach for calculating r and n: the microcontroller chunks along, trying successive small-integer pairs until it's happy. This takes about a millisecond.

[129] The signal applied to the PLL does not have to be at the radiofrequency sent by the distant transmitter; it can be an "intermediate frequency" (IF) generated in the receiving system by the process of *mixing*. This *superheterodyne* technique was invented by Edwin H. Armstrong, who also invented FM. The big guys moved in, claimed his inventions, beat him up in court, and drove him to suicide.

phase detector, generating some output voltage. The phase-shifting network is diabolically arranged to have a phase shift varying linearly with frequency in the region of the input frequency (this is usually done with resonant *LC* networks), thus generating an output voltage with linear dependence on input frequency. That is the demodulated output. This method is called doubly balanced quadrature FM detection, and it is used in some IF amplifier/detector ICs.

Lest we leave the wrong impression, we hasten to add that you can demodulate FM without the help of phase locked loops. The classic techniques exploit the steep amplitude versus frequency characteristic of *LC* tuned circuits. In its simplest form (a "slope detector"), the FM signal is applied to an *LC* resonant circuit tuned off to one side, so it has a rising curve of response versus frequency; the output amplitude then varies approximately linearly with frequency, turning FM into FM+AM. An AM envelope detector completes the job of converting the AM to audio. In practice, a slightly more complicated arrangement (called a ratio detector, or Foster–Seeley detector) is used. Another (and simpler) technique uses averaging of a train of identical pulses at the intermediate frequency.

D. AM detection

Wanted: a technique to give an output signal proportional to the instantaneous *amplitude* of a high-frequency signal. The usual method involves rectification (Figure 13.106). Figure 13.107 shows a fancy method ("homodyne detection," or "synchronous detection") using PLLs. The PLL generates a square wave at the same frequency as the modulated carrier. Multiplying the input signal by this square wave generates a full-wave-rectified signal that only needs some lowpass filtering to remove the remnants of the carrier frequency, leaving the modulation *envelope*. If you use the exclusive-OR type of phase detector in the PLL, the output is shifted 90° relative to the reference signal, so a 90° phase shift would have to be inserted in the signal path to the multiplier.

E. Digital demodulation

A phase locked loop is an essential component in recovering ("demodulating") data from a carrier that has been modulated with a *digital* signal. In a simple form of digital modulation ("binary phase-shift keying," or BPSK), each bit to be transmitted either inverts, or not, the phase of a constant-amplitude carrier (Figure 13.108). These encoded bits are recovered at the receiving end by multiplying the received BPSK-modulated carrier by a signal at the same carrier frequency. Your first thought might be to use a PLL to recover a carrier replica. But that doesn't work, because the BPSK-modulated spectrum has no component at the carrier frequency.

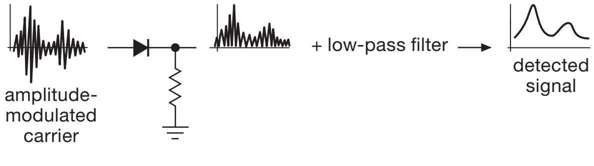


Figure 13.106. AM detection.

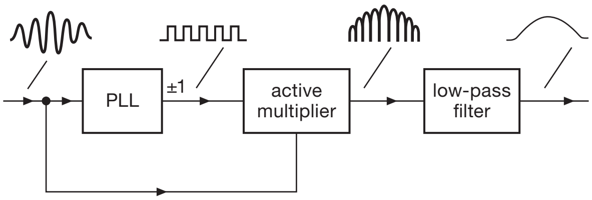


Figure 13.107. Homodyne detection.

One cute solution[130] is to notice that the *square* of the transmitted signal ignores the phase reversals, generating a signal at twice the carrier frequency. Pursuing that idea, you get the "squaring-loop" method of Figure 13.108. The first mixer M1 (a mixer is a multiplier) generates the doubled carrier frequency $2f_c$, which is cleaned up with a bandpass filter and used to lock a PLL, with the VCO acting as a flywheel (low loop bandwidth); a divide-by-2 then creates the carrier replica at f_c, with a phase trim to bring it into alignment with the (suppressed) underlying received carrier. Finally, multiplier M3 synchronously recovers the modulating bits, with a final lowpass filter to remove the $2f_c$ ripple.

If the phased bursts of cycles are thought of as *symbols*, BPSK encodes one bit per symbol. Commonly used digital-modulation schemes typically encode several bits per symbol. For example, you could encode symbols of 2 bits each by sending bursts of carrier cycles, each phased 0°, 90°, 180°, or 270°, according to the 2-bit symbol. This is called quadrature phase-shift keying (QPSK), also known as 4-QAM ("quadrature amplitude modulation," pronounced "quam"). More generally, you can create a "constellation" of symbols, each a (tapered) burst with some amplitude and phase. For example, cable television is commonly delivered as 256-QAM, each symbol carrying 8 bits of information. For all these modulation schemes you still need to recover a signal at the carrier frequency (or

[130] There's a more subtle method of BPSK demodulation, again using a PLL, known as a "Costas Loop." Its performance is comparable, but it's harder to understand what's going on. We like simplicity.

its frequency-shifted replica, an “intermediate frequency”), for which a PLL is essential. A trick that is sometimes used is to transmit a weak “pilot” signal at the carrier frequency, so that schemes like the squaring loop aren’t needed. This is used, for example, in digital television broadcasting in the United States, where 3-bit symbols are encoded as amplitude modulation (four amplitude levels, at 0° or 180°), with a slight dc offset to create the pilot to which the receiver’s PLL can lock.

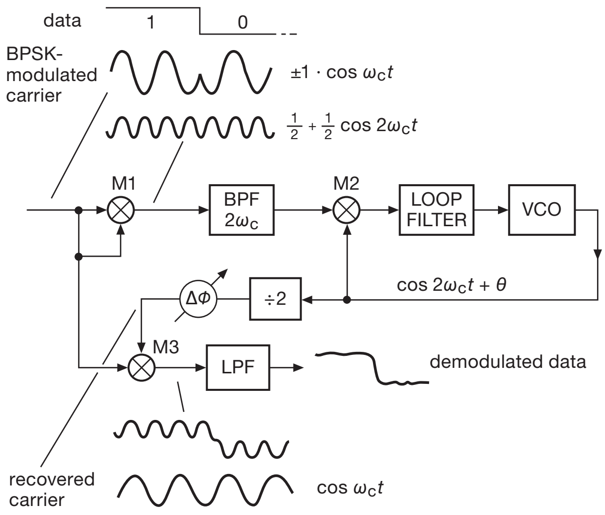


Figure 13.108. Squaring-loop demodulation of BPSK digital signal.

F. Other communications applications

As we suggested earlier, PLLs play an essential role in many aspects of communications. Channelized transmitters (think cellphones) must keep their signals at defined frequencies, with sufficient signal purity to prevent out-of-channel interference. And receivers (cellphones again; or FM radios, televisions, satellite receivers) use a *local oscillator* (LO) to determine their receiving frequency (that’s Armstrong’s superheterodyne technique, nearly a century old). Signal impurity (jitter, spurs) in the LO cause degradation of the received signal, in just the way that it would if it were the transmitter. For applications like these, signal quality is paramount and requires better VCOs than you get with simple capacitive charging circuits like those in the ’4046.

For this kind of application you can get PLL chips that are intended for use with external VCOs and do not include an on-chip oscillator; examples are the NSC LMX2300-series, or the compatible ADI ADF4116–18. These families include members with phase detectors that can run to 6 GHz and beyond. With such PLL chips you can use any commercial VCO; or you can make your own (e.g., a JFET *LC* oscillator, electrically tuned with a varactor, §1.9.5B). An example of the latter is the PLL-disciplined JFET oscillator of Figure 7.29, with a noise spectrum as shown in Figure 7.30.

There’s been considerable effort recently aimed at integrating high-quality VCOs directly onto the PLL chip, so you don’t have to rig up a separate oscillator. Some of these need an external inductor (the hardest part to integrate with the required inductance, and with sufficient quality factor *Q*), for example the ADF4360-8. Others include all components on-chip, for example the LMX2531 or ADF4360-3; these latter are intended for cellphone use and have relatively narrow VCO oscillator tuning ranges, of order 5%. Other technologies used for on-chip oscillators include silicon micromachined (MEMS) resonators (e.g., the SiTime SiT3700-, 8100-, 9100-series), and surface acoustic wave (SAW) resonators (e.g., the IDT M680-series). These have very narrow VCO tuning ranges (∼100 ppm), but you get very low phase noise and jitter, as you do with the competitive technology of crystal oscillators with a similar narrow range of voltage tuning (a VCXO; used, for example, within the IDT 810252 PLL).

G. Pulse synchronization and clean-signal regeneration

In digital signal transmission, a string of bits containing the information is sent over a communications channel. The information may be intrinsically digital or it may be digitized analog signals, as in “pulse-code modulation” (PCM). A closely related situation is the decoding of digital information from magnetic tape or disk, or optical disk storage. In such cases there may be noise or variations in pulse rate (e.g., from tape stretch), and it is desirable to have a clean clock signal at the same rate as the bits you are trying to read. PLLs work very nicely here. The PLL lowpass filter would be designed to follow rate variations inherent in the data stream (e.g., variations in tape or disk speed), while eliminating cycle-to-cycle jitter and noise that come from less-than-ideal received clock signal quality. This widespread application is often called “clock and data recovery” (CDR). An example in the audio world is the Burr–Brown (TI) DIR9001 Digital Audio Interface Receiver, which includes a low-jitter on-chip PLL/VCO clock-recovery subsystem, along with data demodulation. It is flexibly programmable to handle a wide range of data rates (28–108 ksps) and digital formats (with names like S/PDIF, AES3, IEC60958, and CPR-1205).

Table 13.13 Selected Phase-locked Loops[a]

		Output freq		VCO freq		Ref input freq							Supply					Packages				
													Voltage		Current							
Part #	VCO[b]	min (MHz)	max (MHz)	min (MHz)	max (MHz)	min (MHz)	max (MHz)	@V_S (V)	PD type	Fractional-n Synth	Clock Regen/Distrib	Ext bare XTAL ref	min (V)	max (V)	typical (mA)	Power-down mode	Price qty25	DIP	SOIC	TSSOP	Smaller	Comments
LMC568	iRC	0	f	0	1	0	0.5	5	-	-	-	-	2	9	0.75	-	1.25	-	8	-	-	A
CD4046	iRC	0	0.6[m]	0	0.6[m]	0	0.6[m]	10	1,2	-	-	-	3	18	0.09[g]	q	0.48	16	16	16	-	B
74HC4046	iRC	0	12[m]	0	12[m]	0	12[m]	4.5	1,2,3	-	-	-	3	6	-	q	0.48	16	16	16	-	C
74HCT9046	iRC	0	11[m]	0	11[m]	0	11[m]	4.5	1,2	-	-	-	4.5	5.5	-	q	2.23	16	16	16	-	D
74LV4046	iRC	0	24	0	24	0	24	3.3	1,2,3	-	-	-	3	5.5	-	q	1.00	16	16	16	-	E
74HC7046	iRC	0	38	0	38	0	38	4.5	1,2	-	-	-	2	6	-	q	1.28	16	16	-	-	F
74ACT297	eV	0	55	-	-	0	55	5	1	-	-	-	4.5	5.5	-	-	1.53	-	16	-	-	G
TLC2932	iR	11	50	22	50	-	40	5	2	-	-	-	4.7	5.3	5	•	3.29	-	14	-	-	H
TLC2933	iR	64	96	64	96	-	50	5	2	-	-	-	4.7	5.3	5.7	•	2.88	-	-	14	-	I,K
TLC2934	iR	10	130	10	130	-	50	3.3	2	-	-	-	3.1	3.5	10	•	3.00	-	-	14	-	I
CY22800	int	1	200	-	-	0.5	100	3.3	-	-	•	•	3.1	3.5	-	-	2.45	-	8	-	-	J
ICS673	int	0.25	120	2	240	0.001	8	5	2	-	-	-	4.5	5.5	15	•	4.29	-	16	-	-	L
ADF4360-8	iL	65	400	65	400	10	250	3.3	2	-	-	-	3.0	3.6	25	•	6.02	-	-	-	24	M,S
ADF4110	eV	-	55[o]	50	550	5	104	3	2	•	-	-	2.7	5.5	4.5	•	4.72	-	-	16	20	N,S
IDS810252	iX	25[d]	312.5[d]	625	625	0.008	155	3.3	2	-	-	•	3.1	3.5	225[m]	-	29.00	-	-	-	32	O
SY89421	int/eV	20	1120	480	1120	30	560	5	2	-	•	-	4.7	5.3	28	-	16.00	-	20	-	-	P
LMX2316	eV	-	10[o]	100	1200	-	100	3	2	•	-	-	2.3	5.5	2.5	-	7.00	-	-	16	16	Q,S
CDCE72010	eVX	0	800	< 1	1500	-	500	3.3	2	-	•	-	3.0	3.6	to 880	•	17.96	-	-	-	64	R
AD9510	eVX	0	1200	0	1600	0	250	3.3	2	•	•	-	3.1	3.5	170	•	14.31	-	-	-	64	T
MPC9230	int	50	800	800	1800	10	20	3.3	-	-	-	•	3.1	3.5	110[m]	-	5.84	-	28[p]	32	-	S,U
CDCM61001	int	62	625	1750	2050	21.9	28.5	3.3	2	-	•	•	3.0	3.6	95	•	5.74	-	-	-	32	V
LMX2531	int	553	3132	1106	3132	5	80	3	2	-	-	-	2.8	3.2	38	•	10.44	-	-	-	36	S,W
AD9552	int	50	900	3350	4050	6.6	112	3.3	2	•	-	•	3.1	3.5	149	-	9.40	-	-	-	32	X
ADF4350	int	137	4400	2200	4400	10	105	3.3	2	-	-	-	3.0	3.6	120	•	9.03	-	-	-	36	S,Y
ADF4106	eV	-	100[o]	500	6000	20	300	3	2	•	-	-	2.7	3.3	10	•	4.59	-	-	18	20	Z
ADF41020	eV	-	100[o]	4000	18000	10	400	3	2	•	-	-	2.8	3.2	27	•	15.00	-	-	-	20	S

Notes: (a) sorted approximately by increasing VCO $f_{\max}$. (b) eV - external VCO; eVX - external VCO or VCXO; int/eV - int or eV; iL - internal VCO with external inductor; int - internal VCO, no external components; iR - internal VCO with external resistor; iRC - internal VCO with external R and C; iX - internal VCO with external bare xtal. (d) 25, 125, 156.25, and 312.5MHz only. (f) FM and FSK demod, audio bandwidth. (g) at 10kHz and 10V. (m) min or max. (o) phase detector muxed to output. (p) PLCC. (q) no power-down mode, but quiescent current <1μA.

Comments: A: FM and FSK demod applications. **B:** classic 4000B ("HV") CMOS. **C:** classic HCMOS. **D:** improved 4046, no dead zone. **E:** LVCMOS. **F:** 74HCT also available. **G:** digital 1st-order PLL. **H:** can run at 3V, 14-21MHz. **I:** internal ring oscillator. **J:** $f_{\rm ref}$=8-30MHz with bare xtal; ref can be VCXO; can generate spread spectrum. **K:** can run at 3V, 38-55MHz. **L:** can run at 3.3V, 0.25-100MHz; ICS663 (SOIC-8) lacks power-down and output enable. **M:** wireless local-osc with PLL synth; versions with other freq ranges. **N:** wireless local-osc with integer-*n* PLL synth; 1.2GHz, 3GHz, and 4GHz versions. **O:** 2-stage PLL (VCXO PLL drives PLL multiplier, with input and output dividers, for Gigabit and 10-Gig Ethernet. **P:** can run at 3.3V; complementary PECL outputs; ext osc to 2GHz. **Q:** wireless local-osc with PLL synth; 0.55GHz and 2.8GHz versions. **R:** eight individual dividers, output clock distribution, multiple refs, highly complex. **S:** SPI interface. **T:** eight LVDS outputs, 0.2ps jitter, adjustable delay. **U:** wireless local-osc with PLL synth; diff'l PECL; parallel and SPI interfaces; replaces MC12430. **V:** Ethernet clock generator, etc.; four outputs; <1ps jitter. **W:** wireless local-osc with PLL synth; stable low-noise; p/n selects ±5% freq band. **X:** delta-sigma fractional-*n*; 0.5ps jitter; includes xtal osc. **Y:** wireless local-osc with PLL synth; 0.5ps jitter. **Z:** ADF4107=7GHz, ADF4108=8GHz.

H. Clock generators

As we remarked earlier, there are plenty of applications crying out for a suite of standard clock frequencies, synthesized from a single oscillator input, in which the niceties of low phase noise and spurs, etc., are of less concern than minimum parts count and the ability to program among a few standard frequencies. See Table 13.13. An example is the IDT 8430S010i: it's a single chip PLL with multiple synthesized outputs intended for embedded processor applications. You connect a single 25 MHz crystal, and out comes (a) choice of two processor clock frequencies, (b) choice of four PCI or PCIe clock frequencies, (c) choice of four DDR DRAM clock frequencies, (d) gigabit Ethernet MAC clock and PHY clocks, and (e) choice of three SP14.2 link frequencies.

Parts like this may use a simple interchip SPI-like serial programming protocol, or they may be pin-strappable (as this one is). Or they may allow both, as for example

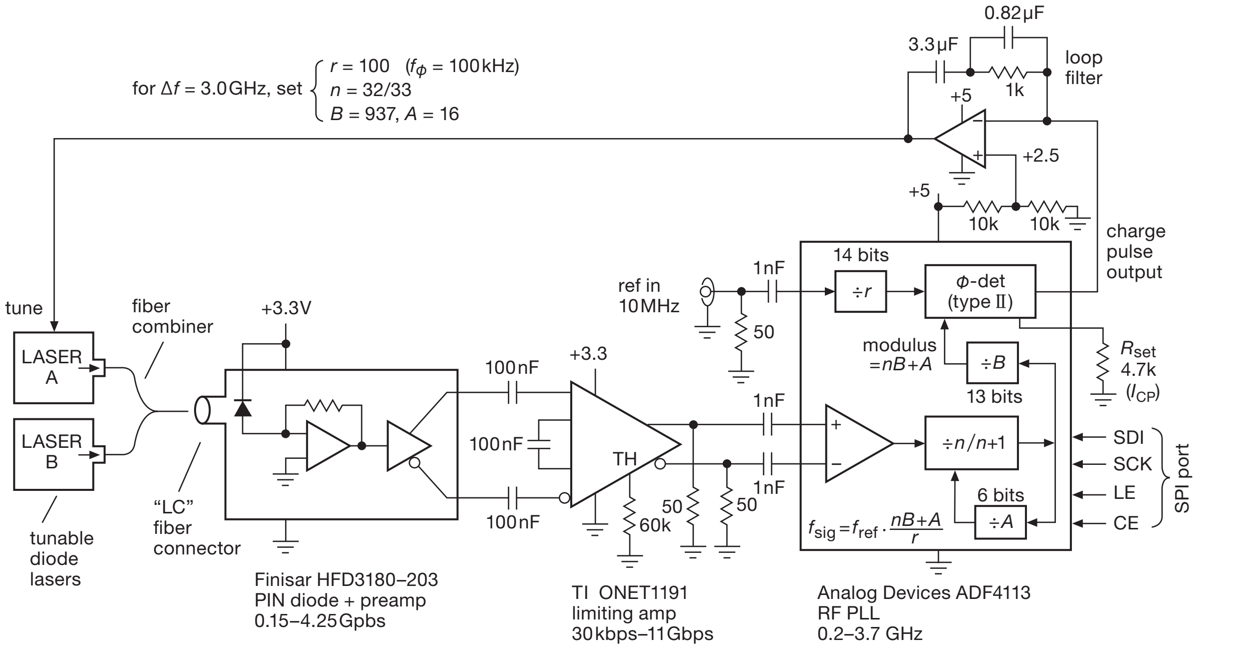


Figure 13.109. Controlling a diode laser to maintain a desired optical frequency difference relative to a reference laser. The components in this circuit cost less than $40, exclusive of the lasers; the latter are in a completely different ballpark, roughly "40 dB$."

the venerable NBC/MC12430 (or equivalent MPC9230), a simple integer-*n* PLL with a 9-bit *n*-counter and a 3-bit *m*-counter, programmable from 50–800 MHz. The on-chip VCO tunes from 400–800 MHz, and likely uses a starved inverter chain ring oscillator (the datasheet isn't talking). We used these as a clocking source in a quirky terasample/second data-acquisition instrument we built to detect intentional pulsed laser signals from possible extraterrestrial civilizations (no kidding).

I. Laser offset locking

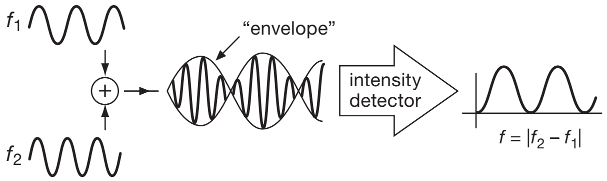
In some scientific applications it's useful to be able to control a tunable laser, such that the frequency of its optical emission is offset by a specific frequency difference from that of a "reference" laser. As a specific example, a favorite technique in the business of "laser cooling" is to subject a beam of atoms to converging beams of laser light at a frequency slightly below that of a natural resonance of the atom. The Doppler effect causes an atom moving toward one of these lasers to see the light shifted upward in frequency, therefore more strongly absorbed by the atom, which is slowed by the transferred momentum.[131]

A phase-locked loop is perfect for this *offset locking*. Figure 13.109 shows how this goes, as implemented in a colleague's laboratory.[132] A portion of the light from the pair of tunable diode lasers is combined and sent to a wideband PIN-diode detector/amplifier module. What happens there is interesting; let's take it in two steps. (a) In a completely linear process, the two combined laser beams create a waveform consisting of a sinewave at the average laser frequency, modulated (multiplied) by a sinewave at half the difference frequency (Figure 13.110). (b) The detector cannot follow the *optical* waves, whose frequency is $\sim 10^{14}$ Hz. It responds only to the *intensity* of the light, proportional to the square of the "envelope" shown in Figure 13.110. And the square of a sinewave is just a sinewave at twice the frequency, plus a dc offset so that it sits atop the horizontal axis as shown.

In other words, at the output of the detector module you get a signal at the lasers' difference frequency (also called the *beat frequency*): $f_{\text{PDout}} = |f_2 - f_1| \equiv \Delta f$. The function of the rest of the circuit is simply to feed back a control signal to laser A so that the difference frequency Δf equals the desired offset. That's done with a fractional-*n* PLL, here preceded by a limiting amplifier that creates a clean

[131] This is known by the colorful term "optical molasses." Add some magnetic fields, and a few more tricks, and you've got a magneto-optical trap.

[132] The ever-capable Dr. Andrew Speck, to whom we are grateful for this and other excellent suggestions.



$$\cos \omega_1 t + \cos \omega_2 t = 2 \cos\left(\frac{\omega_1 - \omega_2}{2}\, t\right) \cos\left(\frac{\omega_1 + \omega_2}{2}\, t\right)$$

Figure 13.110. A linear combination of two sinewaves produces a wave at the midfrequency with a sinusoidal amplitude "envelope." A photodetector cannot capture the optical frequencies themselves ($\sim 10^{14}$ Hz); it responds to the intensity (the square of the envelope), producing an output "beat frequency" equal to the lasers' difference frequency.

saturated 0.6 Vpp signal from a detector output anywhere between 10 mV and 1 V.

For the laser cooling/trapping application, the frequency offset Δf is in the range of 10 MHz or so, relative to the much higher optical resonance frequency; for rubidium atoms the latter is 3.85×10^{14}Hz, corresponding to a wavelength of 780.24 nm.[133] As is often the case, there's much more to love (and hate) in this business: it turns out that there's a "hyperfine splitting" of the ground state of ^{85}Rb, so you need to kick out (the polite term is "optically pump") the atoms that happen to fall all the way to the bottom with some laser light that is offset by that energy difference. That's about 3 GHz,[134] which is why this circuit was designed for offsets in the gigahertz range, as indicated by the notations in the figure.

13.13.7 Wrapup: noise and jitter rejection in PLLs

We've seen applications where the reference is the higher-quality signal (e.g., multiple clocking signals derived from a single stable crystal reference), and applications where the opposite is true, namely that the PLL-generated signal is cleaner than the reference (e.g., clock recovery in a noisy channel, where the "flywheel" action of the VCO cleans up the output).

It's helpful, when thinking about these kinds of issues while setting loop bandwidths and the like, to understand how noise or jitter that originates in different places (reference input, phase detector, or VCO) is filtered by the action of the PLL. You can write lots of equations at this point. But it's not hard to get an intuitive understanding by simply looking at the loop diagram (Figure 13.85): (a) jitter at the reference input is *lowpass* filtered, because variations within the PLL loop bandwidth are tracked by the VCO, whereas fast variations are ignored by the VCO flywheel; (b) intrinsic jitter in the VCO itself is *highpass* filtered, because variations within the loop bandwidth are detected and removed by the loop; and (c) jitter that is introduced by the phase detector is *bandpass* filtered, because slow variations (within the loop bandwidth) are detected and removed, and fast variations are suppressed by the (lowpass) loop filter and the integrating action ($f\rightarrow\phi$) of the VCO.

So, for example, a PLL with clean reference input benefits from a wide loop bandwidth, whereas a PLL presented with an intrinsically stable reference that has acquired additive noise in transmission will benefit from a narrow loop bandwidth (and an intrinsically clean VCO). And the "noise" can be more subtle: the divided VCO signal seen by the phase detector in a fractional-*n* PLL has jitter (introduced by the deliberate changes of modulus); a narrow loop bandwidth smooths this jitter source as well.

Of course, if the PLL output needs agility (as with tone decoding, or FM demodulation) then the loop bandwidth must be tailored accordingly, quite independent of noise and jitter tradeoffs.

13.14 Pseudorandom bit sequences and noise generation

13.14.1 Digital-noise generation

An interesting blend of digital and analog techniques is embodied in the subject of pseudorandom bit sequences (PRBSs).[135] It turns out to be remarkably easy to generate sequences of bits (or words) that have good randomness properties, i.e., a sequence that has the same sort of probability and correlation properties as an ideal coin-flipping machine. Because these sequences are generated by standard deterministic logic elements (shift registers, to be exact), the bit sequences generated are in fact predictable and repeatable, although any portion of such a sequence looks for all the world just like a random string of 0s and 1s.

[133] This happens to be the wavelength used in compact disc recorders; so the system of Figure 13.109 was implemented economically by using a pair of these ubiquitous laser diodes, which can put out 100 mW (caution, definitely *not* eye-safe!). To tune these things, you attach a piezo-tiltable external grating, adjusting the diode current in tandem to keep its wavelength tracking that of the external cavity. Less inventive souls can throw money at the problem: you can buy tunable diode lasers from companies like New Focus, ThorLabs, and Toptica.

[134] Actually, 3.035732439 GHz, if you really need to know.

[135] This is the example we used in Chapter 11 to illustrate programmable logic, where we contrasted PRBS implementations in discrete logic, in programmable logic, and in a microcontroller. See also its use as an analog noise generator in §8.12.4A.

With just a few chips you can generate sequences that literally go on for centuries without repeating, making this a very accessible and attractive technique for the generation of digital bit sequences or analog noise waveforms. When generating eye diagrams (e.g., Figures 12.132 and 14.33), or testing serial links for bit error rates (BERs), it is common to use a PRBS source. PRBSs are used also to "scramble" (deterministically) the serial data in gigabit Ethernet communications, in order to generate a lively bit pattern for the ac-coupled (transformer) physical link; the scrambling is reversed at the receive end, by XOR-ing with a synchronized PRBS running the same sequence.

A. Analog noise

Simple lowpass filtering of the output bit pattern of a PRBS generates bandlimited white Gaussian noise, i.e., a noise voltage with a flat power spectrum up to some cutoff frequency (see Chapter 8 for more on noise). Alternatively, a weighted sum of the shift-register contents (via a set of resistors) performs *digital filtering*, with the same result. Flat noise spectra out to several megahertz can easily be made this way. As you will see later, such digitally synthesized analog noise sources have many advantages over purely analog techniques such as noise diodes or resistors.

B. Other applications

Besides their obvious applications as analog or digital noise sources, pseudorandom bit sequences are useful in a number of applications that have nothing to do with noise. As just mentioned, they are used for pattern generation in serial link testing (eye diagrams, bit error rate), and for bit scrambling (as opposed to real encryption) in serial network protocols like Ethernet. They are used in "direct-sequence" spread-spectrum digital communications (in which each bit to be transmitted is sent as a predetermined sequence of shorter "chips"); such a technique is used, for example, in CDMA (code-division multiple-access) cellular phone systems, and in the airlink privacy cipher of the GSM cellular standard. They're also used in digital TV broadcasting. These sequences are used extensively in error-detecting and error-correcting codes, because they allow the transcription of blocks of data in such a way that valid messages are separated by the greatest "Hamming distance" (measured by the number of bit errors). Their good autocorrelation properties make them ideal for radar-ranging codes, in which the returned echo is compared (cross-correlated, to be exact) with the transmitted bit string. They can even be used as compact modulo-n dividers.

13.14.2 Feedback shift register sequences

The most popular (and the simplest) PRBS generator is the linear feedback shift register (LFSR, Figure 13.111). A shift register of length m bits is clocked at some fixed rate, f_0. An exclusive-OR gate generates the serial input signal from the exclusive-OR combination of the nth bit and the last (mth) bit of the shift register. Such a circuit goes through a set of states (defined by the set of bits in the register after each clock pulse), eventually repeating itself after K clock pulses; i.e., it is cyclic with period K.

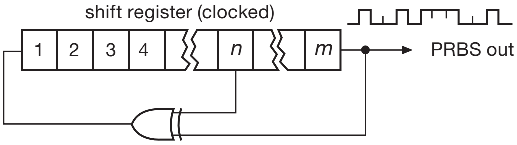


Figure 13.111. Pseudorandom bit sequence generator.

The maximum number of conceivable states of an m-bit register is $K = 2^m$, i.e., the number of binary combinations of m bits. However, the state of all 0s would get "stuck" in this circuit, because the exclusive-OR would regenerate a 0 at the input. Thus the maximum-length sequence you can possibly generate with this scheme is 2^m-1. It turns out that you can make such "maximal-length shift-register sequences" if m and n are chosen correctly, and the resultant bit sequence is pseudorandom.[136] As an example, consider the 4-bit feedback shift register in Figure 13.112. Beginning with the state 1111 (we could start anywhere except 0000), we can write down the states it goes through:

1111
0111
0011
0001
1000
0100
0010
1001
1100
0110
1011
0101
1010
1101
1110

We have written down the states as 4-bit numbers

[136] The criterion for maximal length is that the polynomial $1+x^n+x^m$ be irreducible and prime over the Galois field GF(2).

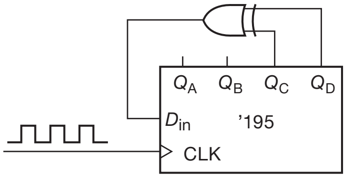


Figure 13.112. 4-bit linear feedback shift register (m=4, n=3; 15 states).

$Q_AQ_BQ_CQ_D$. There are 15 distinct states (2^4-1), after which it begins again; therefore it is a maximal-length register.

Exercise 13.8. Demonstrate that a 4-bit register with feedback taps at the second and fourth bits is not maximal length. How many distinct sequences are there? How many states within each sequence?

A. Feedback taps

Maximal-length shift registers can be made with exclusive-OR feedback from more than two taps (in these cases you use several exclusive-OR gates in the standard parity-tree configuration, i.e., modulo-2 addition of several bits). In fact, for some values of m, a maximal-length register can be made only with more than two taps. Table 13.14 is a listing of all values of m up to 167 for which maximal-length registers can be made with just two taps, i.e., feedback from the nth bit and the mth (last) bit, as previously. A value is given for n and for the cycle length K, in clock cycles. In some cases there is more than one possibility for n, and in every case the value $m-n$ can be used instead of n; thus the earlier 4-bit example could have used taps at n=1 and m=4. Since shift-register lengths in multiples of 8 are common, you may want to use one of those lengths. In that case, more than two taps are necessary. Table 13.15 gives the magic numbers.

It’s rarely necessary to use a register much longer than 32 bits: when clocked at 1 MHz the repeat time is about an hour. Go to 64 bits and you can clock at 1 GHz for six centuries before it comes around again.

B. Properties of maximal-length shift-register sequences

We generate a string of pseudorandom bits from one of these registers by clocking it and looking at successive output bits. The output can be taken from any position of the register; it is conventional to use the last (mth) bit as the output. Maximal-length shift-register sequences have the following properties:

Table 13.14 Single-tap LFSRs

m	*n*	*length*	*m*	*n*	*length*	*m*	*n*	*length*
3	2	7	49	40	5.6e14	108	77	3.2e32
4	3	15	52	49	4.5e15	111	101	2.6e33
5	3	31	55	31	3.6e16	113	104	1.0e34
6	5	63	57	50	1.4e17	118	85	3.3e35
7	6	127	58	39	2.9e17	119	111	6.6e35
9	5	511	60	59	1.2e18	121	103	2.7e36
10	7	1023	63	62	9.2e18	123	121	1.1e37
11	9	2047	65	47	3.7e19	124	87	2.1e37
15	14	32767	68	59	3.0e20	127	126	1.7e38
17	14	1.3e5	71	65	2.4e21	129	124	6.8e38
18	11	2.6e5	73	48	9.4e21	130	127	1.4e39
20	17	1.0e6	79	70	6.0e23	132	103	5.4e39
21	19	2.1e6	81	77	2.4e24	134	77	2.2e40
22	21	4.2e6	84	71	1.9e25	135	124	4.4e40
23	18	8.4e6	87	74	1.5e26	137	116	1.7e41
25	22	3.4e7	89	51	6.2e26	140	111	1.4e42
28	25	2.7e8	93	91	9.9e27	142	121	5.6e42
29	27	5.3e8	94	73	2.0e28	145	93	4.5e43
31	28	2.1e9	95	84	4.0e28	148	121	3.6e44
33	20	8.6e9	97	91	1.6e29	150	97	1.4e45
35	33	3.4e10	98	87	3.2e29	151	148	2.9e45
36	25	6.9e10	100	63	1.3e30	153	152	1.1e46
39	35	5.5e11	103	94	1.0e31	159	128	7.3e47
41	38	2.2e12	105	89	4.1e31	161	143	2.9e48
47	42	1.4e14	106	91	8.1e31	167	161	1.9e50

Table 13.15 Multiple-of-8 LFSRs

m	*taps*	*length*	*m*	*taps*	*length*
8	4, 5, 6	255	96	47, 49, 94	7.9e28
16	4, 13, 15	64K	104	93, 94, 103	2.0e31
24	17, 22, 23	16M	112	67, 69, 110	5.2e33
32	1, 2, 22	4G	120	2, 9, 113	1.3e36
40	19, 21, 38	1.1e12	128	99, 101, 126	3.4e38
48	20, 21, 47	2.8e14	136	10, 11, 135	8.7e40
56	34, 35, 55	7.2e16	144	74, 75, 143	2.2e43
64	60, 61, 63	1.8e19	152	86, 87, 151	5.7e45
72	19, 25, 66	4.7e21	160	141, 142, 159	1.5e48
80	42, 43, 79	1.2e24	168	151, 153, 166	3.7e50
88	16, 17, 87	3.1e26			

1. In one full cycle (K clock cycles), the number of 1s is one greater than the number of 0s. The extra 1 comes about because of the excluded state of all 0s. This says that heads and tails are equally likely (the extra 1 is totally insignificant for any reasonable-length register; a 17-bit register will produce 65,536 1s and 65,535 0s in one of its cycles).
2. In one full cycle (K clock cycles), half the runs of

consecutive 1s have length 1, one-fourth the runs have length 2, one-eighth have length 3, etc. There are the same numbers of runs of 0s as of 1s, again with the exception of a missing 0. This says that the probability of heads and tails does not depend on the outcome of past flips, and therefore the chance of terminating a run of successive 1s or 0s on the next flip is 1/2 (contrary to the person-in-the-street's understanding of the "law of averages").

3. If one full cycle (K clock cycles) of 1s and 0s is compared with the same sequence shifted cyclically by any number of bits n (where n is not 0 or a multiple of K), the number of disagreements will be one greater than the number of agreements. In fancy language, the autocorrelation function is a Kronecker delta at zero delay, and $-1/K$ everywhere else. This absence of "sidelobes" in the autocorrelation function is what makes PRBSs so useful for radar ranging.

Exercise 13.9. Show that the 4-bit shift-register sequence listed earlier (taps at $n=3, m=4$) satisfies these properties, considering the Q_A bit as the "output": 100010011010111.

13.14.3 Analog noise generation from maximal-length sequences

A. Advantages of digitally generated noise

As we remarked earlier, the digital output of a maximal-length feedback shift register can be converted to band-limited white noise with a lowpass filter whose cutoff frequency is well below the clock frequency of the register. Before getting into the details, we point out some of the advantages of digitally generated analog noise. Among other things, it allows you to generate noise of known spectrum and amplitude (see the circuit example in §8.12.4A), with adjustable bandwidth (via clock-frequency adjustment), using reliable and easily maintained digital circuitry. There is none of the variability of diode noise generators, nor are there interference and pickup problems that plague the sensitive low-level analog circuitry used with diode or resistor noise generators. Finally, it generates repeatable "noise" and, when filtered with a weighted digital filter (more about this later), repeatable noise waveforms independent of clocking rate (output noise bandwidth).

13.14.4 Power spectrum of shift-register sequences

The output spectrum generated by maximal-length shift registers consists of noise extending from the repeat frequency of the entire sequence, f_{clock}/K, up to the clock frequency and beyond. It is flat within ±0.1dB up to 12% of the clock frequency (f_{clock}), dropping rather rapidly beyond its -3 dB point of 44% f_{clock}. Thus a lowpass filter with a high-frequency cutoff of 5%–10% of the clock frequency will convert the unfiltered shift register output to a band-limited analog noise voltage. Even a simple *RC* filter will suffice, although it may be desirable to use active filters with sharp cutoff characteristics (see Chapter 6) if a precise frequency band of noise is needed.

To make these statements more precise, let's look at the shift-register output and its power spectrum. It is usually desirable to eliminate the dc offset characteristic of digital logic levels, generating a bipolarity output with 1 corresponding to $+a$ volts and 0 corresponding to $-a$ volts (Figure 13.113). This can be done in various ways, for example, referring to Figure 13.114, (A) with an 'HC4053 linear CMOS switch, operating from dual supplies and with a pair of inputs tied to $\pm a$ volts; (B) with a fast op-amp with dc offset current into a summing junction; (C) with a fast rail-to-rail comparator running between $\pm a$-volt split supplies; (D) with a CMOS logic gate, powered from $\pm a$-volt supplies, driven by a properly shifted and scaled logic swing; or (E) with the same logic gate, driven with a diode-clamped (and current-limited) logic swing. This last method is a bit weird, and works only if the total supply voltage ($V_{\text{total}}=2a$) is in the range of a standard logic family (say 1–5 V); but it excels in speed. It's not dc-coupled, so it requires a logic input that's busy; that's OK here – a PRBS is peripatetic, and cannot rest for longer than m clock periods.[137]

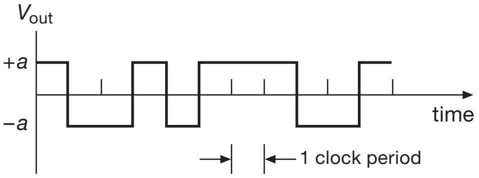


Figure 13.113. A symmetrical PRBS waveform eliminates the dc component.

As we remarked earlier, the string of output bits has a single peak in its autocorrelation. If the output states represent $+1$ and -1, the digital autocorrelation (the sum of the product of corresponding bits, when the bit string is compared with a shifted version of itself) is as shown in Figure 13.115.

[137] To paraphrase Woody Allen, "A clamped ac-coupled logic input is like a shark; it has to keep moving, or it dies."

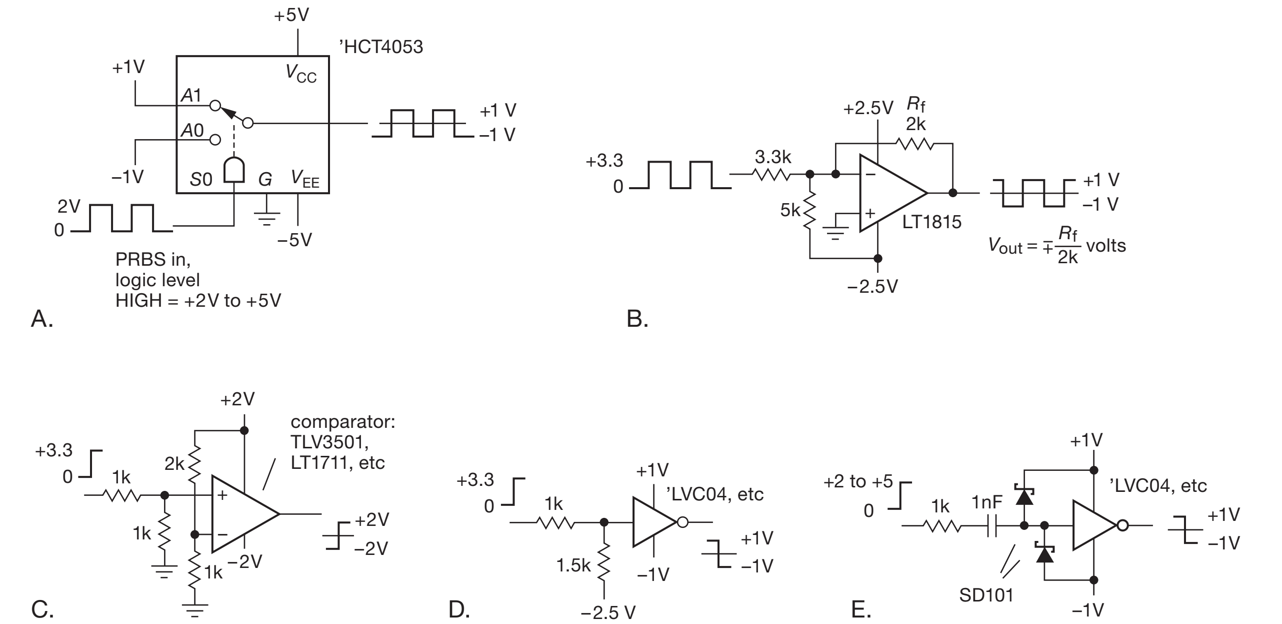


Figure 13.114. Converting a positive-only logic swing to a symmetrical voltage waveform.

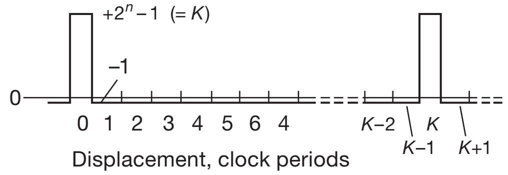


Figure 13.115. Full-cycle discrete autocorrelation for a maximal-length shift-register sequence.

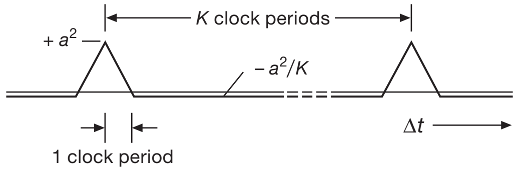


Figure 13.116. Full-cycle continuous autocorrelation for a maximal-length shift-register sequence.

Don't confuse this with a *continuous* autocorrelation function, which we consider later. This graph is defined only for shifts corresponding to a whole number of clock cycles. For all shifts that aren't zero or a multiple of the overall period K, the autocorrelation function has a constant -1 value (because there is an extra 1 in the sequence), negligible when compared with the zero-offset autocorrelation value of K. Likewise, if we consider the unfiltered shift-register output as an *analog* signal (whose waveform happens to take on values of $+a$ and $-a$ volts only), the normalized autocorrelation becomes a continuous function, as shown in Figure 13.116. In other words, the waveform is totally uncorrelated with itself when shifted more than one clock period forward or backward.

The power spectrum of the unfiltered digital output can be obtained from the autocorrelation by standard mathematical techniques. The result is a set of equally spaced series of spikes (delta functions), beginning at the frequency at which the whole sequence repeats, f_{clock}/K, and going up in frequency by equal intervals f_{clock}/K. The fact that the spectrum consists of a set of discrete spectral lines reflects the fact that the shift-register sequence eventually (and periodically) repeats itself. Don't be alarmed by this funny spectrum; it will look continuous for any measurement or application that takes less time than the cycle time of the register. The envelope of the spectrum of the unfiltered output is shown in Figure 13.117. The envelope is proportional to the square of $(\sin x)/x$. Note the peculiar property that there is *no* noise power at the clock frequency or its harmonics.

A. Noise voltage

Of course, for analog noise generation you use only a portion of the low-frequency end of the spectrum. It turns out to be easy to calculate the noise power per hertz in terms of the half-amplitude (a volts) and the clock frequency

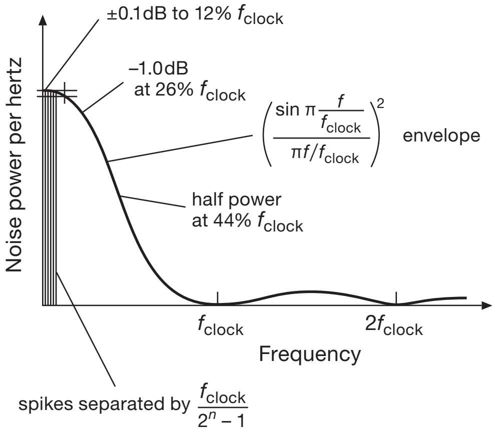


Figure 13.117. Power spectrum of unfiltered digital shift-register output signal.

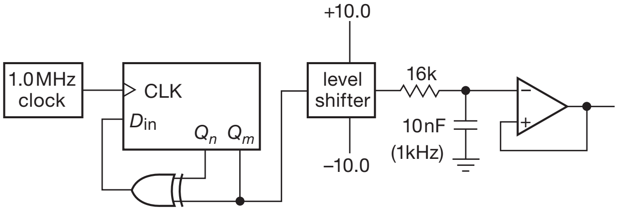


Figure 13.118. Simple pseudorandom noise source.

(f_{clock}). Expressed as an rms noise *voltage*, the answer is

$$v_{\text{rms}} = a\left(\frac{2}{f_{\text{clock}}}\right)^{1/2} \quad \text{V}/\sqrt{\text{Hz}}, \qquad (f \le 0.2 f_{\text{clock}}).$$

This is for the bottom end of the spectrum, the part you usually use (you can use the envelope function to find the power density elsewhere).

For example, suppose we run a maximal-length shift register at 1.0 MHz and arrange it so that the output voltage swings between +10.0 and −10.0 volts. The output is passed through a simple *RC* lowpass filter with 3 dB point at 1 kHz (Figure 13.118). We can calculate the rms noise voltage at the output exactly. We know from the preceding equation that the output from the level shifter has an rms noise voltage of 14.14 mV per root hertz. From §8.13 we know that the noise bandwidth of the lowpass filter is (π/2)(1.0 kHz), or 1.57 kHz. So the output noise voltage is

$$V_{\text{rms}} = 0.01414 \cdot (1570)^{1/2} = 560\,\text{mV}$$

with the spectrum of a single-section *RC* lowpass filter.

13.14.5 Low-pass filtering

A. Analog filtering

The spectrum of useful noise from a pseudorandom sequence generator extends from a low-frequency limit of the reciprocal repeat period (f_{clock}/K) up to a high-frequency limit of perhaps 20% of the clock frequency (at that frequency the noise power per hertz is down by 0.6 dB). Simple lowpass filtering with *RC* sections, as illustrated in the earlier example, is adequate provided that its 3 dB point is set far below the clock frequency (e.g., less than 1% of f_{clock}). To use the spectrum closer to the clock frequency, it is advisable to use a filter with sharper cutoff, e.g., a Butterworth or Chebyshev. In that case the flatness of the resultant spectrum depends on the filter characteristics, which should be measured, since component variations can produce ripples in the passband gain. Likewise, the filter's actual voltage gain should be measured if the precise value of noise voltage per root hertz is important.

B. Digital filtering

A disadvantage of analog filtering is the need to readjust the filter cutoff if the clock frequency is changed by large factors. In situations where you want to change the clocking frequency, an elegant solution is provided by discrete-time digital filtering, in this case performed by taking an analog-weighted sum of successive output bits (nonrecursive digital filtering). In this way the effective filter cutoff frequency changes to match changes in the clock frequency. In addition, digital filtering lets you go to extremely low cutoff frequencies (fractions of a hertz) where analog filtering becomes awkward.

To perform a weighted sum of successive output bits simultaneously, you can simply look at the various parallel outputs of successive shift-register bits, using resistors of various values into an op-amp summing junction. For a lowpass filter the weights should be proportional to $(\sin x)/x$; note that some levels will have to be inverted since the weights are of both signs. Because no capacitors are used in this scheme, the output waveform consists of a set of discrete output voltages.

The approximation to Gaussian noise is improved by use of a weighting function over many bits of the sequence. In addition, the analog output then becomes essentially a continuous waveform. For this reason it is desirable to use as many shift-register stages as possible, adding additional shift-register stages outside the exclusive-OR feedback if necessary. As before, pullups or MOS switches should be used to set stable digital voltage levels (CMOS

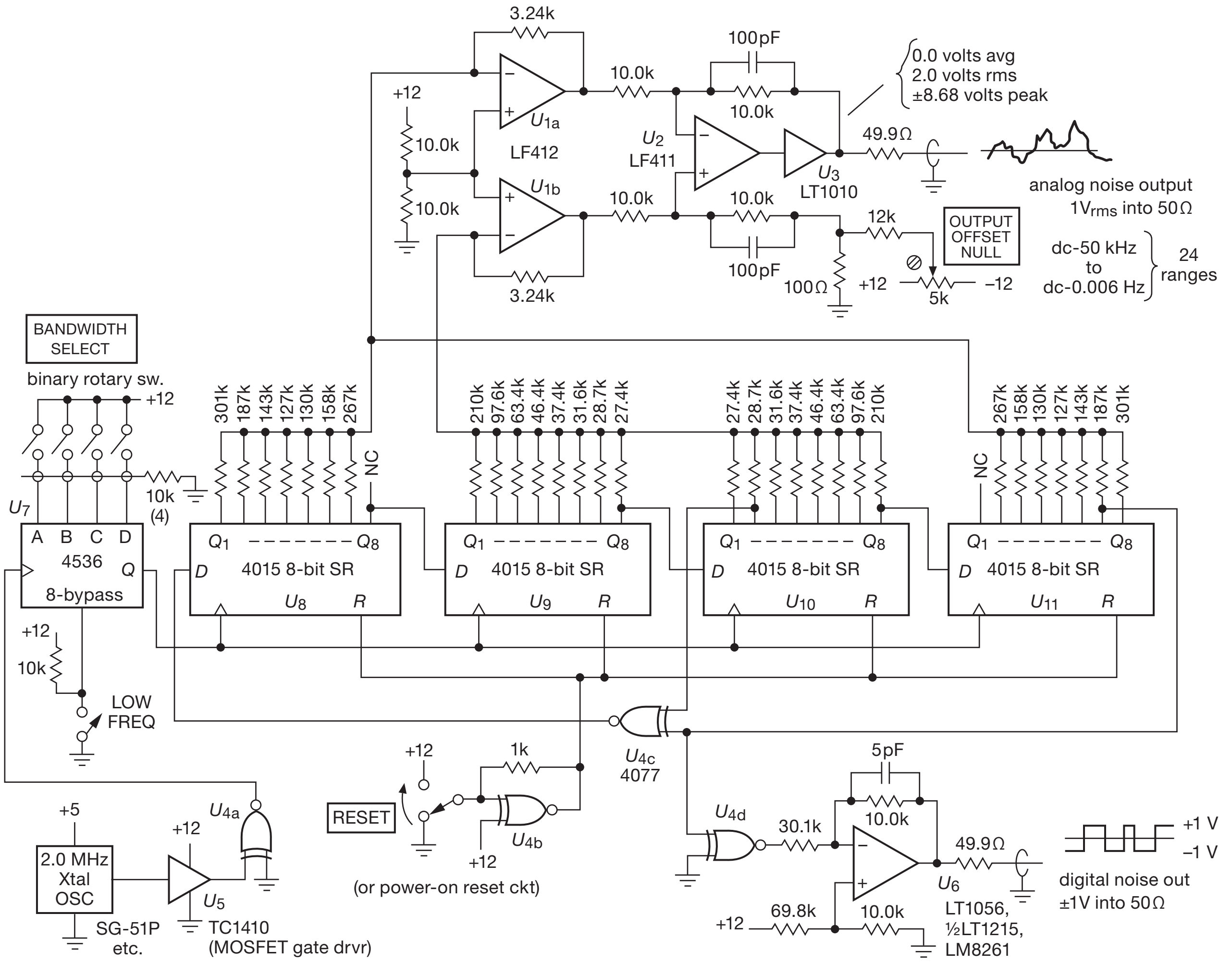


Figure 13.119. Wide-frequency-range laboratory pseudo-random noise source, inspired by Hewlett Packard's Model 3722A Noise Generator ("A precision digital instrument," in the parlance of its time, 1967). If you want to stick with discrete logic but need greater bandwidth, you could substitute 74LV164A 8-bit shift registers (with guaranteed clocking up to 125 MHz with a 5 V supply), with corresponding circuit changes throughout. Or you could use a fast cPLD or FPGA (§11.3.3) – but having gone that far, you might as well implement the digital lowpass filter in a DSP (digital signal processor) chip. Following this thread one step further, you could just code the whole thing in a fast microcontroller (§11.3.5 and Chapter 15), and use its on-chip D-to-A converter to generate the band-limited analog noise output.

logic is ideal for this application, because the outputs saturate cleanly at V_{DD} and ground).

The circuit in Figure 13.119 generates pseudorandom analog noise, with bandwidth selectable over an enormous range, using this technique. A 2.0 MHz crystal oscillator drives a 14536 24-stage programmable divider, generating selectable clock frequencies going from 1.0 MHz down to 0.12 Hz by factors of 2. A 32-bit shift register is connected with feedback from stages 31 and 18, generating a maximal-length sequence with 2 billion states (at the maximum clock frequency the register completes one cycle in a half hour). In this case we have used a $(\sin x)/x$ weighted sum over 32 successive values of the sequence. Op-amps U_{1a} and U_{1b} amplify the inverted and noninverted terms, respectively, driving difference amplifier U_2. The gains are chosen to generate a 1.0 V rms output with no dc offset into a 50 Ω load impedance (2.0 V rms open circuit). Note that this noise amplitude is independent of the clock rate, i.e., the total bandwidth. This digital filter has a cutoff at about $0.05 f_{\text{clock}}$, giving a white-noise output spectrum extending from dc to 50 kHz (at maximum clock frequency) down to dc to 0.006 Hz (at minimum clock frequency), in 24 steps of bandwidth. The circuit also provides an unfiltered output waveform, going between +1.0 V and −1.0 V.

There are a few interesting points about this circuit. Note that an exclusive-NOR gate is used for feedback so that the register can be simply initialized by bringing it to the state of all zeros. This trick of inverting the serial input signal makes the excluded state the state of all 1s (rather than all 0s as with the usual exclusive-OR feedback), but it leaves all other properties unaffected.

A weighted sum of a finite number of bits cannot ever produce truly Gaussian noise, since the peak amplitude is limited. In this case it can be calculated that the peak output amplitude (into 50 Ω) is ±4.34 V, giving a "crest factor" of 4.34. That calculation is important, by the way, because you must keep the gains of U_1 and U_2 low enough to prevent clipping. Look carefully at the methods used to generate an output of zero dc offset from the CMOS levels of +6.0 V average value (LOW=0 V, HIGH=12.0 V).

13.14.6 Wrapup

A few comments about shift-register sequences as analog noise sources.[138] You might be tempted to conclude from the three properties of maximal-length shift registers listed earlier that the output is "too random," in the sense of having exactly the right number of runs of a given length, etc. A genuine random coin-flipping machine would not generate exactly one more head than tail, nor would the autocorrelation be absolutely flat for a finite sequence. To put it another way, if you used the 1s and 0s that emerge from the shift register to control a "random walk," moving forward one step for a 1 and back one step for a 0, you would wind up exactly one step away from your beginning point after the register had gone through its entire cycle, a result that is anything but "random".

However, the shift-register properties mentioned earlier are true only of the entire sequence of $2^n - 1$ bits, *taken as a whole*. If you use only a section of the entire bit sequence, the randomness properties closely approximate a random coin-flipper. To make an analogy, it is as if you were drawing red balls and blue balls at random from an urn initially containing K balls in all, half red and half blue. If you do this *without replacing them*, you would expect to find approximately random statistics at first. As the urn becomes depleted, the statistics are modified by the requirement that the total numbers of red and blue balls must come out the same.

You can get an idea how this goes by thinking again about the random walk. If we assume that the only nonrandom property of the shift sequence is the exact equality of 1s and 0s (ignoring the single excess 1), it can be shown that the random walk as described should reach an average distance from the starting point of

$$X = [r(K-r)/(K-1)]^{1/2}$$

after r draws from a total population of $K/2$ ones and $K/2$ zeros. Because in a completely random walk X equals the square root of r, the factor $(K-r)/(K-1)$ expresses the effect of finite urn contents. As long as $r \ll K$, the randomness of the walk is only slightly reduced from the completely random case (infinite urn contents), and the pseudorandom sequence generator is indistinguishable from the real thing. We tested this with a few thousand PRBS-mediated random walks, each a few thousand steps in length, and found that the randomness was essentially perfect, as measured by this simple criterion.

Of course, the fact that PRBS generators pass this simple test does not guarantee that they would satisfy some of the more sophisticated tests of randomness, e.g., as measured by higher-order correlations. Such correlations also affect the properties of analog noise generated from such a sequence by filtering. Although the noise amplitude distribution is Gaussian, there may be higher-order amplitude correlations uncharacteristic of true random noise. It is said that the use of many (preferably about $m/2$) feedback taps (using an exclusive-OR parity-tree operation to generate the serial input) generates "better" noise in this respect.

Noise-generator builders should be aware of the 4557 CMOS variable-length shift register (six input pins set its length, any number from 1 to 64 stages); you have to use it in combination with a parallel-output register (such as the '4015 or '164) in order to get at the n tap; another useful chip is the HC(T)7731, a quad 64-bit shift register (256 bits in all) that runs to 30 MHz (min). Noise-generator builders should be even more aware of the ease with which programmable logic devices (cPLDs or FPGAs) can be coaxed into PRBS duty, as we illustrate in Chapter 11. Microcontrollers will do the job, too, but a PLD solution will be faster.[139]

When going for absolute maximum speed, though, you may want to fall back on good ol' discrete logic: As an admirable example, the CG635 "Synthesized Clock Generator" from Stanford Research Systems[140] can deliver a

[138] We are indebted to our late colleague Ed Purcell for these insights.

[139] And a full-custom chip fastest of all... if you can afford the $50k buy-in.

[140] Who obligingly provide full schematic diagrams and parts lists for the instrument.

7-stage PRBS (i.e., 2^7-1 states) at rates to 1.55 GHz.[141] This they do with some elegant tricks: (a) they use MC100EP emitter-coupled differential logic in an array of individually clocked flip-flops; this stuff is *fast* – the D-flops (MC100EP52D) have a setup time of 0.05 ns, zero hold time, and 0.33 ns (typ) propagation time; (b) they use differential clock and data lines (quiet, and fast); (c) the speed bottleneck is the XOR propagation time (0.3 ns), so they do a neat trick, namely (d) they arrange the FFs in a circle, with the data going around clockwise, and the clock going counterclockwise. This has the effect of delaying the clock to the first stage, relative to the clocking of the last two stages, by about 0.25 ns, thus allowing the XOR's (delayed) output to meet the first FF's setup requirement. And the cute geometry causes the clocks of the successive FFs to advance by about 0.05 ns each, thus effectively distributing the pain evenly around the array. Figure 13.120 shows how those clever folks laid out the printed circuit board to make it all happen.

13.14.7 "True" random noise generators

We remarked at the outset that algorithmically generated "noise" cannot be truly random – after all, you can *predict* exactly what's coming if you know the algorithm. Volumes have been written about this.[142] If you want *genuine* noise, you've got to look elsewhere.

A good place to start is some *physical* process (like radioactive decay) that is random *in principle*, and of course unpredictable. Circuit designers do not ordinarily include radioisotopes in their toolkits; but there are other physical processes that do just fine. For example, the noise voltage generated during avalanche conduction in a bipolar junction. We used this in the circuit responsible for the random byte collection that is shipped with the CD-ROM from *Numerical Recipes*, and available from Amazon. Figure 13.121 shows the circuit. Here is the description immortalized on the CD-ROM, where also are found the ~250 MB of what author William Press modestly refers to as "still the best 'random' bits anywhere:"

> Professor Paul Horowitz, of Harvard University, kindly constructed for us an electronic source of physical randomness. The analog noise source is a transistor junction that is biased to function as an avalanche diode. Physically, the noise current from this junction is generated by the random creation of electron-hole pairs in the junction, due to thermal noise (and, ultimately, quantum mechanical processes). Experimentally, the output of the device, viewed with a spectrum analyzer, is flat from close to dc out to 50 MHz.
>
> The "Hororan" circuit [Figure 13.121] samples the amplified analog voltage from the noise junction at a rate that is 8 times the desired baud rate for output (the latter typically 38.4 kbps or 115.2 kbps). The sample duration ("aperture") is very short, about 2 nanoseconds for the LT1016 latched comparator, so there is plenty of aliasing of the higher available frequencies into the sampled value. If the sample voltage is positive, a digital "1" bit is generated; if it is negative, a digital "0" bit is generated. The collected bits are continuously exclusive-OR'd (XOR) into a register. After every eight bits are collected, the state of the register is output as one bit in the raw file. After every eight output bits, the next two output bits are discarded during the formatting of the required RS-232 stop and start bits. The digital XOR and start/stop formatting functions are performed by a 26V12 PAL, whose programming is included here.
>
> It is intentional that we do not do any more complicated digital processing (mixing, scrambling, encrypting, etc.) within the Hororan box, because we want to be able to measure, and characterize, the degree of randomness coming out of the box. The box indeed has a measurable nonrandomness in its 1-point statistic. That is, the number of output 1s and 0s are not exactly equal but differ by, typically, a few parts in 10^4. (Indeed, the box has a trim adjustment to minimize this nonrandomness.) Notice that it requires several times 10^8 collected bits even to measure this bias convincingly – but we have satisfied ourselves that it is present. The bias drifts slowly with time, presumably in response to thermal and other environmental changes.
>
> We are unable to find, in numerical experiment, any trace of nonrandomness in the higher-point statistics of the raw collected bits. In particular, we have not been able to find (in several times 10^9 collected bits) any 2-point nonrandomness, either in the autocorrelation at small lags, or in the power spectrum at likely frequencies such as 60 Hz or its first few harmonics. On the basis of the physical construction of the Hororan box, we believe that M-point statistics with $M > 2$ (with the exception of high-point statistics that are

[141] Other sequence lengths commonly used for bit-error testing are $2^{23}-1$ and $2^{31}-1$.

[142] For example, Volume 2 of Donald Knuth's comprehensive classic, *The Art of Computer Programming*.

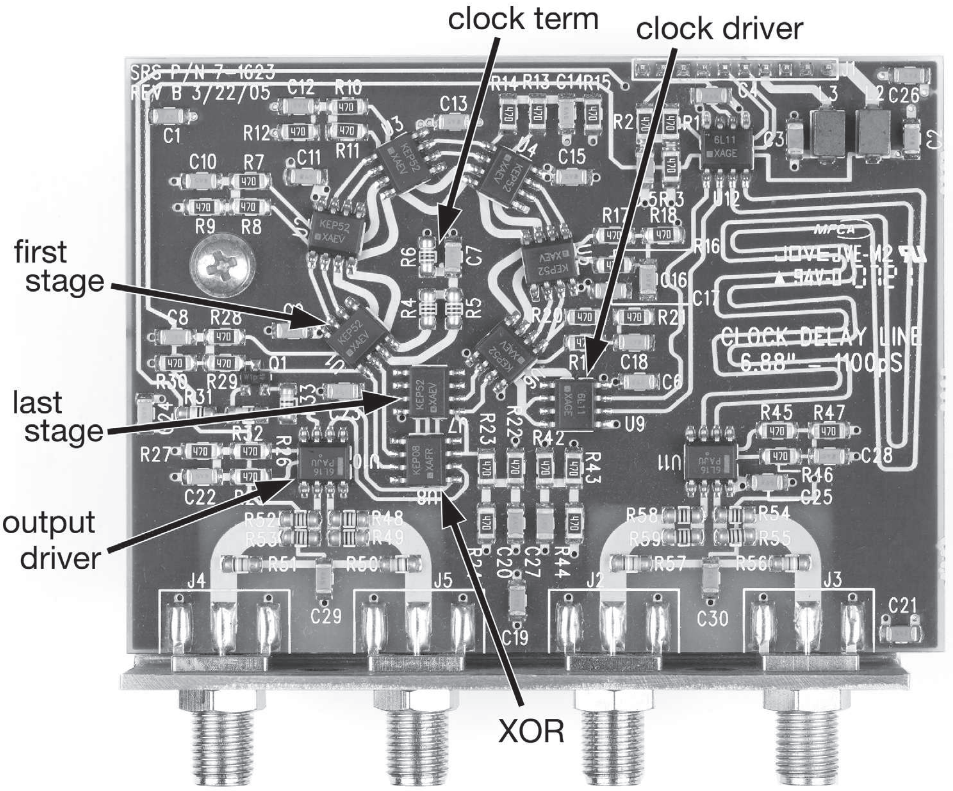


Figure 13.120. Racetrack layout of fast (to 1.55 Gbps) 7-bit PRBS, implemented with 100EP-series discrete LVPECL flip-flops and gates. Data goes clockwise, clock goes counterclockwise, and it all works!

the result of slow drifts in the 1-point bias) should be strongly decreasing with M. The reason is essentially that the transistor junction has no “memory” and has an internal timescale on the order of the reciprocal of its 50 MHz bandwidth.

Thus, while the output of the Hororan box is demonstrably non-random, we believe that its true “entropy per bit” is convincingly bounded as being close to 1 (the fully random value). If $1-e$ denotes the entropy per bit in a large file of N bits (a more detailed definition is given below), then we think that, with a high degree of experimental certainty (there is no way to “prove” this mathematically) we have $e < 0.01$.

Incidentally, the raw output of the Hororan box easily passes all of the tests in Marsaglia’s “Diehard battery of tests.”

To return to that CD-ROM: the output of this hardware was merely the starting point (the “collection phase”) for a complex sequence of operations on the way to the final published bytes: the “perfection phase” consists of multiple passes and shuffling through triple-DES encryption, XOR-ing with freshly collected bytes between each pass and again at the end. A full description can be found on the CD-ROM itself, in a folder labeled “Museum.”

13.14.8 A “hybrid digital filter”

The example of Figure 13.119 revisits the interesting topic of *digital filtering*, discussed earlier in §6.3.7. What we have here is simply a 32-sample finite-impulse-response (FIR) lowpass filter, in this case implemented in hybrid fashion – the samples are *digital* (logic HIGH or LOW, in the 32 flip-flop outputs of the shift register U_8–U_{11}), but the weighted sum is performed as an *analog* summing of currents into a pair of op-amp (U_{1ab}) summing junctions. In the more general case, a digital filter would operate on data representing a sampled analog waveform, with each sample represented as a multi-bit quantity (e.g., 16-bits for each channel, in the case of stereo CD audio), sampled at a rate sufficient to retain the full input frequency range of interest (for CD audio f_{samp}=44.1 kHz, about 10% above

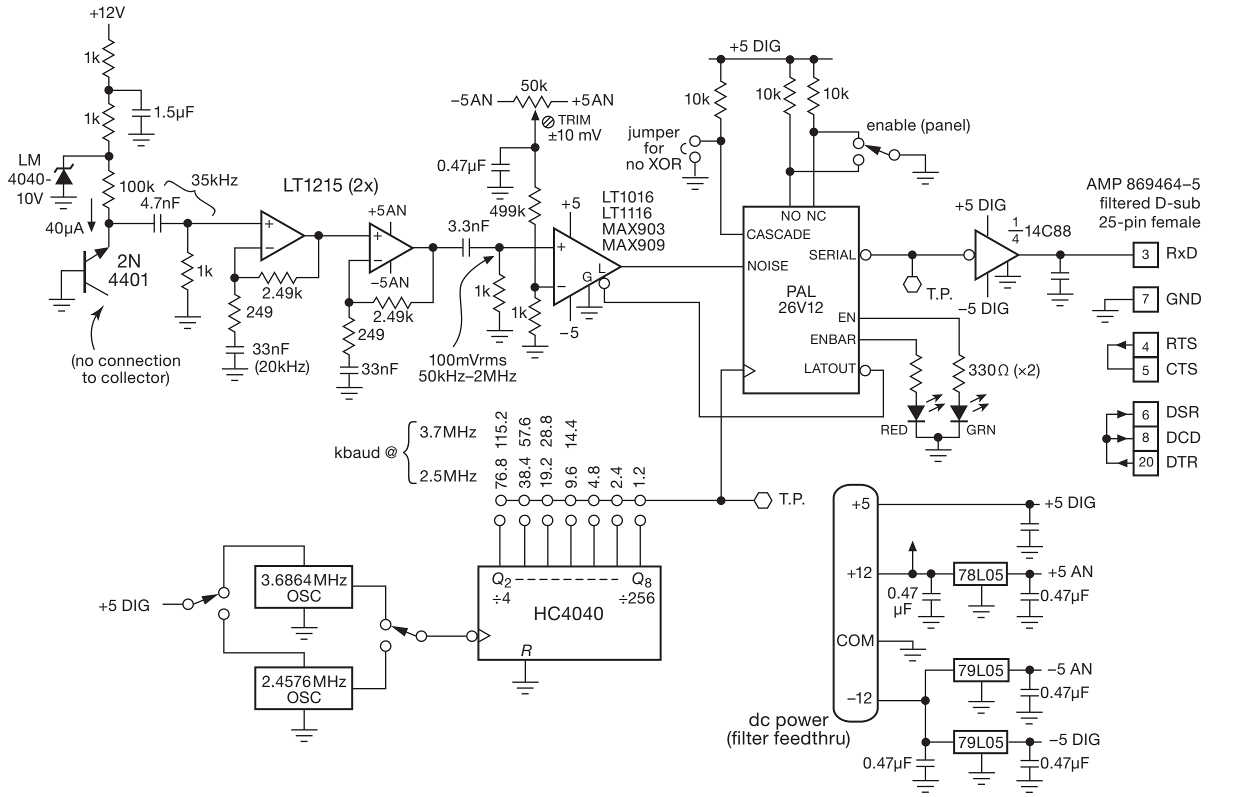


Figure 13.121. Using a physical process (base–emitter avalanche breakdown) to generate random bytes.

the minimum critical rate of 2×$f_{\max}$). And the weighting and summing would be done purely digitally (i.e., numerically), in multipliers and adders. Note however that high-quality digital processing does not *require* that the initial analog-to-digital sampling produce large word sizes: the discussion of delta–sigma conversion earlier in this chapter (§13.9) is a fine illustration of a "1-bit" digitized data stream, upon which digital filters can do their magic.

Additional Exercises for Chapter 13

Exercise 13.10. Why can't two n-bit DACs be used to make a $2n$-bit DAC by just summing their outputs proportionally ($\mathrm{OUT}_1 + \mathrm{OUT}_2/2^n$)?

Exercise 13.11. Verify that the peak output of the pseudorandom noise generator (Figure 9.94) is ±8.68 V.

Review of Chapter 13

An A-to-I summary of what we have learned in Chapter 13. This summary reviews basic principles, facts, and application advice in Chapter 13.

¶A. DACs and ADCs – Performance Parameters.

The subject of conversion between analog and digital signals dominated the chapter, with extensive description of the internals of ADCs and DACs, their performance, and example applications. Essential ADC performance parameters include resolution (number of bits), accuracy (linearity, monotonicity, stability, ENOB,[143] and spur-free dynamic range), speed (conversion time and latency), input range, output format (serial I^2C or SPI; or parallel), reference (internal or external), packaging, and additional features such as internal programmable-gain amplifier. The essential DAC performance parameters are similar, but include output type (voltage or current), glitch energy, and variants such as multiplying DAC (MDAC) or programmable internal digital scaling.

¶B. DAC Types.

The choice of converter for a particular application should be based mostly on its performance parameters (¶A), rather than on the method of conversion. But it's important to know about a converter's inner workings so that you won't be caught unaware by its idiosyncrasies.

Resistor-string DAC. This simplest technique (§13.2.1) consists of a series string of 2^n equal-value resistors, biased by a stable voltage reference, with digitally controlled MOS switches that pick off the dc voltage at a chosen tap as selected by the n-bit binary input code. These inexpensive DACs range from 8 to 16 bits of resolution; the output is strictly monotonic (a desirable feature for a digital control loop).

R–2R ladder DAC. This popular technique (§13.2.2) exploits the binary-weighting property of an R–$2R$ ladder (Figure 13.5), requiring only $2n$ (rather than 2^n) resistors and switches to create an n-bit DAC. These converters are inexpensive, but not inherently monotonic (although there are many that guarantee monotonicity). Their integral nonlinearity (INL) is generally superior to resistor-string DACs, making them better for precise voltage-setting applications.

Current-steering DAC. In contrast to these voltage-output converters (resistor string and ladder types), the current-steering DAC (§13.2.3) excels in speed, and also permits easy combining of outputs. It uses an array of n switches to steer a set of binary-weighted currents (Figure 13.6). These DACs are wickedly fast, but their current output is limited in compliance; if you tack on a transresistance amplifier to create a voltage output, it will limit the dynamic performance (speed and settling time). Best to drive a load that wants a current input, or simply drive a 50 Ω load resistor (75 Ω for video), appropriate for RF and high-speed applications. *A caution*: the stability and initial accuracy of current-output DACs can be quite poor if you are foolish enough to ignore the matched internal feedback resistor; see the warning in §13.2.5 and the correct use of feedback shown in Figure 13.9.

Delta–sigma DAC. This technique (§§13.9 and 13.9.8), sometimes called a "1-bit DAC," is a two-step process in which a bitstream is first produced in an integrating digital modulator, then lowpass filtered to produce the analog output voltage; see the block diagram in Figure 13.50. It is popular in professional audio, where it can deliver stunningly linear high resolution output signals, for example dual (stereo) 20-bit resolution at 192 ksps. The delta–sigma technique is popular also for ADCs; in the pro-audio arena, for example, a popular part is the CS5381 dual 24-bit 192 ksps converter (§13.9.11D).

Multiplying DAC. An MDAC (§13.2.4) accepts a varying V_{ref} input, specifying both a voltage range (usually bipolarity, e.g., −15 V to +15 V) and a reference-multiplying bandwidth (typically a megahertz or more). An MDAC permits digital amplitude control of signals within its bandwidth, and it lets you do ratiometric measurements with an imprecise reference voltage. A caution: capacitive feedthrough severely limits the bandwidth when operated with small-value digital codes; see Figure 13.7.

Pulse-width modulation and F-to-V conversion. When driving a load that is inherently slow (for example, a heater), a simple and effective form of D/A conversion is PWM (§13.2.8). Related techniques are F/V converters and rate multipliers (§§13.2.9 and 13.2.10).

¶C. DAC Application Examples.

In conversion applications the devil is very much in the details. In this chapter we explored four real-world applications, revealing their satanic complications.

(1) A four-channel general-purpose laboratory source (§13.3.1) illustrated precision and stable low-noise design,

[143] Where the "effective number of bits" ENOB=$1.44\log_e(V_{\text{span}}/V_{\text{n(rms)}})$. As explained in nice detail in Analog Devices' tutorial MT-003 ("Understand SINAD, ENOB, SNR, THD, THD+N, and SFDR so you don't get lost in the noise floor"), ENOB is related to SINAD (signal-to-noise+distortion) by the relation ENOB=(SINAD−1.76 dB)/6.02, where SINAD (in dB) is given, in terms of rms quantities for signal, noise, and distortion, by SINAD=$20\log_{10} S_{\text{rms}}/(N{+}D)$.

with an external current-to-voltage configuration providing flexible output ranges.
(2) A simpler eight-channel voltage source (§13.3.2) is easily designed with a fully integrated voltage-output DAC (LTC2656); it lacks the flexible output range capability of the previous example, along with some degradation in terms of noise and stability.
(3) A nanoamp-scale bipolarity current source of wide compliance (§13.3.3) is a circuit of considerable subtlety, offering low noise and low drift while sourcing or sinking nanoamps over a range of ±10 V. There's additional teaching here, on the subject of floating current sources in general.
(4) A precision bipolarity coil driver (§13.3.4) illustrates a DAC application that pushes the limits of resolution (20 bit, i.e., parts-per-million), with corresponding control of noise and stability. This example includes plenty of discussion of requisite amplifier and reference choices, and of loop compensation and stability.

¶D. DAC Choices.

For highest linearity, delta–sigma DACs are best, with accuracy and linearity to 20 bits at audio speeds, and sometimes with excellent dc specs as well (e.g., TI's millisecond-speed 20-bit DAC1220); however, watch out for broadband and clock noise (the DAC1220 has $\sim$1000 nV/$\sqrt{\text{Hz}}$ at 1 kHz, compared with $\sim$10 nV/$\sqrt{\text{Hz}}$ for resistor ladder converters).

For high-accuracy applications at medium speed, there are many excellent R–$2R$ and linear ladder DACs, for example TI's DAC8552 (dual 16-bit, serial in, voltage out, ext ref, very low glitch, 10 μs settle; DAC8560/4/5 similar, with internal ref); or ADI's AD5544 or TI's DAC8814 (quad 16-bit MDAC, serial in, current out, 0.5–2 μs settle with external I-to-V op-amp); or LTC's LTC1668 (16-bit, parallel in, differential current out, 20 ns settle into 50 Ω as "voltage out"); or TI's DAC9881 (18-bit serial in, rail-to-rail voltage out, ext ref, low-noise, 5 μs settle).

For the highest speed you can't beat current-steering converters, for example the TI DAC5681/2 (16-bit, 1 Gsps) or the ADI AD9739 (14-bit, 2.5 Gsps).

¶E. ADC Types.

As with DACs, the choice of converter for a particular application should be based mostly on its performance parameters (¶A), rather than on the method of conversion. As with DACs, though, it's important to know about a converter's inner workings so that you won't be caught unaware by its idiosyncrasies.

Flash, or "parallel." In this simplest technique (§13.6) the analog input voltage is compared with a set of fixed reference voltages, most simply by driving an array of 2^n analog comparators, to generate an n-bit result, with resolutions to n=8 bits and speeds to 20 Gsps (not in the same device). Variations on this theme include pipelined or folded architectures, in which the conversion is done in several steps, each of which converts the "residue" of the previous low-resolution conversion. These converters top out at 16-bit resolution and speeds to several Gsps (not in the same device); the pipelined architecture creates latency, which can be as much as 20 clock cycles.

Successive approximation. In this technique (§13.7) successive trial codes (generated by internal logic) are converted to voltages by an internal DAC and compared with the analog input voltage. It requires just n such steps to do an n-bit conversion. The internal DAC can be implemented as a conventional n-stage R–$2R$ resistor ladder, or, interestingly, as a set of 2^n binary-scaled capacitors; the latter method is known as a *charge-redistribution* DAC. SAR-type converters are intermediate in speed and accuracy (flash converters are faster; delta–sigma converters are more accurate). The input must be stable during the conversion (thus requiring some form of input track-and-hold); SAR converters can have missing codes.

Voltage-to-frequency. This technique (§13.8.1) produces an output pulse train (or other waveform) whose frequency is accurately proportional to the analog input voltage. In an *asynchronous* V/F the oscillator is internal and free-running; a *synchronous* V/F requires an external source of clock pulses, gating a fraction of them through such that the *average* output frequency is proportional to the analog input. V-to-F converters can be used for simple control of averaging loads (like a heater); but they barely deserve listing as an ADC, and are best thought of as a primitive approximation to the superior delta–sigma converter (see below).

Single-slope integration. In this technique (§13.8.2) an internally generated analog ramp (capacitor charged by a current source) goes from zero volts to the analog input voltage, timed by counting pulses from a fast fixed-frequency clock; the count is proportional to the value of the analog input. Single-slope integration is not particularly precise, yielding that turf to dual- or multi-slope integration (see below). But it is used in applications such as pulse-height analysis, where speed and uniformity of adjacent codes is required; it is also used for time-to-amplitude conversion (TAC).

Dual-slope and multi-slope integration. These techniques (§§13.8.3, 13.8.4, and 13.8.6) are variations on single-slope integration, effectively eliminating errors from comparator offsets and component stability. In *dual-slope integration* the capacitor is ramped up for a fixed time with a current proportional to the input signal, and ramped back down with a fixed current; the latter time interval is proportional to the analog input. In *quad-slope integration* the input is held at zero while a second such "autozero" cycle is done. The so-called *multi-slope* technique is somewhat different, with a single conversion consisting of a succession of fast dual-slope cycles (in which the input is integrated continuously, combined with subtractive fixed-current cycles), and with a correction based upon the partial-cycle residue at both ends. In some aspects it's a close cousin of the delta–sigma method (next). These integrating ADC methods are ideal for low-speed (milliseconds) high-resolution (20–28 bits) voltmeter-style measurements.

Delta–sigma ADCs. In this technique (§13.9) a *modulator* converts the oversampled analog input voltage to a serial *bitstream*; then a digital lowpass filter accepts that bitstream as input, producing the final *n*-bit digital output. In its simplest form the modulator consists of an integrator acting on the difference between the analog input voltage and the value of the 1-bit output serial bitstream, to determine the next output bit. Variations include higher-order modulators (a succession of weighted integrators), or bitstreams that are several bits wide, or both. Delta–sigma converters deliver excellent resolution (to 24 bits) and conversion rates to a few megasamples per second. They are hugely popular in professional audio.

¶F. ADC Application Examples.

In this chapter we illustrated several application examples.
(1) A fast flash ADC (§13.6.2), driven differentially, illustrated input filtering and low-jitter clock generation.
(2) A low-noise high-stability 18-bit successive-approximation converter with a 2 Msps conversion rate (§13.7.3) illustrated careful input conditioning to achieve low noise and low drift.
(3) Three applications of delta–sigma converters (Figures 13.66, 13.67, and 13.68) illustrated a wide range (±10 V) multiplexed converter with good accuracy (18 bits or better); an industry-standard 24-bit $\Delta\Sigma$ converter of good noise, drift, and linearity specifications, and with chopper-stabilized programmable-gain amplifier for accurate measurements of low-level signals; and a pro-audio conversion application, where dc accuracy and latency are largely irrelevant, instead featuring excellent channel matching and with audio-type output formats.

In this chapter we illustrated further conversion examples, including some unusual ADCs and DACs (§13.11), and a few conversion *system* examples (§13.12). The latter included a multiplexed 16-channel data-acquisition system (§13.12.1), a parallel multichannel successive-approximation data-acquisition system (§13.12.2), and an analogous parallel multichannel delta–sigma data-acquisition system (§13.12.3). The last two examples included both external and integrated multichannel solutions, in tutorial narratives.

¶G. ADC Choices.

For low-speed (to kilosamples/sec) and high-accuracy (to 24 bits or more) applications, integrating converters (dual-slope, multislope) are best; there are also some excellent delta–sigma converters (which can be thought of as integrating converters).

For moderate-speed applications (to several Msps), both delta–sigma and successive-approximation converters are competitive, with resolution to 20 bits or more; delta–sigma converters have significantly longer latency.

For high-speed (100s of Msps) choose pipelined flash converters, mindful of their high latency. And for the highest conversion rates (>250 Msps) some variant on folding flash gets you to 8–12 bits and 3 Gsps. Simple flash is the fastest, but you pay the price in resolution (e.g., the Analog Devices HMCAD5831: 3-bit conversion at 20 Gsps).

¶H. Phase-locked Loops.

The chapter continued with the important mixed-signal subject of PLLs (§13.13), circuits in which feedback forces a signal derived from a voltage-controlled oscillator to match an input signal's frequency. PLL applications include frequency multiplication and frequency synthesis, clock generation and recovery, and demodulation of AM, FM, and digitally modulated signals. In addition to a voltage-controlled oscillator (VCO), the essential components of a PLL (Figure 13.85) are (a) the phase detector and (b) the loop filter.

The *phase detector* (PD) compares the phases of the input signal and the VCO-derived signal, generating an output signal representative of their relative phase. The simplest phase detector (type I) multiplies the input signals. It is applicable to either analog or digital input signals; for the latter it is just an exclusive-OR gate (Figure 13.86). The other common phase detector (type II) generates output pulses according to the relative timing of transitions at its two inputs (Figure 13.87). It works only with digital signals. Type II phase detectors have the benefit of locking with zero phase difference, and of introducing no ripple at the phase-detection frequency; however, they are more sensitive to input signal jitter than type I phase detectors.

The *loop filter* smooths the PD's output, with a time constant that sets the loop's response; the latter should be long compared with the comparison frequency, but fast enough to follow changes in the input frequency as required by the PLL's application. To clarify this last point, you'd want a long time constant if the PLL has a low-noise VCO and is used to generate a clean stable clock from a noisy input reference of fixed frequency; but if you want the PLL to follow a wobbling input frequency (e.g., clock recovery from a tape or disk drive) you would make the loop fast enough to respond to the changing input frequency. *A caution*: there's an inherent 90° lagging phase shift in the PLL: a measurement of phase is used to adjust frequency, but phase is the integral of frequency. So a simple lowpass loop filter (which introduces additional lagging phase shift, asymptotic to 90°) creates a marginally stable loop. The solution is to slow the rolloff in the region of unity loop-gain (a "zero," in the language of *s*-plane analysis; see the discussion in Chapter *1x*), as seen in Figure 13.98.

The procedure for designing the loop filter is illustrated in §13.13.3. And the use of PLLs for the important applications of frequency synthesis, analog and digital demodulation, pulse synchronization, and laser offset locking is discussed in some detail in §13.13.6.

¶I. Pseudorandom Digital Noise.

The chapter concluded with the fascinating topic of the generation of deterministic pseudorandom bit sequences (PRBSs) from linear feedback shift registers (LFSRs). These find application in communication channel testing (eye diagrams, see for example Figure 12.131) and in spectrum-spreading applications in digital communications (e.g., the transmitted navigational signals from GPS). And, they're just plain fun. The basic technique (described in some detail also in Chapter 11, in §11.3) is a clocked shift register whose input is derived from an XOR combination of two (or more) carefully chosen bits of the register (Figure 13.111).

As demonstrated in Chapter 11, PRBS generators are easily made with a tiny bit of PLD logic, or some microcontroller code, or even just a few logic chips (e.g., a 74HC7731, 74HC164, and 74HC86, clocked at 25 MHz, makes a PRBS whose cycle length is more than 10^{62} years; that's 10^{52} times the age of the universe). The output sequence can be lowpass filtered to generate an analog noise source of settable bandwidth; see the analog-filtered colored noise source in Chapter 8 (Figure 8.93) and the hybrid digitally-filtered white noise source of Figure 13.119.

Pseudorandom noise is not truly random, even though its statistical properties may mimic genuine randomness as measured by various tests (see §13.14.6), rendering it suitable for some applications. To generate honest random bit sequences you need to exploit the random properties of some physical process like β-decay in unstable isotopes. More convenient for the circuit designer is something like thermal noise in resistors, or avalanche noise in semiconductor junctions; the latter is illustrated in §13.14.7.