EDITED BY MARTIN ROWE
AND FRAN GRANVILLE

designideas

READERS SOLVE DESIGN PROBLEMS

Precision analog bests digital in speed, noise, simplicity, and ease of implementation

Paul Antonucci, Alberti's Window, Watertown, MA

Every once in a while, I read that analog is on the way out, and everything should be digital. Recently, I was involved in a design project that illustrates that this belief doesn't apply in many situations.

The problem was to put two new optical breaks into the drive mechanism of a group of robotically controlled, Internet-accessible telescopes in an education application. The drive mechanisms have been showing signs of wear from excess slipping at end of travel: The sensors would signal end of travel, so there would be no slipping. The fork arms and other locations enclosed the internal wiring harnesses of the telescopes so that rewiring the telescopes would have been awkward.

So, the best idea was to encode the signals from the new sensors into the current wiring. There was one digital signal available; the challenge was to encode the two new sensors onto that signal.

Using a digital approach would have involved adding a small microcontroller to the base and encoding a serial digital signal to send up the tube, with appropriate synchronous pulses, data, and check sums, which then would undergo decoding at the CPU. This approach would have required some sort of reset provision because the telescope needed to operate independently for months, and those serial digital signals would have undesirable switching noise on them. The CPU would also have had to spend time grabbing, decoding, and synchronizing the signals, taking up more time. In addition, we would have had to have written some messy bit-banging code: not a huge challenge—but not a simple or elegant one, either.

Instead, this approach uses a variation on a simple adder circuit in which each sensor contributes a different amount. Taking the basic binary idea that one sensor adds ±1; the next, ±2; and the last, ±4, the approach uniquely represents each state.

The basic requirements are a voltage reference, some op amps, and a summing junction. This application uses IC_4, a Texas Instruments (www.ti.com) REF3040 voltage reference, which has an output tolerance of 0.2% yet costs only approximately $1 (**Figure 1**). This reference generates a voltage of 4.096V and produces enough current to run the op amps, which run rail to rail to within a few millivolts. Be careful, however: Some "rail-to-rail" op amps have insufficient current drive near the rails. This circuit uses 0.1%-precision resistors, which cost only about 20 cents. Remember that you can use two 10-kΩ resistors in series and two in parallel to create the 20- and 5-kΩ resistances that you see in the **figure**. The assembly and bill of materials are simpler and precision is better because the distributions around the ideal resistor value tend to cancel out. **Table 1** lists the predicted output voltages.

TABLE 1 PREDICTED OUTPUT VOLTAGES

Sensor 1	Sensor 2	Sensor 3	Output
0	0	0	0.256
0	0	1	0.768
0	1	0	1.28
0	1	1	1.792
1	0	0	2.304
1	0	1	2.816
1	1	0	3.328
1	1	1	3.84

Tests with a voltmeter show that all output voltages were within 1 mV of the predicted output values. The error budget of less than 1% shows that you could use this method to encode several more sensors. In the telescope, the CPU's ADC reads the outputs. Read it twice to ensure that you aren't catching it at a transi-

DIs Inside

▶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.

tion. The advantages of the circuit include the fact that its dc signals ensure that there's no noise and that the updates are nearly instantaneous. Also, because op amps are simple, virtually indestructible, and insensitive to noise, no reset circuits are necessary. Best of all, the design requires no programming. EDN

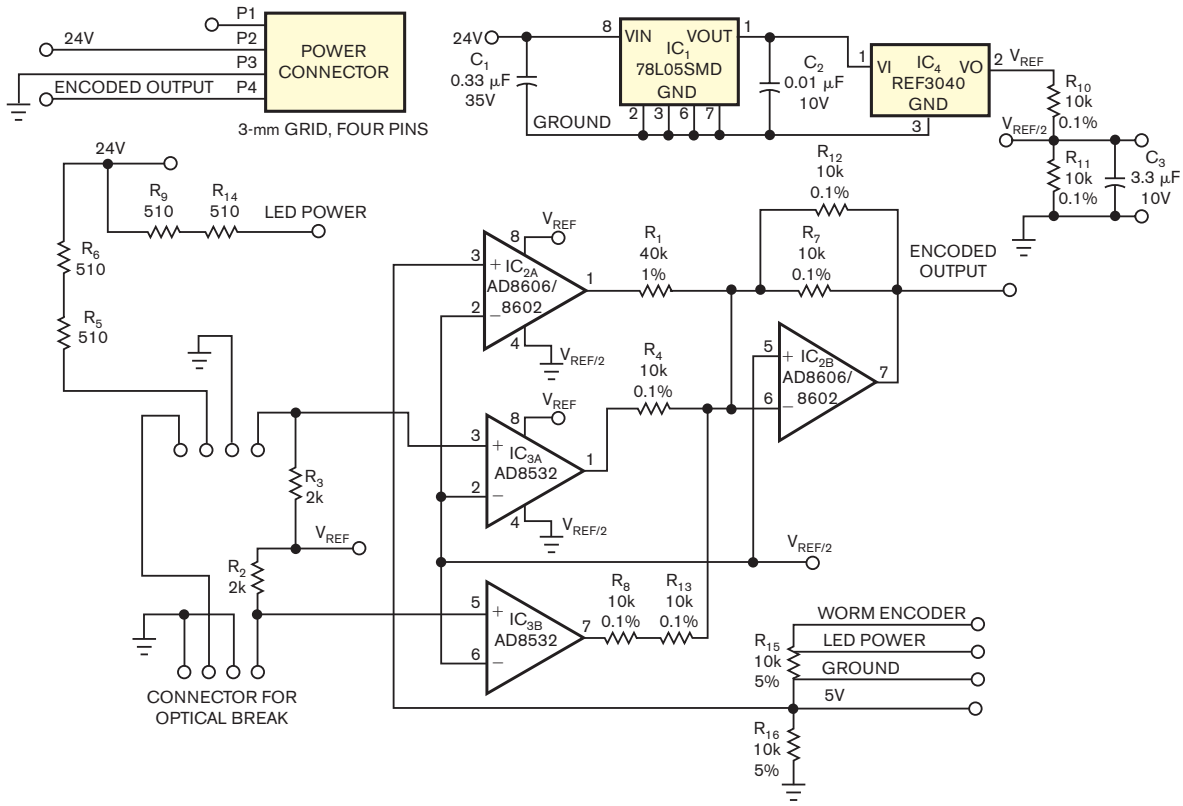

Figure 1 This application uses IC_4, a REF3040 voltage reference, which has an output tolerance of 0.2% yet costs only approximately $1.

Mutliplexing technique yields a reduced-pin-count LED display

Saurabh Gupta and Dhananjay V Gadre,
Netaji Subhas Institute of Technology, Dwarka, New Delhi, India

"Charlieplexing" as a method of multiplexing LED displays has recently attracted a lot of attention because it allows you, with N I/O lines, to control N×(N−1) LEDs (**references 1** through **5**). On the other hand, the standard multiplexing technique manages to control far fewer LEDs. **Table 1** lists the number of LEDs that you can control using Charlieplexing and standard multiplexing by splitting the available number of N I/O lines into a suitable number of rows and columns. **Table 1** also shows the duty cycle of the current that flows through the LEDs when they are on.

Clearly, Charlieplexing allows you to control a much larger number of LEDs with a given number of I/O lines. However, the downside of this technique is the reduced duty cycle of the current that flows through the LEDs; thus, to maintain a given brightness, the peak current through the LEDs must increase proportionately. This current can quickly reach the peak-current limit of the LED. Nonetheless, Charlieplexing is a feasible technique for as many as 10 I/O lines, allowing you to control as many as 90 LEDs. To control an equivalent number of LEDs using the standard

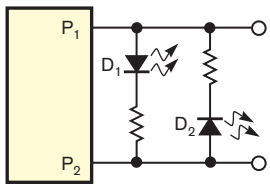

Figure 1 "Charlieplexing" with two I/O lines allows you to control two LEDs.

TABLE 1 NO. OF LEDs AND DUTY CYCLE

No. of I/O lines	Multiplexing-controlled LEDs	Duty cycle with multiplexing (%)	Charlieplexing-controlled LEDs	Duty cycle with Charlieplexing (%)
Two	Two	100	Two	50
Three	Three	100	Six	16.67
Four	Four	50	12	8.33
Five	Six	50	20	5
Six	Nine	33	30	3.33
Seven	12	33	42	2.4
Eight	16	25	56	1.78
Nine	20	25	72	1.38
10	25	20	90	1.11

TABLE 2 OUTPUT VOLTAGE

P_1	P_2	Voltage at node PR_1
0	0	V_{CC}
0	1	V_{CC}
0	Z	V_{CC}
1	0	0
1	1	0
1	Z	0
Z	0	$V_{CC}/2$
Z	1	$V_{CC}/2$
Z	Z	$V_{CC}/2$

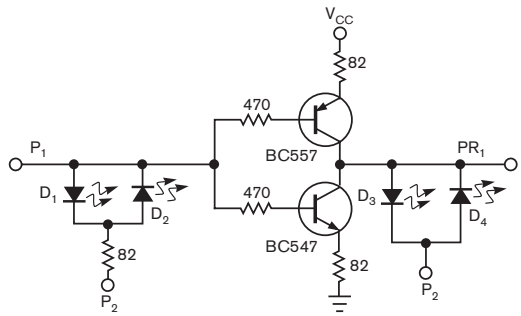

Figure 2 "GuGaplexing" with two I/O lines allows you to control four LEDs.

multiplexing technique would require 19 I/O lines.

This Design Idea proposes a modification to the Charlieplexing technique that allows you to control twice as many LEDs. Thus, the proposed method, "GuGaplexing," allows $2\times N\times(N-1)$ LEDs using only N I/O lines and a few additional discrete components (**Figure 1**). To turn on LED D_1 using the Charlieplexing method, set P_1 to logic one and P_2 to logic zero. To turn on LED D_2, set P_1 to logic zero and P_2 to logic one. **Figure 2** shows the proposed GuGaplexing scheme with two I/O lines controlling four LEDs. The GuGaplexing technique exploits the fact that each I/O line has three states: one, zero, and high impedance. Thus, with two I/O lines, states 00, 01, 10, and 11 of eight possible states control the LEDs.

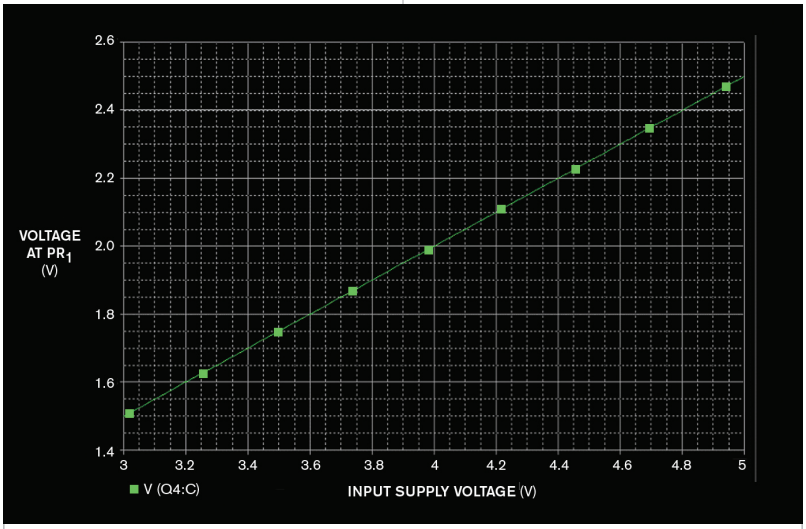

Figure 3 This graph plots the voltage at node PR_1 for various supply-voltage values when the input to the transistor pair is floating.

TABLE 3 I/O LINES AND PR_1 VOLTAGE

P_1	P_2	Voltage at node PR_1	LED that turns on
0	0	V_{CC}	L_3
0	1	V_{CC}	L_2
1	0	0	L_1
1	1	0	L_4
Z	Z	$V_{CC}/2$	None

Table 2 lists the voltage at the output of the transistor pair for various states of the two I/O lines, P_1 and P_2. The transistor pair comprises a BC547 NPN and a BC557 PNP transistor; matched transistor pairs are recommended. For N I/O lines, the GuGaplexing technique requires N−1 transistor pairs. **Table 3** shows the state of the I/O lines P_1 and P_2 and the voltage at node PR_1 to control the four LEDs. The circuit requires that the LED turn-on voltage should be slightly more than $V_{CC}/2$. Thus, for red LEDs with a turn-on voltage of approximately 1.8V, a suitable supply voltage is 2.4V. Similarly, for blue or white LEDs, you can use a 5V supply voltage. Modern microcontrollers, especially the AVR series of microcontrollers from Atmel (www.atmel.com), operate at a wide variety of supply voltages ranging from 1.8 to

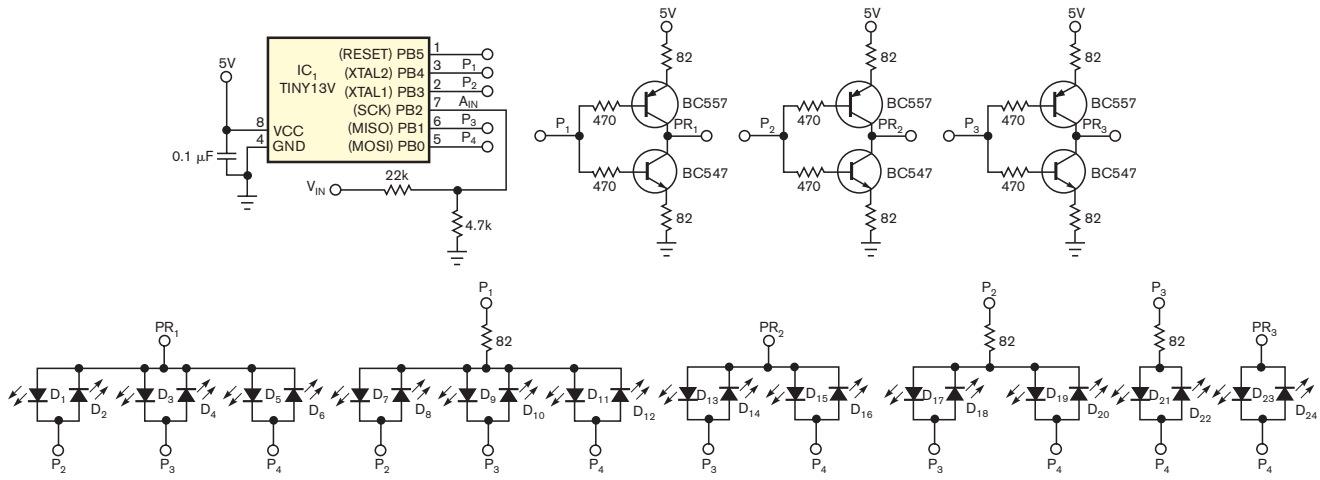

Figure 4 With the GuGaplexing technique, controlling 24 LEDs requires only four I/O lines and three sets of transistors.

5.5V, and this design uses a Tiny13 microcontroller to implement the GuGaplexing technique.

Figure 3 plots the voltage at node PR_1 for various supply-voltage values when the input to the transistor pair is floating. The Spice simulation ensures that the circuit would work properly to provide $V_{CC}/2$ at the PR_1 node for wide operating-supply-voltage values when the input is floating.

A 24-LED bar display validates the scheme in a real application (**Figure 4**). The display is programmable and uses a linear-display scheme for the input analog voltage. The input analog voltage displays in discrete steps on the 24-LED display. Controlling 24 LEDs requires only four I/O lines and three pairs of transistors. The system uses 5-mm, white LEDs in transparent packaging and a 5V supply voltage. The GuGaplexing implementation uses an AVR ATTiny13 microcontroller. The analog input voltage connects to Pin 7 of the ADC input of the Tiny13 microcontroller.

The control program for the ATTiny13 microcontroller is available with the Web version of this Design Idea at www.edn.com/081016di1. The source code is in C and was compiled using the AVRGCC freeware compiler. You can modify the source code to display only one range of input voltage between 0 and 5V. For example, it is possible to have a linear-display range of 1 to 3V or a logarithmic scale for input voltage of 2 to 3V. EDN

REFERENCES

1 Lancaster, Don, *Tech Musings*, August 2001, www.tinaja.com/glib/muse152.pdf.

2 "Charlieplexing: Reduced Pin-Count LED Display Multiplexing," Application Note 1880, Maxim, Feb 10, 2003, http://pdfserv.maxim-ic.com/en/an/AN1880.pdf.

3 Chugh, Anurag, and Dhananjay V Gadre, "Eight-Pin Microcontroller Handles Two-Digit Display With Multiple LEDs," *Electronic Design*, May 24, 2007, http://electronicdesign.com/Articles/ArticleID/15512/15512.html.

4 Gadre, Dhananjay V, and Anurag Chugh, "Microcontroller drives logarithmic/linear dot/bar 20-LED display," *EDN*, Jan 18, 2007, pg 83, www.edn.com/article/CA6406730.

5 Benabadji, Noureddine, "PIC microprocessor drives 20-LED dot- or bar-graph display," *EDN*, Sept 1, 2006, pg 71, www.edn.com/article/CA6363904.

Derive a simple high-current source from a lab supply

Roger Griswold and Alfredo Saab,
Maxim Integrated Products, Sunnyvale, CA

When electronic testing requires an adjustable current source, you must often build that piece of test equipment in the lab. You can easily make such a current source from a standard force-sense lab power supply (**Figure 1**). The circuit requires an additional power supply for the ICs and a separate control voltage. The feedback signal to the force-sense supply comes from a MAX4172 high-side current monitor from Maxim (www.maxim-ic.com). In the configuration in **Figure 1**, the circuit offers a 1-to-1 ratio of control voltage to load current (1A/V). **Figure 2** shows load current as a function of load resistance.

To change the voltage-to-current ratio, simply change the value of R_{SHUNT}; a lower value of R_{SHUNT} gives higher current and vice versa. The

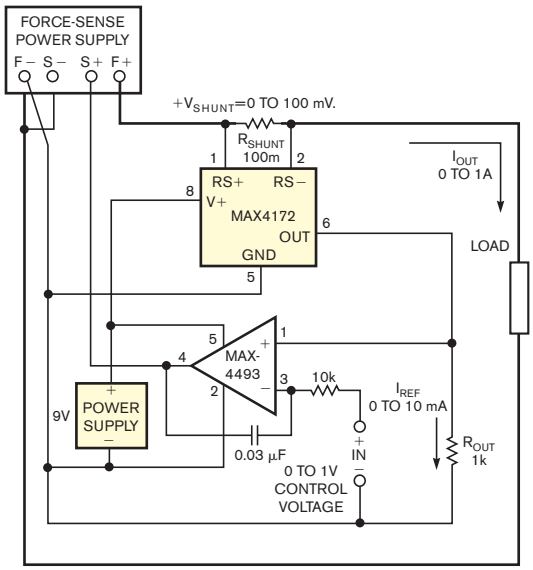

Figure 1 Adding these components to a standard force-sense lab supply makes a simple voltage-controlled current source. As configured, the circuit produces a control ratio of 1-to-1A/V.

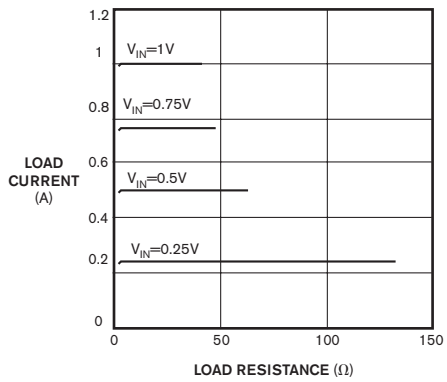

Figure 2 This graph shows load current versus load resistance for the circuit in **Figure 1**.

maximum allowed voltage of 150 mV between the RS+ and RS− terminals, the maximum positive RS voltage of 32V, and the maximum current capability of the force-sense supply all limit the output current of the supply.

Because voltage and current meters in the force-sense supply display inaccurate values while this circuit is operating, you should use external meters to monitor the load voltage and load current. Also, be aware that, if you remove the load so that the output current is 0A, the open-circuit voltage of the force-sense supply goes to the maximum value it can generate. EDN