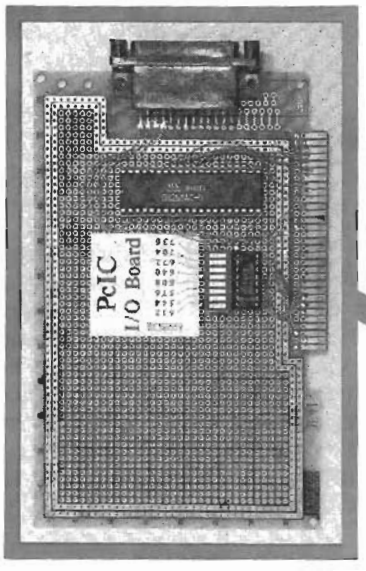
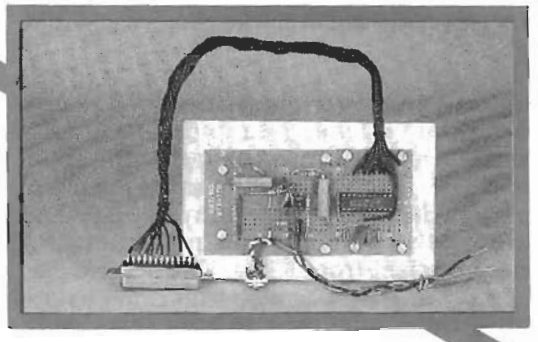
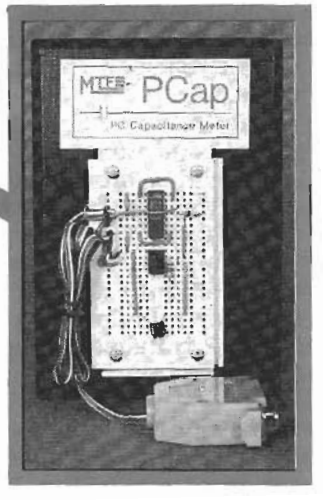
EXPERIMENTING WITH ADC FOR YOUR PC







JAMES J. BARBARELLO

Build our low-cost data-capture system and let your PC measure temperature, displacement, and other quantities.

IN PREVIOUS ARTICLES, WE INTRODUCED several general-purpose analog and digital I/O systems for the PC. (See "Experimenting With PC-Based Test Equipment" in May, June, and July of 1991.) The first installment described a test instrument for measuring capacitance; the second described a digital R/C meter. Both devices interfaced to the PC via a standard parallel port. The third installment described a simple PC expansion card, the PC IO, that adds 24 digital I/O lines by means of an 8255 Parallel Input/Output (PIO) device, the standard in the PC architecture.

In this article, we'll build a very low cost analog-to-digital converter (ADC) interface that allows you to measure temperature, displacement, audio signals, and any other 0–5 volt analog signal. Our ADC builds off the PC IO: By adding an amplifier and ADC circuit, and some simple BASIC software, you can capture and display data and log it to your PC at a rate of 1000 (or more, depending on the speed of your PC) data points per second.

This article shows you how to use the ADC to build an accurate temperature sensor, but it's easy to modify the circuit to accept other types of analog input devices.

Circuit theory

As shown in the block diagram of Fig. 1, the circuit consists of two functional blocks: a signal conditioner and an A/D converter. The signal conditioner is a variable-gain amplifier with an adjustable DC offset that allows you to calibrate the circuit for a variety of sensors. To understand why calibration is necessary, let's look at the IC that does the actual analog-to-digital conversion, a standard eight-bit device called the ADC0804. Unlike a traditional voltmeter, the ADC0804 responds to AC voltage changes very quickly, in fact at a rate greater than 1000 per second. The ADC0804 converts each sample to digital form, after which a computer may read the digital outputs for display or subsequent analysis.

The ADC0804 accepts an analog input of 0–5 volts DC and converts it to a binary number between 0 and 255. With a maximum range of 5 volts, and 256 steps between 0 and 5, resolution is 5/256 = 0.0195 volts, or almost 20 millivolts. So for any analog input voltage between 0.0000 and 0.0195, the ADC0804 will produce a binary 0 (00000000); for any voltage between 0.0195 and 0.0390, a binary 1 (00000001), and so on.

Twenty millivolts may seem like more than enough resolution, but what if you wanted to measure a signal with a maximum value of 40 millivolts? You'd only be able to distinguish two values in the given range. That's where the signal-conditioning portion

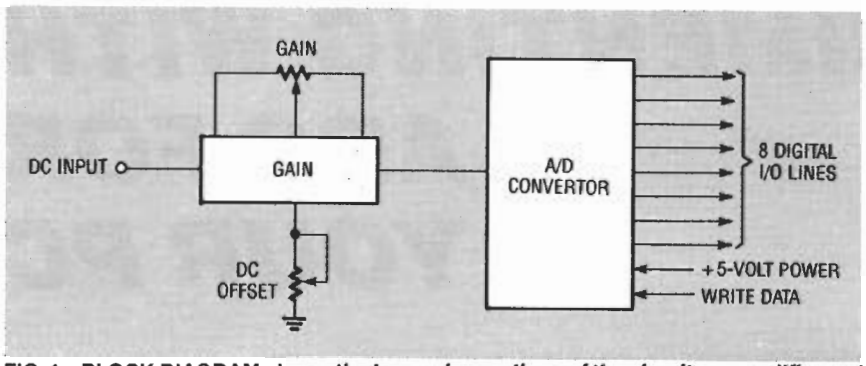


FIG. 1—BLOCK DIAGRAM shows the two major sections of the circuit, an amplifier and the ADC proper.

of the circuit comes in. By amplifying the signal so that it has an effective range close to the maximum range of the IC, we can break it down into much finer increments, and then let software scale it back to the appropriate value.

The DC offset portion of the signal conditioner lets us cancel any steady-state DC voltage and measure only the change in the DC voltage. That's required for the temperature sensor probe, which is built around the base-collector junction of a standard 2N2222 transistor. More on that in a moment.

The circuit

Referring to the schematic diagram shown in Fig. 2, IC1 is a standard 1458 (dual 741) op-amp. One op-amp (IC1-a) amplifies the input voltage by the ratio −R3/R6. With the values shown, that's a voltage gain of −10. Resistor R8 ensures minimum DC offset from IC1-a. The second op-amp (IC1-b) also functions as an amplifier, but in this case, voltage gain is −(R5 + R11)/R7, where R11 is a 15-turn, 10K potentiometer that allows the gain to be adjusted between values of −1 and −2. Like R8, R9 ensures minimum DC offset from IC1-b. The two inversions in the op-amps result in a non-inverted output signal.

The voltage divider consisting of R2 and R12 allows insertion of a DC offset voltage of 0–+2.5 volts DC. This can offset any positive quiescent DC voltage from an input device.

The ADC is IC2; it is configured for a free-running mode that samples the input signal (pin 6) continuously. To ensure that the A/D is initialized properly, the software drives pins 3 ($\overline{WR}$) and 5 ($\overline{INTR}$) low momentarily on start-up. The eight digital outputs (IC2 pins 12–18) drive the PIO directly, by way of P1.

A pair of 9-volt batteries supplies power for the op-amps; the host PC supplies +5-volt power for the voltage-divider circuit and IC2 via pin 25 of P1. Doing so ensures a more stable reference voltage than if the batteries were used. Of course, you're free to use a dual-polarity power supply in place of the batteries.

Temperature sensing

A standard 2N2222 transistor can readily serve as a $0.20 temperature sensor. Referring to Fig. 3-*a*, note that the emitter and base of Q1 are shorted together. That connection provides a diode—the base/collector junction. When power is applied across the junction, we would expect a constant drop of about 0.7 volts across it. The term *about* is important here, because the actual drop depends on the temperature of the junction.

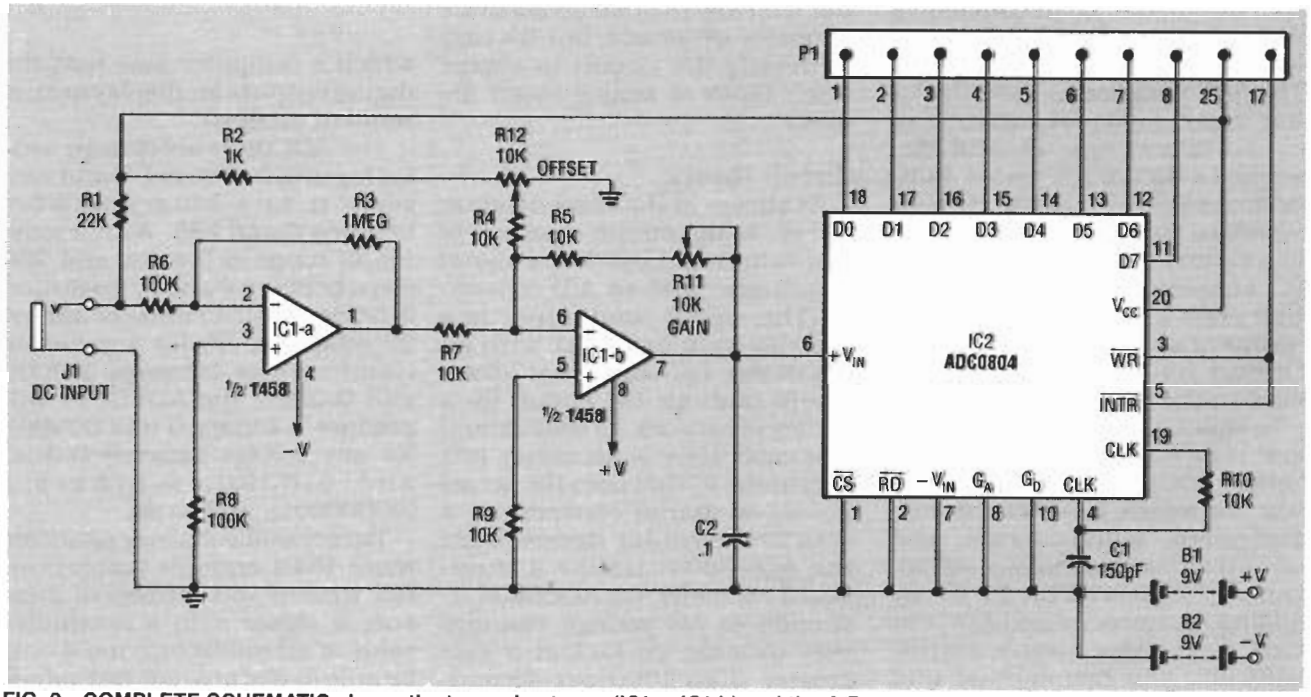


FIG. 2—COMPLETE SCHEMATIC shows the two gain stages (IC1-a, IC1-b) and the A/D converter (IC2).

PARTS LIST

All resistors are ¼-watt, 5%, unless otherwise noted.
R1—22,000 ohms
R2—1000 ohms
R3—1 megohm
R4, R5, R7, R9, R10—10,000 ohms
R6, R8—100,000 ohms
R11–R13—10,000 ohms, 15-turn potentiometer

Capacitors
C1—150 pF, ceramic disk (any value between 150–330 pF OK)
C2—0.1 μF, ceramic disk

Semiconductors
IC1—5558 or 1458 dual op-amp, 8-pin DIP
IC2—ADC0804 or ADC0803 8-bit A/D converter
Q1—2N2222 or PN2222 general purpose transistor

Other components
B1, B2—9-volt battery
J1—miniature (⅛-inch) phono jack
P1—DB-25 male connector
Miscellaneous: 24-gauge stranded wire, perforated construction board, 9-volt battery clips, housing for probe, shielded cable.

Note: The complete PC IO Board (with PC board, and all components) is available for $39.95 (part #PCIO). The ADC0804 and a calibrated PN2222A temperature sensor transistor are available for $8.00 (part #ADC). Software, including compiled and source code versions with continuous and interval sampling and data logging/listing is available for $8.00 (part #ADC-S). Specify part numbers and send check or Money Order to JJ Barbarello, 817 Tennent Road, Manalapan, NJ 07726. The author will be glad to answer any questions, but they must be accompanied by a self-addressed stamped return envelope.

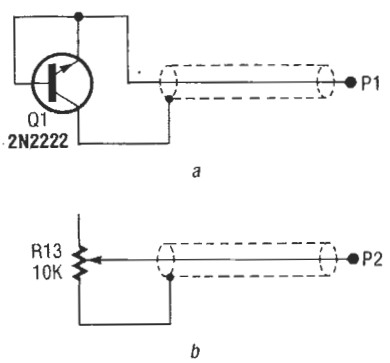


FIG. 3—DRIVE THE ADC with a temperature-sensing transistor (*a*) or a calibration potentiometer (*b*).

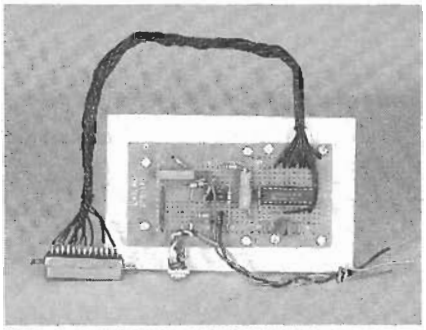


FIG. 4—THE AUTHOR'S PROTOTYPE was built on several pieces of scrap perforated construction board.

Silicon transistors used in this way have a temperature coefficient of about −2mV/°C. That is, at 0°C, the drop would be about 0.6 volt. As the temperature increases, the drop decreases. At 100°C the drop will be about 0.4 volt. Although every transistor has a slightly different temperature coefficient, all transistors vary linearly with temperature. So by calibrating for the particular transistor used, it's possible to obtain an accurate yet low-cost temperature sensor.

Construction

Assembly is not critical; you can build the circuit on perforated construction board as shown in Fig. 4. We recommend using sockets for the IC's.

To connect to the PC IO board, connect eleven one-foot lengths of wire to the appropriate points on IC2, and the other ends to the corresponding pins of P1, a standard DB-25 male connector.

Next, build the temperature probe. Use either a 2N2222A (metal case) or PN2222A (plastic case) and a length of shielded cable. Twist the base and emitter leads together, and solder them to the center conductor of the cable. Connect the collector to the shield. To avoid shorts, cover the leads near the transistor with heat-shrink tubing. Mount the transistor in a cylindrical case (a hollowed-out ballpoint pen body, for example), making sure the case can withstand the temperature range you will be measuring. Fill the probe with silicone or epoxy. Then attach a miniature phono plug to the free end of the cable, making sure that the center conductor goes to the tip and the shield to the ring.

The software

The QuickBASIC program that reads the voltage output of the circuit and converts it to a temperature is shown in Listing 1; note that line numbers are included for reference only. (The software is also available on the RE-BBS, 516-293-2283, 1200/2400, 8N1, as a file called PCADMATE.LST.) The program requires several constants to work. Rather than store that information in the QB file, which would require recompiling every time we recalibrate, we store it in a sequential data file, TEMP.DAT, which contains the values we need, each separated by a comma. First comes the decimal I/O port address of the PC I/O card (described in July), followed by the voltage at the low temperature, the low temperature, the voltage at the high temperature, and the high temperature. (Voltages should be specified in volts and temperatures in °C.)

For example, if those values were 640, 4.1, 0, 0, and 100, TEST.DAT would contain

640, 4.1, 0, 0, 100

followed by a carriage return and line feed. The file can be created with any word processor; just remember to save it in ASCII or text format, not the word processor's native format.

Lines 2–4 of the program open TEMP.DAT, read the values, close the file, and then set up the 8255 on the PC IO card so that lines 1–16 are inputs, and 17–22 are outputs.

Lines 5–8 format the screen for a pleasing look.

Lines 9–17 are the real meat of the program, the measuring and display loop. Line 10 pulses the $\overline{\text{WRITE}}$ line low to obtain a reading. Line 11 then retrieves that reading into variable X. The program converts that number into a voltage (V) between 0 and 5.0. Next, line 12 calculates the Centigrade (cent) and Fahrenheit (faren) temperature values. Then lines 13–15 format and display the values. Line 16 pauses before the next sample is taken, and line 17 checks whether the Escape key has been pressed. If so, the program ends; otherwise, execution loops back to line 10.

Calibration

First create the data file (TEMP.DAT) with nominal values for port address, voltages, and temperatures (640, 4.1, 0, 0, 100); we'll fine-tune those values momentarily. Then run the program to initialize the PC IO card.

Next, connect the ADC circuit to the PC IO card, plug the probe into J1, and place the tip of the probe against a piece of ice. Using a digital voltmeter on a low range, measure the voltage across the tip and ring of the plug. Record the temperature (0°C) and the resultant voltage.

Pour some boiling water in a styrofoam cup, place the probe in the water, and repeat the process, recording both temperature (100°C) and voltage. Use the two voltage values to determine the temperature coefficient of your probe. For instance, if the 0°C reading were 552 mV and the 100°C reading were 342 mV, the temperature coefficient would be $(0.342 - 0.552)/(100) = -2.1$mV/°C.

Prepare a 10K potentiometer as shown in Fig. 3-*b*. At this point, the ADC board should be connected to the PC IO, the 9-volt batteries (or other power source) should be connected, and the BASIC program should be running. Connect a DVM across J1, set the calibration potentiometer so the value on the DVM equals the high value taken earlier, and adjust R12 for 0.000 volts, as shown on the PC's screen. Then set the calibration potentiometer so the value on the DVM equals the low value, and adjust R11 for a value (as shown on the screen) between 4.0 and 4.5 volts. The actual value doesn't matter, just the difference between the high and low values.

Check the high reading setting again to make sure it is still 0.0 volts, and recalibrate if necessary. Go back and forth between the two readings several times.

Now enter the correct values into TEMP.DAT. Make sure the file is stored in the same subdirectory as the program.

Now you're ready to use the probe. Just place it against the item to be measured, and hold it there until you get a steady temperature reading.

LISTING 1

```
REM**************************************
REM** ADCTEMP.BAS - V910629  *
REM** ADC0804 A/D IC & 2N2222 Temp Probe *
REM**************************************
1 CLS : DEFINT A, X: DEF SEG = 64
2 OPEN "TEMP.DAT" FOR INPUT AS 1
3 INPUT #1, add,lowvolt, lowval, hivolt, hival
4 CLOSE #1: OUT ADD + 3, 146
REM***************** SET UP SCREEN ********************
5 LOCATE 1, 23: PRINT "PcTEMP TEMPERATURE MEASURING SYSTEM"
6 LOCATE 2, 1: PRINT STRING$(79, 220): LOCATE 8, 32: PRINT
STRING$(16, 220)
7 FOR i = 9 TO 16: LOCATE i, 32: PRINT CHR$(219); SPACE$(14);
CHR$(219): NEXT
8 LOCATE 12, 33: PRINT STRING$(14, 220): LOCATE 16, 33: PRINT
STRING$(14, 220)
REM***************** SAMPLING LOOP ********************
9 again:
10 OUT add + 2, 0: OUT add + 2, 1:  REM:  Take A Sample
11 x = INP(add): v = x * 5 / 255
12 cent=hival - (v * (hival - lowval) / lowvolt):faren = 1.8 *
INT(cent) + 32
13 LOCATE 4, 33: PRINT USING "Output = #.## v"; v
14 LOCATE 10, 37: PRINT USING "###"; cent; : PRINT CHR$(248); "C"
15 LOCATE 14, 37: PRINT USING "###.#"; faren; : PRINT CHR$(248);
"F"
16 FOR i = 1 TO 500: NEXT
17 IF INKEY$ = CHR$(27) THEN END ELSE GOTO again
```

LISTING 2

```
IF CENT < 10.5 THEN OUT ADD+1, 1
IF CENT > 10.5 THEN OUT ADD+1, 0
```

LISTING 3

```
4 CLOSE #1: OUT ADD + 3, 146: OPEN "READING.DAT" FOR OUTPUT AS 1
16 FOR i = 1 TO 500: NEXT: OUTPUT #1, V
17 IF INKEY$ = CHR$(27) THEN CLOSE: END ELSE GOTO again
```

LISTING 4

```
OPEN "READING.DAT" FOR INPUT AS 1
DO WHILE NOT EOF(1)
INPUT #1, V: PRINT V
LOOP
CLOSE: END
```

Alternate ranging

It is possible to adjust the circuitry and the computer program to any temperature range you desire. Just recalibrate the circuit and adjust the constants in TEMP.DAT. For example, assume you want to measure temperatures between −35°F and +104°F.

First, determine the temperature coefficient as described above. Next, convert the desired temperature range from °F to °C using the formula $C = 5/9 \times (F - 32)$. In our example, +104°F is 40°C, and −35°F is −37.2°C. Then calculate the drop at those two temperatures using the following formula.

$$V_D = V_Z + (T \times TC)$$

where V_Z is the voltage drop at 0°C, T is the temperature in °C, and TC is the temperature coefficient.

At 40°C, V_D = 600 mV + (40°C × −2.1 mV/°C) = 600mV −84mV = 516mV.

At −37.2°C, V_D = 600 mV + (−35°C × −2.1 mV/°C) = 600 mV + 78.17 mV = 678.17 mV.

Now plug in the calibration potentiometer (shown in Fig. 3-*b*), set it to the high-temperature drop (516 mV), and adjust R12 until the output is 0.000. Then set the potentiometer to the low-temperature drop (678 mV), and adjust R11 for a value between four and five volts. Last, insert the temperature values and the high-temperature output voltage into the data file. Now the circuit is fully calibrated for the new temperature range.

Software modifications

You could use the probe as a low-temperature detector. Connect a sensitive 5-volt relay to pins 9 (+) and 23 (gnd) of P1. If the temperature goes below a

continued on page 82

continued from page 62

specified value, the relay will be energized. When the temperature rises again, the relay will be de-energized. For example, to trip at 10.5°C, add the lines shown in Listing 2 between lines 16 and 17 of the original program.

You can also change the value in the FOR-NEXT loop to alter the time between samples. For instance, on the author's system, changing the value to 5000 causes samples to be collected once every 10 seconds. The exact time depends on your PC's speed, so you'll have to experiment.

With a longer time between samples, you can log data samples to disk. To do so, change lines 4, 16 and 17 as shown in Listing 3.

To read and display the resultant data, run the program shown in Listing 4.

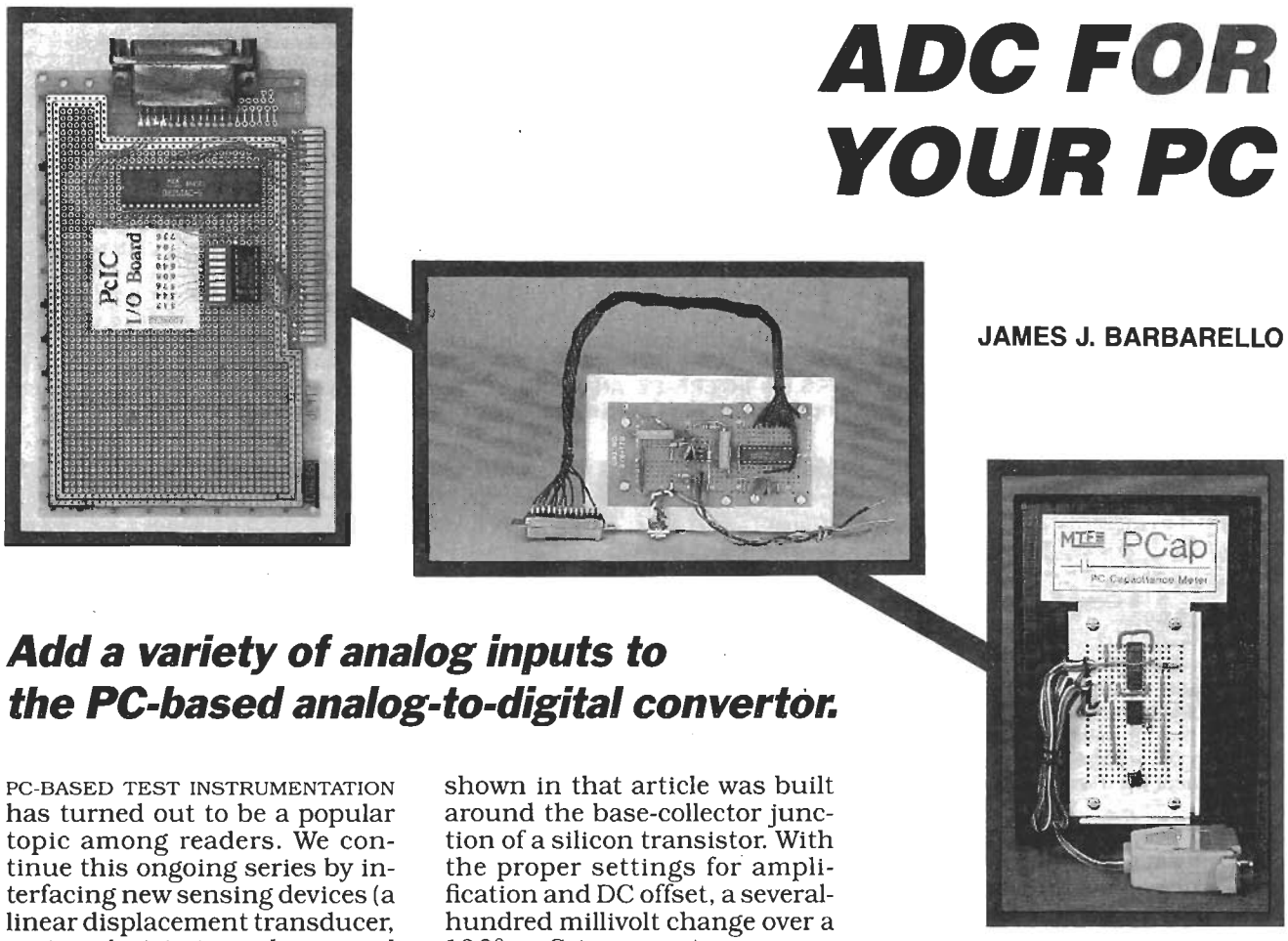
Other inputs

The signal conditioner circuit can accept other input sources, including linear-displacement potentiometers (LDP's), piezoelectric transducers, and microphones. An LDP is a specialized potentiometer whose wiper moves linearly instead of in the normal circular fashion. Usually these extremely accurate devices, which can be purchased from surplus electronics suppliers, are mechanically connected to a moving mechanism. They convert linear motion into a corresponding resistance. If a DC voltage is placed across the LDR, the output at the wiper is a voltage proportional to the amount of displacement of the slider.

Piezoelectric devices can be attached directly to a surface to measure tension and stress. Unlike an LDP, a piezoelectric device produces a voltage directly, hence doesn't need a DC voltage source. A microphone can also be used to measure sound-related phenomenon.

In fact, any device that can provide a voltage that varies between zero and a few hundred millivolts can be connected to the circuit described in this article. R-E

EXPERIMENTING WITH ADC FOR YOUR PC



JAMES J. BARBARELLO

Add a variety of analog inputs to the PC-based analog-to-digital convertor.

PC-BASED TEST INSTRUMENTATION has turned out to be a popular topic among readers. We continue this ongoing series by interfacing new sensing devices (a linear displacement transducer, a piezoelectric transducer, and a microphone) to the analog-to-digital converter (ADC) discussed in the January 1992 issue. If you're just getting started with this series, here's a brief review of progress thus far.

Quick review

The heart of the system is a PC I/O expander called the PC IO that adds 24 lines of digital I/O by means of an 8255 Parallel Input/Output (PIO), all on a single 8-bit expansion card. In the January 1992 installment, we added an outboard analog-to-digital converter (ADC), based on an ADC0804. The ADC allows you to measure and capture any 0–5 volt analog signal (e.g., temperature, displacement, and audio). The circuit includes a variable-gain amplifier for low-level inputs, and a DC-offset adjustment for inputs with unwanted DC components. A temperature probe shown in that article was built around the base-collector junction of a silicon transistor. With the proper settings for amplification and DC offset, a several-hundred millivolt change over a 100°reeC temperature range produced a digital output spread over the full input range of the ADC0804.

In general, an A/D converter and a PC can provide very low-cost solutions to measurement problems that would ordinarily require costly test equipment. Following are some ideas, practical circuits, and software for interfacing other types of devices. These ideas cover the three most common types of sensing: mechanical, electrical, and acoustic.

Mechanical sensing

As an example of mechanical sensing, the author and a friend were contracted to measure the displacement, impact force, and velocity of electromechanical rappers used in the pollution-control industry. Factory smokestacks have environmental controls that contain huge, electrostatically charged metal plates. As the smoke goes up the smokestack, it passes by those plates; the plates "pull" the suspended particles to them, removing pollution that would otherwise be released to the outside air. Particles accumulate on the plates, and eventually have to be removed. To "clean" the plates, rappers (like giant solenoids) provide quick, hard jolts to the plates, breaking trapped particles free in huge sheets that fall to the bottom of the smokestack. Our task was to determine the characteristics of existing commercial rappers, and then determine whether our customer's rapper could be made better than the competition's.

Our test instrumentation consisted of an 8-bit A/D converter hooked up to a computer, and fed by a rectilinear potenti-

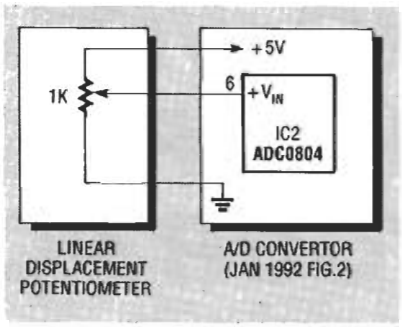


FIG. 1—INTERFACE A LINEAR displacement transducer to the PC IO as shown here. The wiper of the LDT drives the ADC0804 directly, so remember to disconnect the op-amp output (pin 7) from the ADC's input. (The original circuit was shown in Fig. 2 in the January 1992 installment.)

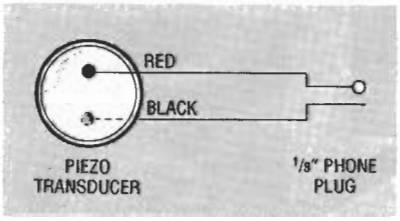


FIG. 2—INTERFACE A PIEZOELECTRIC transducer as shown here. Make sure the red lead goes to the tip and the black lead to the ring of the plug. Also, to avoid feeding DC voltage into the transducer, disconnect the J1 end of R1.

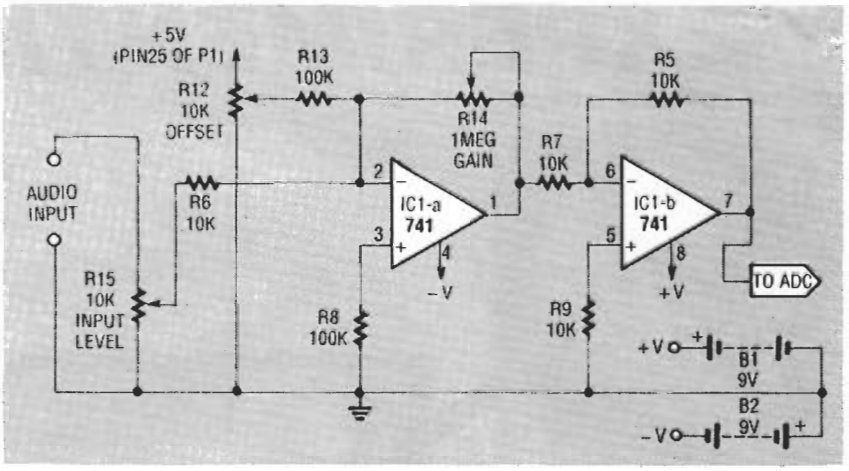


FIG. 3—INTERFACE AN AUDIO SOURCE as shown here. The nonsequential part numbers occur to keep numbering consistent with the original circuit.

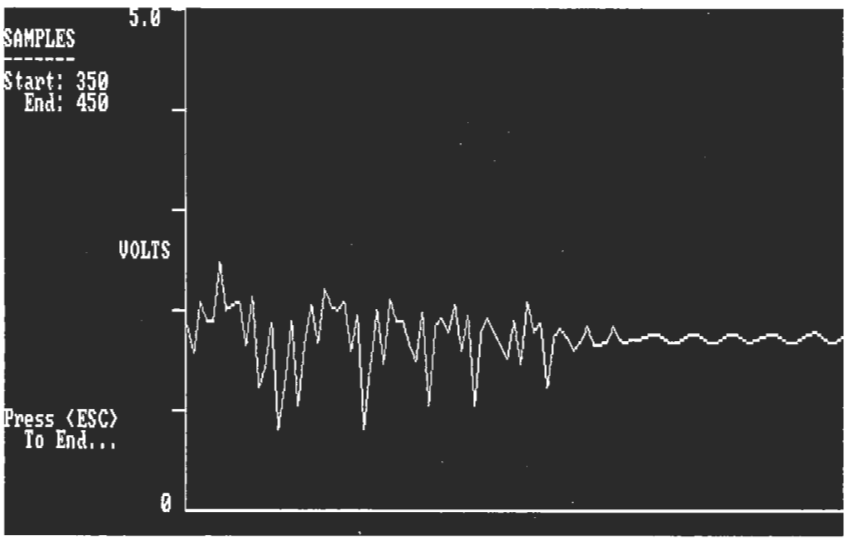


FIG. 4—GRAPHICAL OUTPUT from GRAFDATA.BAS (Listing 2), which allows you to display as many as 1000 data points captured by ADC.BAS (Listing 1).

ometer (also called a linear displacement transducer, or LDT). LDT's are precision potentiometers that move linearly, rather than circularly. Even at surplus prices, LDT's aren't cheap ($20–$50), but they do a great job. Figure 2 in the January 1992 issue shows the complete ADC circuit; we simply supplied it with the input shown here in Fig. 1. Note that we power the LDT directly with a 5-volt DC supply, so neither amplification nor DC offset were required. Instead, we drive the ADC0804 (IC2) directly, as shown in the figure.

If you use this type of arrangement, be sure to disconnect the op-amp's output (pin 7) from the ADC's input (pin 6). In addition, make sure the potentiometer has a resistance of at least 1K, so as not to draw too much current from your PC's power supply. Also, the mechanical linkage between the LDT and the rapper must be firm so there is little or no slippage at the speeds encountered. Otherwise, the potentiometer will not give a true indication of displacement. This type of setup would also be useful for measuring the movement of solenoids, pistons, cams, and similar devices.

Using that circuit, we obtained a series of outputs that showed exactly how the rapper's plunger moved a quarter of a second after it triggered. Eight bits of digital data made it easy to calculate the rate of change of displacement (velocity), as well as the maximum displacement. We determined impact force using strain gages.

The only problem we had was in determining just when to read the LDT's resistance. The solution turned out to be straightforward, as we'll see momentarily.

Electrical sensing

The temperature probe developed last time is one example of electrical sensing. Another good sensing device is piezoelectric material, available in both film and crystal form. An inexpensive source of piezoelectric crystal is Radio Shack's Piezo Transducer (273-073). Simply break open the plastic case, and you have a circular piezoelectric transducer with red and black leads. When flexed in any way, the transducer produces a voltage that varies proportionally to the amount of flexing.

A piezo transducer can be used to measure force of impact, or the amount of bending and flexing. Figure 2 shows how to connect a piezo transducer to

LISTING 1

```
1 REM***********************************************
2 REM** ADC.BAS - V910803 - JJ Barbarello *
3 REM** Capture/Store Analog Data with PcA/D-MATE *
4 REM***********************************************
10 CLS : DEFINT A, X: DIM A(1000): GOSUB 370: LOCATE 1, 23
20 PRINT "PcA/D DATA COLLECTION SYSTEM": LOCATE 2, 1: PRINT
STRING$(79, 220)
30 LOCATE 5, 1: PRINT SPACE$(79); : LOCATE 5, 10
40 LINE INPUT "Enter Time (0-3600 seconds) between samples...";
interval$
50 interval = VAL(interval$)
60 IF interval < 0 OR interval > 3600 THEN BEEP: GOTO 30
70 LOCATE 7, 10: PRINT "Press <Enter> to begin, or <Esc> to
abort...";
80 GOSUB 350: IF A$ = CHR$(27) THEN CLOSE : END
90 LOCATE 5, 10: PRINT SPACE$(60): LOCATE 7, 10: PRINT SPACE$(60)
100 LOCATE 10, 22: PRINT "DIGITAL DATA (0-255):": LOCATE 12, 19
110 IF interval > 0 THEN PRINT "SECONDS BETWEEN SAMPLES: "; interval
120 LOCATE 13, 29: PRINT "SAMPLE NUMBER: "
130 LOCATE 16, 20: PRINT "Press <Esc> to End Sampling...";
140 acount = 1: xsample = 1
150 told = VAL(MID$(TIME$, 4, 2)) * 60 + VAL(RIGHT$(TIME$, 2))
160 IF interval = 0 THEN GOTO 210
170 T = VAL(MID$(TIME$, 4, 2)) * 60 + VAL(RIGHT$(TIME$, 2))
180 IF T = 0 AND (told - T > interval) THEN told = told - 3600
190 IF INKEY$ = CHR$(27) THEN GOTO 260
200 IF T - told > 0 THEN told = told + interval ELSE GOTO 170
210 OUT ADD + 2, 0: OUT ADD + 2, 1
220 x = INP(ADD): LOCATE 10, 44: PRINT USING "###"; x
230 LOCATE 13, 44: PRINT xsample; : xsample = xsample + 1
240 IF acount < 1001 THEN A(acount) = x: acount = acount + 1
250 IF INKEY$ <> CHR$(27) THEN GOTO 160
260 BEEP: LOCATE 16, 15
270 PRINT "Press <Enter> to save data, <Esc> for no save...";
280 GOSUB 350: IF ASC(A$) = 27 THEN CLS : END
290 LOCATE 16, 15
300 LINE INPUT "Enter File Name To Save Data (ex: TEST01.ADD)..."; f$
310 OPEN "r", 1, f$, 2: FIELD 1, 2 AS d1$
320 LOCATE 18, 20: PRINT "Saving Data...";
330 FOR i = 1 TO acount: LSET d1$ = MKI$(A(i)): PUT 1, LOF(1) / 2 +
1: NEXT
340 PRINT "Done.  Press ANY key."; : A$ = INPUT$(1): CLOSE : CLS :
END
350 A$ = INKEY$: IF A$ = "" THEN GOTO 350
360 IF ASC(A$) = 13 OR ASC(A$) = 27 THEN RETURN ELSE BEEP: GOTO 350
370 OPEN "R", 1, "HWADDRES.DAT", 4: FIELD 1, 4 AS A$
380 GET 1, 1: ADD = VAL(A$): CLOSE : DEF SEG = 64: OUT ADD + 3, 146:
RETURN
1 REM** GRAFDATA.BAS
2 REM** Graph PcA/D-Mate .ADD File Data (Requires Graphics Monitor)
3 REM** V910802 - JJ Barbarello
4 REM**
10 CLEAR : DEFINT I-J, Y: KEY OFF: CLS
20 COLOR 0, 7: LOCATE 3, 27: PRINT " GRAPH PcA/D-MATE DATA "; : COLOR
7, 0
30 LOCATE 10, 20: INPUT "Enter File Name (ex: TEST1.ADD)..."; DFN$
40 OPEN "R", 1, DFN$, 2: FIELD 1, 2 AS D$: IF LOF(1) > 0 THEN GOTO 80
50 CLOSE : KILL DFN$: BEEP: LOCATE 25, 13
60 PRINT "That File Doesn't exist.  Press ANY key to try again...";
70 A$ = INPUT$(1): CLS : GOTO 20
80 LOCATE 21, 34: COLOR 16, 7: PRINT " Reading data "; : COLOR 7, 0
90 DIM I((LOF(1) / 2) + 3)
100 FOR I = 1 TO LOF(1) / 2: GET 1, I: I(I) = 190 - CVI(D$): NEXT
110 LOCATE 21, 34: PRINT SPACE$(20)
120 LOCATE 12, 20: PRINT "File Has"; LOF(1) / 2; "Samples."
130 LOCATE 14, 20: INPUT "Enter Start Sample To View..."; ISTART
140 LOCATE 15, 20: INPUT "Enter End Sample To View"; IFIN
150 IF IFIN - ISTART > 1000 THEN IFIN = ISTART + 999
160 I(I) = I(I - 1): I(I + 1) = I(I): I(I + 2) = I(I + 1)
170 YINC = 500 / (IFIN - ISTART)
180 IF (IFIN - ISTART) > 499 THEN istp = 2 ELSE istp = 1
190 SCREEN 2: LOCATE 2, 1: PRINT "SAMPLES": LOCATE 3, 1: PRINT "-----
--"
200 LOCATE 4, 1: PRINT "Start:"; ISTART: LOCATE 5, 3: PRINT "End:";
IFIN
210 LOCATE 20, 1: PRINT "Press <ESC>": LOCATE 21, 3: PRINT "To
End..."
220 LINE (138, 0)-(138, 190): LINE (138, 190)-(639, 190): J = 139
230 FOR I = 0 TO 152 STEP 38: LINE (128, I)-(137, I): NEXT
240 LOCATE 1, 13: PRINT "5.0": LOCATE 24, 16: PRINT "0";
250 LOCATE 12, 12: PRINT "VOLTS";
260 FOR I = ISTART TO IFIN STEP istp
270 LINE (J, I(I))-(J + YINC, I(I + istp)): J = J + YINC: NEXT I
280 A$ = INKEY$: IF A$ = "" THEN 280
290 IF ASC(A$) = 27 THEN SCREEN 0: END ELSE BEEP: GOTO 280
```

our circuit. We want only the voltage generated by the piezo sensor, so disconnect the J1 end of R1, shown in Fig. 2 last time (R1 provided +5-volts DC to the temperature probe). With no input signal, adjust R11 for maximum gain, and R12 so output is about 2.5-volts DC. (The latter setting allows the piezo voltage to vary from that quiescent level.) You might also want to adjust the value of R3 so that the piezo sensor produces maximum indication at its maximum output.

To test the circuit, use a piece of masking tape to secure the piezo sensor to a flat surface, making sure you aren't flexing it. Tap the surface firmly and note the response. In actual use, you would mount the sensor using a clamping device or flexible adhesive. This type of device is useful for measuring applied impact, oscillation after impact, or amount of flex or bend.

Acoustic sensing

You've probably seen audio signals displayed on an oscilloscope. Now imagine "freezing" those signals—i.e., storing them in digital form. Scientists capture digitized audio in this way to study the sounds that whales, birds, and other animals make. In digitized form, it's easy to compare and contrast different sounds to determine which have similar characteristics, or to look at other factors that wouldn't otherwise be obvious.

To accommodate a microphone input, we modified the ADC circuit as shown in Fig. 3. The changes allow us to increase the gain of IC1-a to a maximum of 100, and to attenuate high-level inputs. Gain is determined by the ratio R14/R6 = $10^6/10^4 = 10^2 = 100$. If the part numbering in Fig. 3 seems strange, it's because we tried to stay consistent with the main circuit shown in Fig. 2 in January. Note that in the January version, R6 was 100K; here we decreased its value to 10K. Also, we no longer use R1–R4.

In operation, first set R12 to mid-value. Then adjust R14 for maximum range, as described last time. Last, readjust R12 as necessary to set the quiescent level to about 2.5-volts DC. For example, a microphone might provide a maximum output

LISTING 2

```
1 REM** GRAFDATA.BAS
2 REM** Graph PcA/D-Mate .ADD File Data (Requires Graphics Monitor)
3 REM** V910802 - JJ Barbarello
4 REM**
10 CLEAR : DEFINT I-J, Y: KEY OFF: CLS
20 COLOR 0, 7: LOCATE 3, 27: PRINT " GRAPH PcA/D-MATE DATA "; : COLOR
7, 0
30 LOCATE 10, 20: INPUT "Enter File Name (ex: TEST1.ADD)..."; DFN$
40 OPEN "R", 1, DFN$, 2: FIELD 1, 2 AS D$: IF LOF(1) > 0 THEN GOTO 80
50 CLOSE : KILL DFN$: BEEP: LOCATE 25, 13
60 PRINT "That File Doesn't exist.  Press ANY key to try again...";
70 A$ = INPUT$(1): CLS : GOTO 20
80 LOCATE 21, 34: COLOR 16, 7: PRINT " Reading data "; : COLOR 7, 0
90 DIM I((LOF(1) / 2) + 3)
100 FOR I = 1 TO LOF(1) / 2: GET 1, I: I(I) = 190 - CVI(D$): NEXT
110 LOCATE 21, 34: PRINT SPACE$(20)
120 LOCATE 12, 20: PRINT "File Has"; LOF(1) / 2; "Samples."
130 LOCATE 14, 20: INPUT "Enter Start Sample To View..."; ISTART
140 LOCATE 15, 20: INPUT "Enter End Sample To View"; IFIN
150 IF IFIN - ISTART > 1000 THEN IFIN = ISTART + 999
160 I(I) = I(I - 1): I(I + 1) = I(I): I(I + 2) = I(I + 1)
170 YINC = 500 / (IFIN - ISTART)
180 IF (IFIN - ISTART) > 499 THEN ISTP = 2 ELSE ISTP = 1
190 SCREEN 2: LOCATE 2, 1: PRINT "SAMPLES": LOCATE 3, 1: PRINT "-----
--"
200 LOCATE 4, 1: PRINT "Start:"; ISTART: LOCATE 5, 3: PRINT "End:";
IFIN
210 LOCATE 20, 1: PRINT "Press <ESC>": LOCATE 21, 3: PRINT "To
End..."
220 LINE (138, 0)-(138, 190): LINE (138, 190)-(639, 190): J = 139
230 FOR I = 0 TO 152 STEP 38: LINE (128, I)-(137, I): NEXT
240 LOCATE 1, 13: PRINT "5.0": LOCATE 24, 16: PRINT "0";
250 LOCATE 12, 12: PRINT "VOLTS";
260 FOR I = ISTART TO IFIN STEP ISTP
270 LINE (J, I(I))-(J + YINC, I(I + ISTP)): J = J + YINC: NEXT I
280 A$ = INKEY$: IF A$ = "" THEN 280
290 IF ASC(A$) = 27 THEN SCREEN 0: END ELSE BEEP: GOTO 280
```

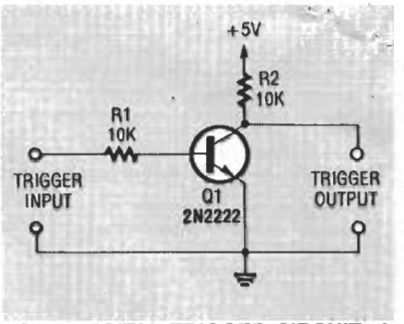


FIG. 5—DIGITAL TRIGGER CIRCUIT allows the PC IO to start reading data based on the appearance of a predetermined signal. Connect this circuit between that signal and an unused PC IO input, and modify the software accordingly.

ORDERING INFORMATION

The following items are available from JJ Barbarello, 817 Tennent Road, Manalapan, NJ 07726.

- **Complete PC IO Board with PC board and all components (part# PCIO)—$39.95**
- **An ADC0804 and a calibrated PN222A temperature sensor transistor (part #ADC)—$8.00**
- **PC IO software, including compiled and source code versions with continuous and interval sampling and data logging/listing. Specify disk size, 3.5" or 5.25". (part #ADC-S)—$8.00.**
- **Software from this installment, including a menu-driven, compiled program (with source code), expanding on the identified functions and adding discrete data listing, graphed data sample selection/value display, and other features (part #ADM-S)—$10**

Specify part number(s) and send check or Money Order.

The author will be glad to answer any questions, but they must be accompanied by a self-addressed stamped return envelope.

voltage of 0.5. Adjust R14 so the maximum is about 4.0 volts. Then readjust R12 so the no-signal (quiescent) value is about 2.5-volts DC. Now you'll get the best resolution from the circuit.

Software

We wrote two programs for data capture and display. Listing 1 shows the capture program (ADC.BAS), and Listing 2 the display program (GRAFDATA.BAS)—both are available on the RE-BBS (516-293-3000, 1200/2400, 8N1) as a file called PCADMAT2.LST. Figure 4 shows sample output from GRAFDATA. Enhanced versions of both programs are available on disk from the author for a nominal fee; see the sidebar for details.

ADC.BAS is a general-purpose data capture and storage utility. To use it, you must create a data file called HWADDRES.DAT in the same directory as the program. The file contains the decimal address of the PC IO card. You can create the file with any text editor or word processor capable of storing text in ASCII format. Assuming your card is set to the default address (640), create a one-line file that contains "640D" (without the quotes) followed by a carriage return.

Connect the ADC circuit to your PC and run ADC.BAS. It initializes the PC IO card, then asks you for the interval (in seconds) to wait between samples. If you want continuous samples (with no delay), enter 0. Otherwise, you can enter any number between 1 and 3600 (one hour). When you press Enter, sampling begins.

The program displays three pieces of information: current data (line 100), time between samples (line 110), and current sample number (line 120). Press Esc at any time to end sampling (line 130).

Lines 150–200 take care of interval timing. The sampling program showed last time used a timing loop that was system-dependent; the current version reads the system clock, hence is system-independent. Sampling takes place in lines 210–250; as many as 1000 samples are saved in array A.

When you press Esc to end sampling, execution continues at line 270. If you press Esc again, data is abandoned and the program ends. If you press Enter, the program requests a file name for storing the data. Specify the complete file name, including any path and extension (e.g., C:\SUB1\TEST01.ADD.) If you don't specify a complete path, the file will be stored in the current directory. If the file already exists, current data will be appended to the end of the file.

GRAFDATA.BAS allows you to plot your data on-screen. When you run the program, it asks you to specify a data file. As with ADC.BAS, specify the full file path if the file is not in the cur-

(Continued on page 76)

continued from page 68

rent directory. If the file exists, the program then tells you how many samples it contains, and asks start and stop points for the plot; the maximum number is 1000.

The program (line 170) scales the graph to produce the widest possible graph; lines 220–270 graph the data. Press Esc to terminate the program. To "magnify" a limited portion of the data set, run the program again, and enter appropriate start and end values (e.g., 350 and 450). You can print your graphs by running the DOS program GRAPHICS.COM before GRAFDATA.BAS. When the desired graph appears on the screen, press PrintScrn. (You must have a compatible printer in order to print out any of the graphs.)

Triggered sampling

Now with the software under our belt, it's time to finish the "rapper" story begun earlier. The rapper was triggered by an electronic control circuit. We tried to trigger sampling by hand, but it was impossible to tell when the rapper was about to energize, and reaction time was too slow and unsure for accurate triggering. Our solution was to create a trigger input. We tapped a digital signal from the control circuit that changed just before the rapper triggered.

The ADC circuit uses only 8 of the 24 input lines on the PC IO. What we did is use bit 0 of port B as a trigger input. Then by adding a single line of code, we could trip on either a low-to-high signal:

```
145 IF (INP(ADD + 1) AND 1) = 0 THEN
    GOTO 145
```

or a high-to-low signal:

```
145 IF (INP(ADD + 1) AND 1) = 1 THEN
    GOTO 145
```

If you want to add a slight delay between the trigger and the start of sampling:

```
146 FOR I = 1 TO DELAYTIME : NEXT
```

However, the delay time in this case would vary with the speed of your PC. In addition, make sure the trigger signal is TTL-compatible (logic low = 0V, and logic high = 5V). If it isn't, or you want to buffer the trigger signal from your system, you can use the circuit of Figure 5. Remember, this circuit inverts the incoming trigger signal, so a positive-going trigger will be negative-going when it exits the buffer circuit. Adjust added line 145 accordingly.

Conclusions

The analog world of nature is not so distant from the digital world of computers. Simple circuitry can allow your PC to function as a digital eye, ear, or hand on the external world. The circuits and software shown here are only the beginning. **R-I**