DC-accurate, 32-bit DAC achieves 32-bit resolution

W Stephen Woodward, Chapel Hill, NC

Some applications, such as ADC testing and calibration, require a DAC with extremely good resolution, monotonicity, accuracy, and resolution. In these categories of performance, the circuit in **Figure 1** is hard to beat. Its typical specifications follow:

- Resolution=32 bits=3×10^{-10}=1.2 nV=192 dB.
- DNL (differential nonlinearity)=27 bits=400 nV=162 dB.
- INL (integral nonlinearity)=22 bits=1.6 µV=130 dB.
- Full-scale accuracy (untrimmed)=11 bits=±2.5 mV=66 dB.
- Zero accuracy=23 bits=±500 nV±10 nV/°C=140 dB.
- Ripple and noise=21 bits=2 µV p-p=128 dB.

The basis of the DAC's 32-bit resolution is the summing of two 16-bit PWM signals by analog switches S_1 and S_2 and precision resistor network R_2 through R_6. The DAC's monotonicity and DNL are theoretically infinite, and, in practice, the only limit is the 1-to-2^{16} ratio of R_2: $(R_6+R_5+R_{S2\text{-}ON})$ and R_3: $(R_6+R_4+R_{S2\text{-}ON})$. Typical accuracy of 0.1% resistors yields a DNL of approximately 0.1 ppm=27 bits.

The less-than-0.1Ω output impedance of the AD586 reference and the 130-dB CMR (common-mode rejection) of chopper-stabilized "zero-drift" amplifier A_1 mostly limit INL. R_7 suppresses a potential contribution from asymmetry in $R_{S1\text{-}ON}$, yielding the typical INL of approximately 0.3 ppm=22 bits.

▶What are your design problems and solutions? Publish them here and receive $150! Send your Design Ideas to edndesignideas@reedbusiness.com.

▶To see all of *EDN*'s Design Ideas, visit www.edn.com/designideas.

Zero-accuracy and output-noise specs are at the low-microvolt level because of the excellent specifications

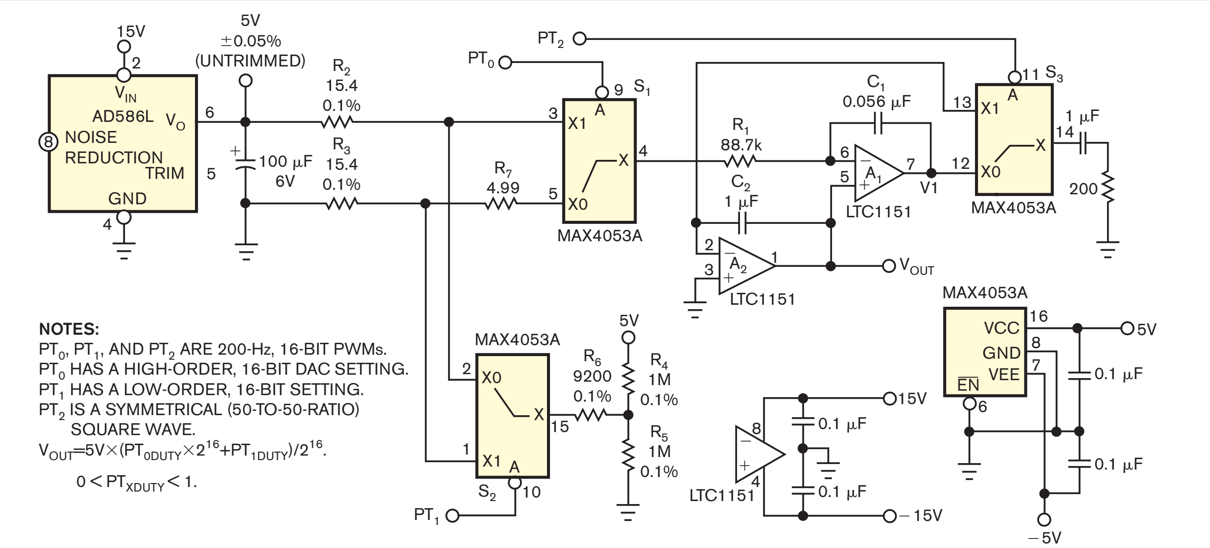

Figure 1 This DAC circuit sums two 16-bit PWM signals using precision analog switches to achieve 32-bit resolution.

of the LTC1151 A_1 and A_2 op amps and the charge-injection performance of the MAX4053A S_2: approximately 0.4 ppm, or 23 bits.

The precision of the AD586L 5V reference, which is ±500 ppm untrimmed, limits absolute accuracy. If your design requires greater accuracy, then you can use an Analog Devices (www.analog.com) simple trim circuit to further tweak it. There's nothing critical about the suggested 200-Hz PWM cycle. You need to change only R_1 and C_1 to accommodate any convenient frequency. How closely the R_1C_1 time constant matches the PWM-cycle time determines the settling time of the A_1-S_2-A_2 synchronous "zero-ripple" integrate-and-hold filter, and can be as fast as one cycle if the match is exact. **EDN**

Digitally programmable instrumentation amplifier offers autozeroing

Marián Štofka, Slovak University of Technology, Bratislava, Slovakia

The current trend in advanced instrumentation amps is to use no external resistors. In these amplifiers, a gain-control word, comprising a binary-coded one, sets the voltage gain. Several integer gains within one to 1000 are currently available; however, this range does not yet include a gain of three. Although external-resistor-free amplifiers with a gain of three are available, they are neither instrumentation amps nor autozeroed devices (**Reference 1**). These features are essential in applications requiring accurate processing of low-level voltages. You can use the circuit in **Figure 1** for applications requiring instrumentation amps having voltage gains of three or 10 and the ability to process voltages as low as 1 mV.

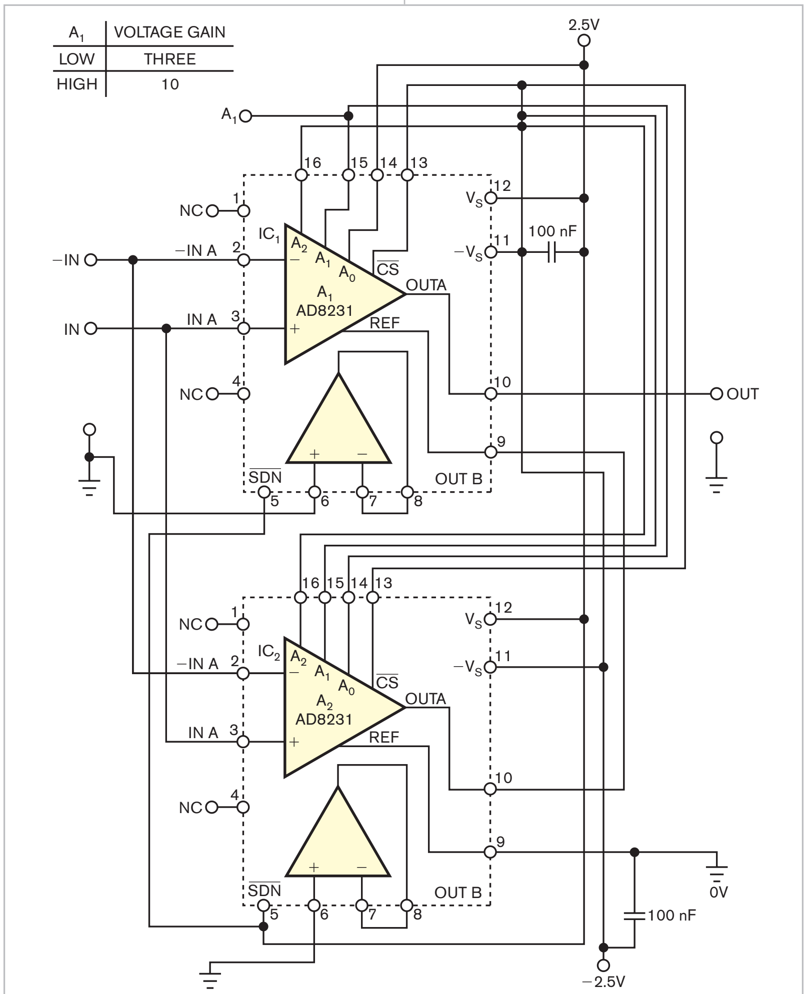

Figure 1 The autozeroed instrumentation amp, digitally programmable for voltage gains of three and 10, can help you to overcome the current inavailability of monolithic ICs for this task.

This design achieves a voltage gain of three by using the "algorithm" of 3=2+1. The circuit comprises two units of the Analog Devices (www.analog.com) digitally gain-programmable, autozeroed AD8231 instrumentation amp. These ICs have voltage gains that are programmable as powers of two—that is, one, two, four … 128 (**Reference 2**). Amplifier A_1 in IC_1 is preset to provide a gain of two, and auxiliary amp A_2 in IC_2 is preset to a gain of one. The noninverting and inverting inputs of A_1 and A_2 connect together. The output of A_2 connects with reference input REF_1, and reference input REF_2 serves as a freely usable reference. You can thus calculate the output voltage as $V_{OUT}=V_{OUT1}+V_{REF1}=V_{OUT1}+V_{OUT2}=2\Delta V_{IN}+\Delta V_{IN}=3\Delta V_{IN}$, where ΔV_{IN} is the input-difference voltage.

Similarly, you can achieve a voltage gain of 10 according to a symbolic formula of 10=8+2. This time, A_1 has a voltage gain of eight, and A_2 has a gain of two. Using **Reference 2**, you can derive that, for gains of both three and 10, $A_{1A1}=A_{0A2}$. Therefore, the gain-control pins connect and remain low for a gain of three, and the high at these pins sets the gain to 10. Note that three approaches the square root of 10, or approximately 3.16. You can therefore consider it as roughly the geometric center of a decade. EDN

REFERENCES

1. Štofka, Marian, "Gain-of-three amplifier requires no external resistors," *EDN*, Aug 16, 2006, pg 74, www.edn.com/article/CA6360318.
2. "Zero Drift, Digitally Programmable Instrumentation Amplifier, AD8231," Analog Devices Inc, 2007, www.analog.com/en/prod/0,2877,AD8231,00.html.

C# application controls simple ADC

Yury Magda, Cherkassy, Ukraine

This Design Idea describes a simple and low-cost ADC that you control using the serial port of a PC running Windows XP/Vista. The hardware comprises Microchip's (www.microchip.com) 12-bit SAR (successive-approximation-register) MCP3201 ADC, which attaches to the serial port of the PC through the RTS, CTS, and DTR lines (**Figure 1**).

The circuit uses an SPI (serial-peripheral-interface)-compatible interface to communicate with the MCP3201. The MAX232 chip transforms the RS-232 levels into TTL-compatible levels that the MCP3201 converter requires to operate. The analog signal comes through the IN+ pin of the MCP3201. The output digital stream of bytes on the D_{OUT} pin goes through the CTS line to the serial port of the PC. The RTS line of the serial port provides clock pulses that go through the CLK pin of the converter. Each separate bit appears on D_{OUT} on the falling edge of CLK, and the application should latch the bit on the rising edge of the clock pulse.

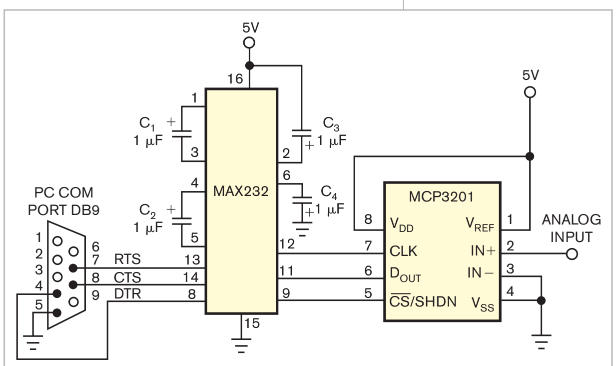

Figure 1 This simple, low-cost ADC comprises a 12-bit SAR ADC, which attaches to the serial port of the PC through the RTS, CTS, and DTR lines.

The DTR line produces the $\overline{CS}$ signal that frames the conversion process. The $\overline{CS}$ signal must be low while the conversion is in progress (**Figure 2**).

The meaningful bits, with MSB first, appear on D_{OUT} after the third CLK pulse goes low. It implies that, if you miss the first three data bits, the software would programmatically realize it. The software that controls the device is written in free Microsoft (www.microsoft.com) Visual C# 2008 Express Edition. It uses a built-in SerialPort component that allows you to get full control over the serial port of the PC. You implement the software as a simple console application containing **Listing 1**, which is available with the Web version of this Design Idea at www.edn.com/081030di1. The program is uncomplicated, so you can easily modify it. For instance, you could send the data from ADC over the Internet or pass it into Microsoft Excel or Microsoft Access for further processing.

You can improve the simplified circuit in **Figure 1** for higher accuracy by placing a lowpass filter in the analog-signal chain. You should also always use a bypass capacitor with the MCP3201. Place a capacitor with a recommended value of 1 µF as close as possible to the device's pin. You can also replace the MCP3201with a similar SAR ADC that works with an SPI-compatible interface. For instance, you may use an LTC1286 or an LTC1297 device from Linear Technology (www.linear.com). If you plan to use a different ADC, you must

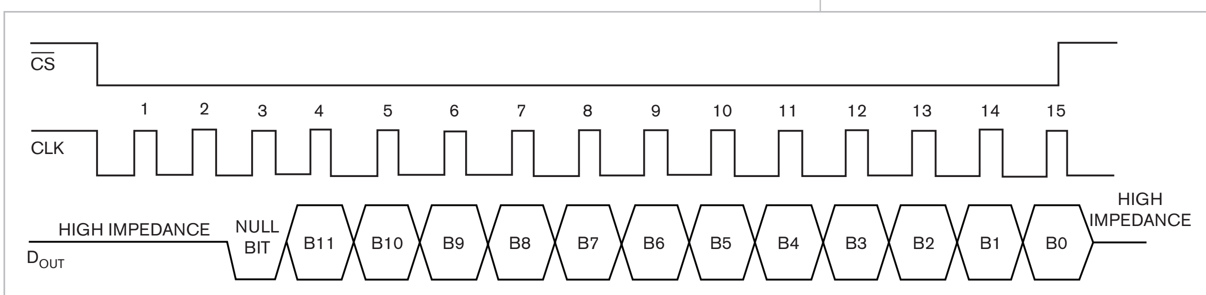

Figure 2 The DTR line produces the $\overline{CS}$ signal that frames the conversion process. The $\overline{CS}$ signal must be low while the conversion is in progress.

make some changes in the hardware and software. The changes necessary to the hardware are obvious, and you may need to change the software source code of the application to correct the *for (...)* loop statement according to the timing diagram of the selected part.EDN

Perform bitwise operation in Excel spreadsheets

Bruno Muswieck, Eletroeste, Uruguaiana, Brazil

Microsoft's (www.microsoft.com) Excel helps engineers with calculus and graphics to solve problems. But engineers often have to perform bitwise operations, too. **Figure 1** shows the bitwise operations' tables. The bitwise functions work for decimal values. If you need to use hexadecimal or binary values, you must use the Dec2Bin and Dec2Hex functions to convert all the decimal values for the desired format.

To install the add-in bitwise functions, you can download the ins.xla file from the Web version of this Design Idea at www.edn.com/081030di2. In Excel, go to Tools, then Add-Ins, and then Browser. Find the downloaded add-in xla file and click OK. Now, Excel can run the bitwise functions.

You can also download some examples from the *EDN* Web site at www.edn.com/081030di2 (**Reference 1**).EDN

REFERENCE

1. Kagan, Aubrey, *Excel by Example: A Microsoft Excel Cookbook for Electronics Engineers*, Newnes Elsevier, 2004, ISBN 0-7506-7756-2.

AND		RESULT
0	0	0
1	0	0
0	1	0
1	1	1

OR		RESULT
0	0	0
1	0	1
0	1	1
1	1	1

XOR		RESULT
0	0	0
1	0	1
0	1	1
1	1	0

NOT	RESULT
0	255
1	254

SHIFT RIGHT		RESULT
BINARY	SHIFTED	BINARY
11010010	6	00000011

SHIFT LEFT		RESULT
BINARY	SHIFTED	BINARY
01100100	2	10010000

Figure 1 With the help of some new add-in functions, you can perform these bitwise operations in Excel.