design ideas

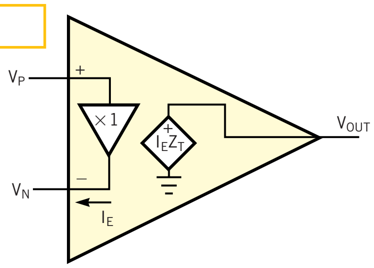
Edited by Bill Travis

RF oscillator uses current-feedback op amp

Victor Koren, Tioga Technologies Ltd, Tel Aviv, Israel

A CURRENT-FEEDBACK AMPLIFIER is a well-known component with many uses. Its basic block diagram shows that its input stage is a voltage follower—in practice, a symmetrical emitter follower (**Figure 1**). The configuration samples the output current, converts it to voltage across a large impedance, and amplifies it to the output using a high-power, low-output-impedance amplifier. The idea is to use the amplifier's input stage as a voltage follower in a basic Colpitts oscillator. This circuit uses the noninverting input of the current-feedback amplifier as the follower input and the inverting input of the amplifier as the follower output. You use the output amplifier to obtain a relatively high-power buffered output. The circuit in **Figure 2** shows a basic Colpitts oscillator that uses the amplifier's input-voltage follower as the active element of the oscillator.

Figure 1



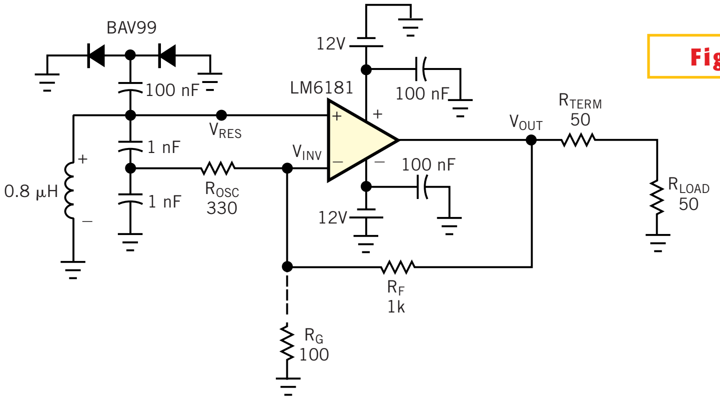
In a typical current feedback amplifier, the input stage is a voltage follower.

Take note of two aspects of this oscillator circuit: First, back-to-back diodes connect across the resonator to limit the oscillations to a specific level, thus maintaining the linearity of the voltage follower. Second, the voltage follower output connects to the resonator tap through resistor R_{OSC} to improve the linearity and define the feedback magnitude. The value of R_{OSC}, 330Ω, lets you obtain soft clipping operation of the diodes across the resonator (V_{RES}=1V p-p, which is 0.5V peak across each diode). **Figure 3** shows V_{RES}, the measured voltage at the top of the resonator. R_F is the amplifier's feedback resistor; the amplifier's manufacturer recommends its value. This design uses the LM6181 from National Semiconductor (www.national.com), and the value of R_F is 1 kΩ.

It is easy to calculate the output voltage: V_{RES}=1V p-p, and $V_{INV}=V_{RES}$=1V p-p. The voltage-buffer gain is unity: $V(R_{OSC})=V_{INV}-V_{RES}/2$. The voltage at the resonator tap is $V_{RES}/2$, because the resonator capacitors are equal in value. $V(R_{OSC})=V_{RES}-V_{RES}/2=0.5V$ p-p. $I(R_{OSC})=V(R_{OSC})/R_{OSC}$. $I(R_F)=I(R_{OSC})$. The negative feedback nulls the amplifier's inverting-input current. $V_{OUT}=V(R_F)+V_{INV}=R_F\times I(R_F)+V_{INV}=1000\times(0.5/330)+1=2.51V$ p-p. If you need more voltage, you can add R_G—in this case, 100Ω—from the inverting input to ground. $I(R_G)=V_{INV}/R_G$. Now, the current through R_F is the sum of the currents through R_{OSC} and R_G. So, $V_{OUT}=V(R_F)+V_{INV}=R_F\times I(R_F)+V_{INV}=1000\times(0.5/330+1/100)+1=12.51V$ p-p. **Figure 4** shows the measured output voltage.

The LM6181's maximum output current is 100 mA, so it can easily drive a current of ±63 mA p-p (±6.3V/100Ω) into a total load of 100Ω (50Ω output-termination resistor and 50Ω load resistor). The voltage across the 50Ω load is 3.15V peak, or 2.23V rms, which is close to 20 dBm (100 mW). This power level can directly drive high-level diode double-balanced mixers, or it can drive a higher power amplifier while delivering a clean sinusoidal waveform. You can modify the resonator circuit to accommodate differ-

Figure 2



This Colpitts oscillator uses a current-feedback amplifier to provide a clean sinusoidal output.

Publish your Design Idea in *EDN*. See the What's Up section at www.edn.com.

ent tuning elements. You can use the circuit as a crystal oscillator by changing the inductor to a crystal and changing the resonator capacitors to an appropriate value, such as 2×68 pF. You need a high-value, such as 10-kΩ, bias resistor from the noninverting input to ground to provide bias current to this input.

Is this the best Design Idea in this issue? Select at www.edn.com.

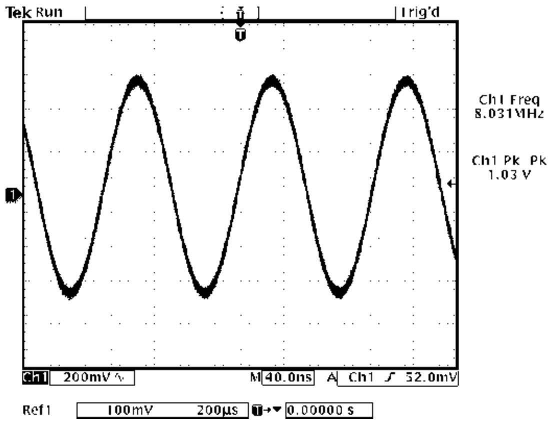


Figure 3 This clean sinusoid is the signal at the top of the resonator, V_{RES}, in Figure 1.

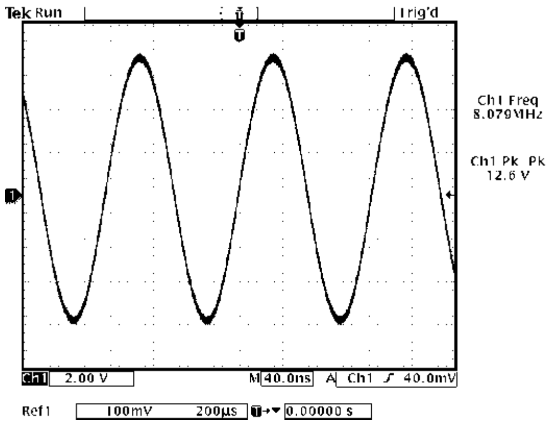


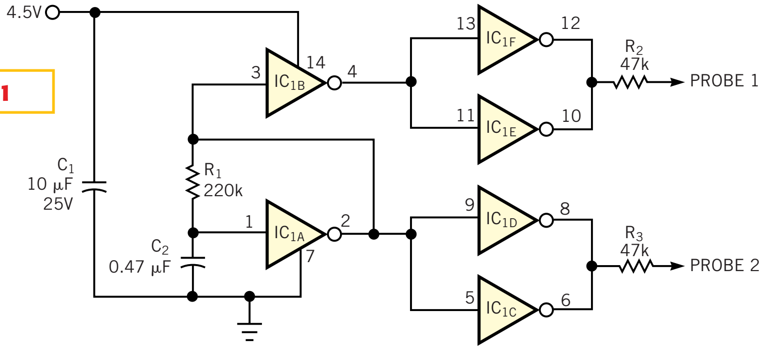
Figure 4 The Colpitts oscillator in Figure 2 produces a pure sinusoidal output.

Simple tester checks LCDs

D Prabakaran, NL Polytechnic College, Tamil Nadu, India

Manufacturers of electronic equipment use LCDs for calculators, watches, mini-videogames, and pagers, for example. In comparison with LED-based displays, which consume power on the order of tens of milliwatts, an LCD consumes only a few microwatts. The LCD thus saves power by a factor of approximately 1000. Checking an LED is as simple as checking a semiconductor diode but, in the case of LCDs, involves some added complexity. An LCD requires an ac electric field to excite the organic compound in the display. Applying a dc voltage could permanently damage the LCD. The circuit in **Figure 1** is a simple configuration to test the performance of an LCD. The circuit produces biphase square waves with negligible dc content. The circuit is based on a CD40106 hex Schmitt-trigger inverter. The circuit comprises an oscillator, IC_{1A}; a phase splitter, IC_{1B}; and a pair of buffer/drivers comprising IC_{1C}/IC_{1D} and IC_{1E}/IC_{1F}.

Figure 1



This simple circuit tests LCDs by producing a biphase square wave with no dc component.

The buffers and drivers connect to test probes through 47-kΩ series resistors, which protect the IC in the event of short circuits. With the component values shown in **Figure 1**, oscillator IC_{1A} provides a square-wave frequency of approximately 45 Hz. The circuit can operate from 3 to 5V. To test any segment of an LCD, touch the backplane using either of the two test probes while touching the segment with the other probe. If the segment under test is operational, it will light up. If the LCD under test is a multiplexed type, then all segments, which are connected, will glow if they are operational. Usually, the rightmost or leftmost connection is the backplane of the LCD. If it is not, you have to find it by trial and error.

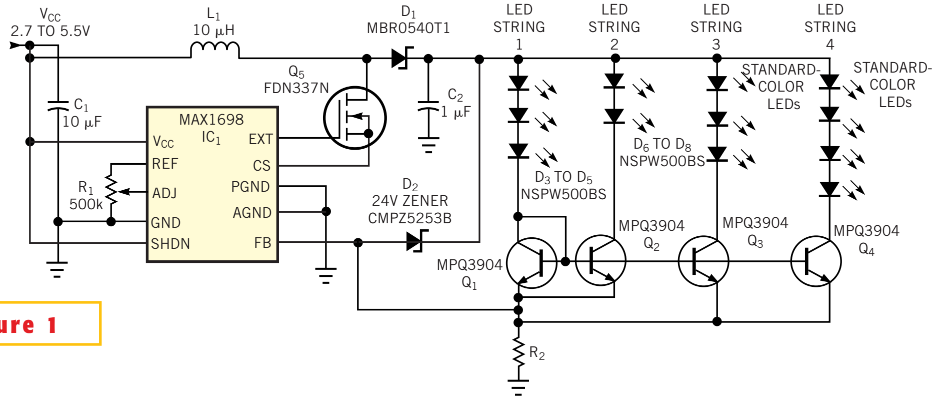
Is this the best Design Idea in this issue? Select at www.edn.com.

Circuit drives mixed types and quantities of LEDs

Mark Pearson, Maxim Integrated Products, Sunnyvale, CA

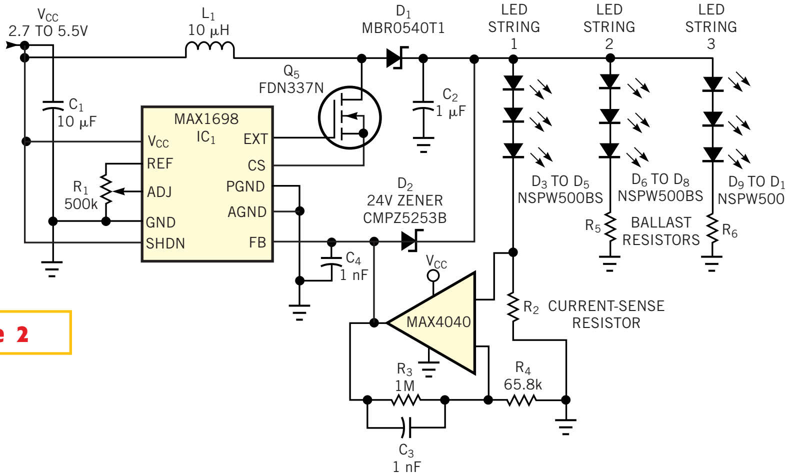
Portable systems often use LEDs of different colors and in varying quantities of each color. Some examples are white for the display backlight, green for keypad illumination, and red for power. Typically, the LEDs derive power from at least two power supplies: one for "standard" LEDs (red and green) and one for white LEDs. (White LEDs exhibit a higher forward voltage.) The keypad and other indicator LEDs have current-limiting resistors associated with them. To eliminate these resistors and drive groups of dissimilar LEDs from the same source, you can regulate the current through multiple strings. Four strings of varying LED types derive power from a single power source (**Figure 1**). The circuit mixes LEDs of different forward-bias requirements, yet keeps the loads reasonably well-balanced through use of a current mirror comprising transistors Q$_1$ through Q$_4$. It also eliminates the need for a separate current-limiting "ballast" resistor on each LED or string of LEDs and provides a common control point (IC$_1$'s ADJ pin) for adjusting the LED intensities.

Figure 1



In this LED-driver circuit, a switching converter, IC$_1$, and associated components lets you mix LED quantities and types.

Figure 2



Modifying Figure 1 as shown reduces the overall power dissipation in a standard application.

Transistors Q$_2$ through Q$_4$ mirror the current in the diode-connected transistor, Q$_1$. Note that the Q$_1$ current-set string (LEDs D$_3$ through D$_5$) should have an equal or larger voltage than that of subsequent LED strings. (If it doesn't, the current-mirrored strings may have too little voltage overhead to function properly.) You can easily meet that requirement in the first string by placing either LEDs with larger forward voltage drops, such as the approximate 2.8 to 3.7V range of white LEDs, or more similar LEDs. Then, the circuit can easily accommodate the subsequent strings with lower voltage burdens. The matched-transistor current mirrors maintain a constant and equal current in all LEDs, regardless of quantity and type. That configuration allows the use of a single power supply and a single point for adjusting LED brightness. Any power difference between the reference string and a mirrored string dissipates in the current-mirror transistor for that string: P_{MAX} (transistor)=(V_{OUT}−300mV−V_{LEDs})×I_{LEDMAX}. The current-sense resistor value is R_2=300 mV/I_{LEDMAX}, where I_{LEDMAX} is the sum of currents from all the strings. (For a comprehensive circuit and parts list, refer to Maxim's MAX1698 (www.maxim-ic.com) EvKit data sheet.)

When driving the same LEDs without the current mirror, you can reduce power dissipation in the sense resistor and ballast resistors by substituting a micropower op amp across the current-sense resistor (**Figure 2**). That circuit improves efficiency by reducing the resistor values and their associated loss. Increasing the gain of the current-sense signal by approximately 16 allows an equivalent reduction in the value of R$_2$ and the ballast resistors. A typical value for R$_2$ is 15Ω, which represents a loss of 18 mW: (20 mA)^2×15Ω for each of three resistors. If R$_2$=R$_5$=R$_6$=0.931Ω, then the resistor power loss drops to 1.12 mW. The op amp draws only 20 μA maximum, which represents a dissipation of 100 μW.

Is this the best Design Idea in this issue? Select at www.edn.com.

MOSFET serves as ultrafast plate driver

Clive Bolton, Bolton Engineering Inc, Melrose, MA

The circuit in **Figure 1** provides a 20-MHz square wave across a set of highly capacitive ion-deflection plates in an experimental instrument. To get the required deflection, the plate voltage must be 20 to 30V, much higher voltage than conventional logic or driver families can provide. To minimize artifacts, the rise and fall times must be very fast, with a minimum of overshoot and ringing. Identical circuits, phased 180° apart, drive the plates. The driver uses a Directed Energy (www.directedenergy.com) DEIC420 high-speed MOSFET gate driver to drive a 1000-pF capacitive load from 0 to 25V in less than 5 nsec. With smaller loads of a few hundred picofarads, the rise time decreases to approximately 3 nsec. Series resistors R_1 and R_2 control the output rise and fall times, allowing you to trade off the rise and fall times against overshoot and ringing. A high-speed Analog Devices (www.analog.com) ADUM1100BR ferromagnetic signal isolator prevents system ground loops by providing dielectric isolation for the input signal; you could also use high-speed optocouplers. A low-power MC78L05CD regulator provides power for the signal-isolator output stage.

A snubber network, composed of a thin-film, high-power resistor R_3 in a TO-220 package and high-quality NP0 capacitors C_1 and C_2, terminates the load at the plates. You empirically determine snubber values by observing the radiated field on an RF spectrum analyzer using a passive RF probe. You “tune” the snubber network to reduce higher order signal harmonics. Note that placing an oscilloscope probe on the outputs significantly increases the observed higher order harmonics, indicating that adding the probe to the circuit increases ringing and overshoot. The DEIC420 is mounted in a high-speed, high-power package that minimizes lead inductance. The part requires multiple bypass capacitors at each of its power pins. You should choose the capacitors so that their self-resonant frequencies do not significantly overlap. Having a full ground plane and using high-speed and RF-signal-layout techniques are critical to the proper operation of this circuit. The input must be well-isolated from the output. Double-pulsing, ringing, and even oscillation may occur if you don’t strictly follow these practices. The tracks or cabling between the driver and the load should be impedance-controlled and should be as short as possible. The DEIC420 requires good heat-sinking when you operate it at high speeds and high voltages. When operating at 20 MHz from a 25V supply, the two drivers and snubber together dissipate 130W.

Is this the best Design Idea in this issue? Select at www.edn.com.

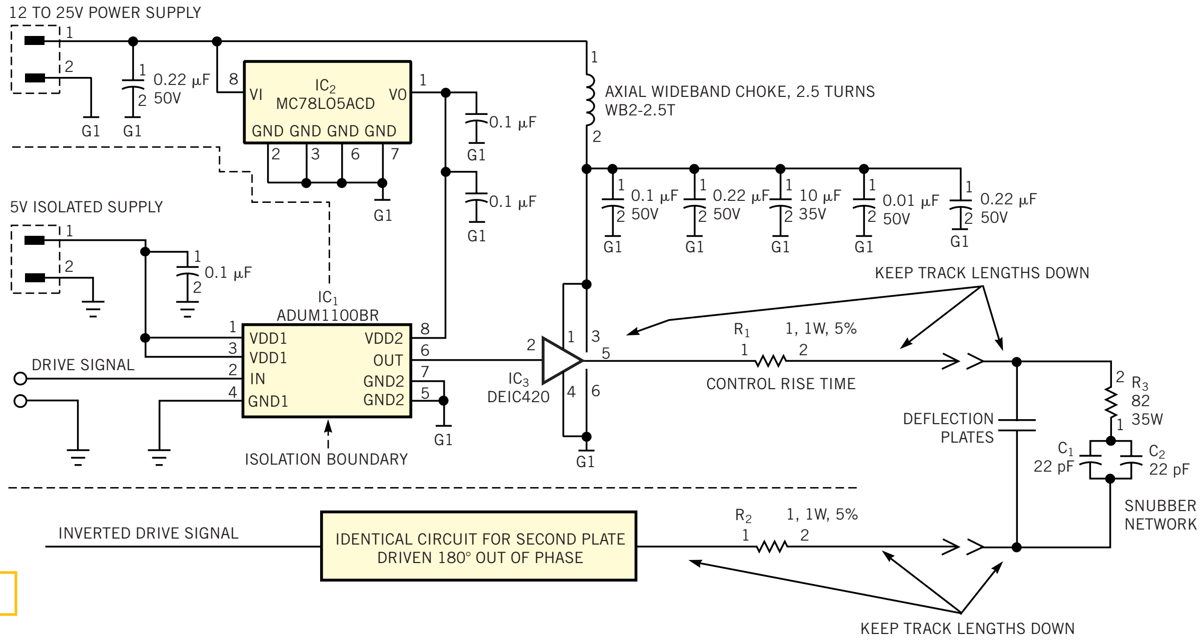


Figure 1

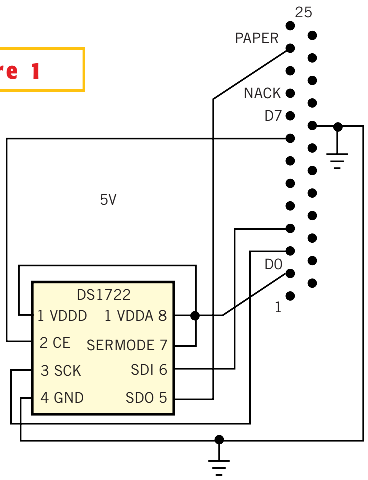
You can use a high-speed, MOSFET-driver IC to drive ion-deflection plates.

Parallel port provides high-resolution temperature sensing

Martin Connors and Mike Foote, Athabasca University, AB, Canada

High-resolution temperature sensing at low cost is possible using only one chip attached to the PC's parallel port (**Figure 1**). The Dallas Semiconductor (www.dalsemi.com) DS1722 digital thermometer allows measurement resolution as fine as 0.0625°C in digital form and with linear response. The accuracy specification is only 2°C, but you can improve this figure by careful calibration. Moreover, the accuracy spec is unimportant in applications in which you measure only changes in temperature or in which you must closely maintain a noncritical temperature. The measurement range is −55 to +120°C, the part can use either three-wire or SPI interface, and the cost is approximately $1. The eight-pin part is available in SO or µSOP packaging and in large quantities as a flip-chip measuring only about 1 mm sq.

Figure 1



The DS1722 connects to a PC's parallel port through a male DB-25 connector, seen from the pin side.

In this application, the chip attaches directly to the PC's parallel port through a male DB-25 connector. Because the device draws a maximum of 0.5 mA, the port can supply the power, and its supply range tolerates variations in voltage levels that may exist on varying ports. The chip is in SPI mode with the SCK

LISTING 1—TURBO C FOR DATA-TRANSFER CYCLE

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <process.h>
#include <alloc.h>

#define VDD_ON  0x01  /* power to PIC through VDD_ON on pin 2 (D0) */
#define SCK        0x02  /* serial clock for SPI provided by PC      */
#define PCSDO      0x04  /* serial data out from PC on Data bit 2     */
#define PCSDI      0x20  /* note this is on Status register (bit 1)   */
#define SSOUT      0x40  /* ~CE, active high, not low as for PIC SPI  */
#define MCLR_HI 0x80  /* MCLR* on pin 9 (Data bit 7) normally high */
#define OPERATE VDD_ON | MCLR_HI  /* normal operation of DS1722    */
/* adjust these to match the CPU speed */
#define DELTIME 10000          /* settling time after transfers */
#define SECDELA 10000000         /* to get about 1 s sampling    */

void dodelay(long);
void outportd(unsigned char);
int dport,sport;

void main(void)
{
unsigned char LSB,transfer(unsigned char,unsigned char);
char MSB; /* note this is signed */
void outportd(unsigned char outbyte);
int i,j,it;
float T;

/* LPT1 port addresses */
if(!(dport = peek(0x40,0x08)))
 { printf("\n\n\nLPT1 not available... aborting\n\n\n"); exit(1); }
sport = dport + 1; /* status port address */

/* Initialize the Printer DATA Port for PIC operation */
/* Includes putting SCK in the neutral 0 position: ~ is bitwise negation */
outportd(OPERATE&~SSOUT);

printf("hit key to stop list\n");
transfer(0x80,0xE8);   /* initialize DS1722 */
for(j=0;j<20;j++)
 { if(kbhit()) break;
   printf(" config %X: ",transfer(00,0));   dodelay(SECDELA);
   printf(" LSB %X: ",LSB=transfer(0x01,0)); dodelay(SECDELA);
   printf(" MSB %X: ",MSB=transfer(0x02,0)); dodelay(SECDELA);
   printf("T=%10.4f\n",T=MSB+(float)LSB/256.);
 } /* for loop */
}

unsigned char transfer(unsigned char outbyte,unsigned char outdata)
{ /* output address byte on Data2, data byte, getting inbyte on Status5 */
  unsigned char outmask,inbyte,statusmask;
  int ibit;
  inbyte=0x00;
/* raise SSOUT for 2 byte transfer, SCK also in lowered neutral position */
  outportd(OPERATE|SSOUT);
  for(ibit=0;ibit<8;ibit++)                        /* output outbyte  */
   { outportd(OPERATE|SCK|SSOUT);                    /* raise clock SCK */
     outmask=outbyte&0x80; outbyte=outbyte<<1;
     if(outmask)outportd(OPERATE|SCK|PCSDO|SSOUT);   /* output of 1 or  */
       else outportd(OPERATE|SCK|SSOUT);             /* of 0 on PCSDO   */
      if(outmask)outportd(OPERATE|PCSDO|SSOUT);      /* lower clock SCK */
       else outportd(OPERATE|SSOUT);                 /* retaining data  */
   }
   for(ibit=0;ibit<8;ibit++)                        /* output outdata  */
    {                                   /* & input inbyte  */
     outportd(OPERATE|SCK|SSOUT);                    /* raise clock SCK */
     outmask=outdata&0x80; outdata=outdata<<1;
     if(outmask)outportd(OPERATE|SCK|PCSDO|SSOUT);   /* output of 1 or  */
      else outportd(OPERATE|SCK|SSOUT);              /* of 0 on PCSDO   */
     if(outmask)outportd(OPERATE|PCSDO|SSOUT);       /* lower clock SCK */
      else outportd(OPERATE|SSOUT);                  /* retaining data  */
     statusmask=inportb(sport);                     /* read status port*/
     statusmask=statusmask&PCSDI;                    /* mask input line */
     inbyte=inbyte<<1; inbyte=inbyte&0xFE; inbyte|=statusmask>>5;
   }
/* lower SSOUT at end of 2 byte transfer, lower SCK to neutral position  */
  outportd(OPERATE&~SSOUT);
  return inbyte;
}

void outportd(unsigned char outbyte)
{  dodelay(DELTIME); outportb(dport,outbyte); dodelay(DELTIME); }

void dodelay(long deltime)
{ long i; for(i=0; i<deltime; i++) ; }
```

clock signal supplied by the PC; in this way, data-transfer timing is noncritical. A simple Turbo C program (**Listing 1**) running in DOS mode effects the data-transfer cycle in the PC, whereas the transfer is automatic in the chip upon reception of SCK. The routine reads a low byte and a signed high byte and creates a floating-point value by simply adding the low byte, divided by 256, to the high byte. In the highest resolution mode, which this design uses, a data read can occur only every 1.2 sec, and you should adjust the timing loops accordingly. You may also need to adjust the settling time, DELTIME, depending on the speed of the PC you use. The sample program prints the bytes transferred as well as the temperature, and you can easily modify it. The data sheet explains the use of the configuration register and changes to make if you need a higher data rate with lower resolution. You can download **Listing 1** from the Web version of this Design Idea at www.ednmag.com.

The data transfer takes place beginning with the write of an address byte to the chip's SDI in the order A7 to A0 (high bit to low bit). If A7 is high, a write takes place; otherwise, a read occurs. For a write, D7 to D0 route to the chip's SDI. For a read, D7 to D0 are available on the chip's SDO. The program always uses both SDI and SDO and ignores whichever it doesn't need. For example, data goes to the chip's SDI even during a read, but the chip ignores this data. Each byte transfers as 8 bits, and each transfer involves the following steps:

1. The PC raises D1/SCK and places 0 or 1 on D2 for the chip's SDI.
2. The PC then reads PAPER.
3. Finally, the PC drops D1/SCK.

This action repeats for each bit of the pair of bytes being transferred (one in, one out). By using the other parallel port's output pins as chip selects, you could string together several devices. You can also use these pins to control a heater by use of a switching transistor or an SCR. With this scheme, you can achieve high-resolution temperature control with minimal parts and a simple program. Alternatively, if you need only low accuracy, you can implement a very-low-cost thermostat with this part.

Is this the best Design Idea in this issue? Select at www.edn.com.