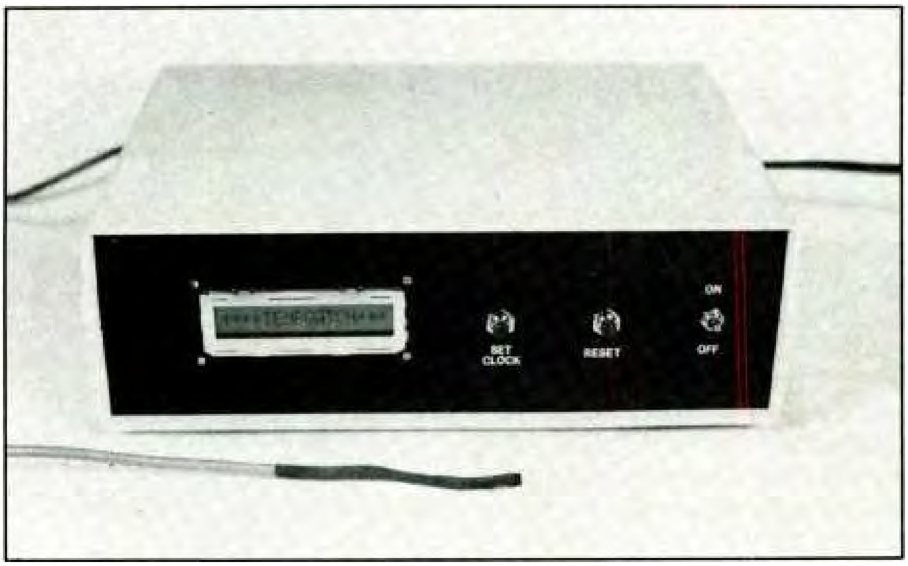
Microprocessor Control with BASIC

(Conclusion)

Adding a "Tempwatch" accessory to last month's Microsys development unit creates a "smart" thermometer

By Jan Axelson & Jim Hughes



Last month in Part 1 of this article, we showed you how to build and use the Microsys development system for microprocessor-based projects. The basic system described so far consists of an 8052AH-BASIC microcontroller, RAM, EPROM, and a serial port for connection to a terminal (or a computer being run as a terminal). Now we'll add inputs and outputs that will turn the Microsys into a complete, stand-alone temperature-measuring instrument. We call this our Tempwatch "smart" thermometer.

Tempwatch isn't just an ordinary thermometer. It displays the present temperature and time and also the maximum and minimum temperatures measured and the times these occurred. To add these capabilities to the basic Microsys, you must add a few more modules. For input, you'll add a temperature sensor, an ADC (analog-to-digital converter), and two pushbutton switches for user control. A 16-character dot-matrix LCD (liquid-crystal display) module serves as the "output" device.

About the Circuit

In Fig. 5 is shown the schematic diagram for the ADC and its associated circuitry. Input to the ADC is provided by LM34 temperature sensor *IC13*. The output of this three-terminal device is an analog voltage that is proportional to temperature at a rate of 10 millivolts per degree Fahrenheit. At 70 degrees, its output is within 0.8 millivolt of 700 millivolts. Resistor *R18* and capacitor *C21* form a low-pass filter to reduce noise on the input.

The output from *IC13* is the analog input that is applied to input pin 6 of ADC0804 analog-to-digital converter *IC12*. The ADC converts this analog input into an eight-bit word whose value represents the analog voltage.

LM385-1.2 precision reference diode *IC14* provides a stable 1.2-volt reference at pin 9 of *IC12*. The potential at pin 9 is half the voltage that causes a full-scale output on the ADC. That is, a 2.4-volt input (which is twice the reference voltage) causes an output of 11111111 at AD0 through AD7 of *IC12*. This gives the ADC a resolution of a little less than 10 millivolts, which translates into 1 degree Fahrenheit per bit.

Measurement range of the Tempwatch depends partly on the version of chip used for *IC13*. The LM34C, for example, is accurate over a range of -40 to $+230$ degrees Fahrenheit

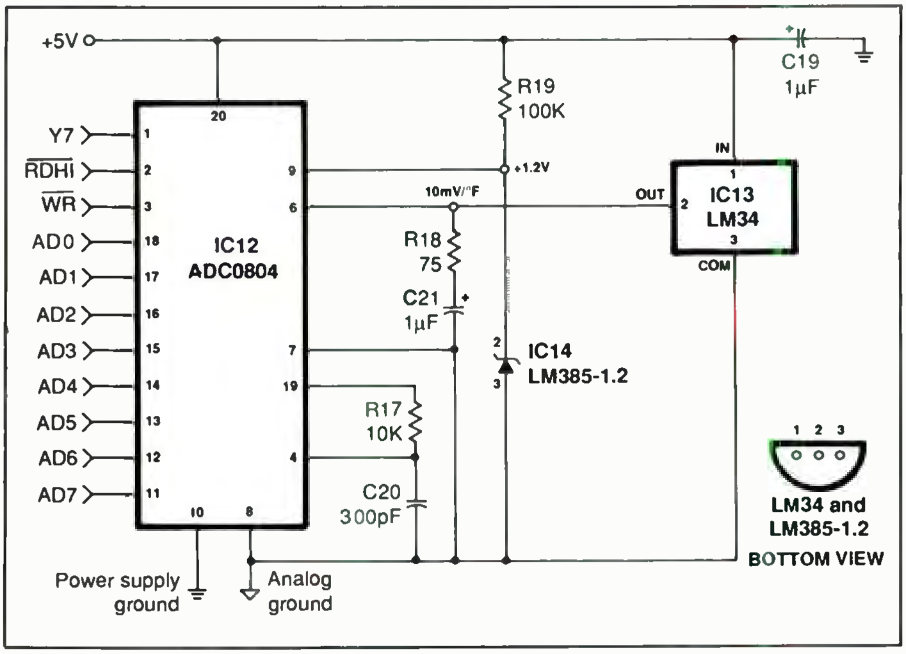


Fig. 5. Schematic diagram for the temperature-sensor and A/D converter circuitry that connect to the Microsys.

(though the Tempwatch measures only down to zero degrees). The LM34D, on the other hand, has a narrower range of +32 to +212 degrees Fahrenheit, as well as a lower price to match; so it may be a better choice if this range suits your needs.

A separate ground return for analog signals in the sensing circuit helps to reduce noise, which improves measuring accuracy. Ground connections for *IC13*, *IC14*, and *C21* connect to *IC12*'s analog ground at pin 8. A single ground wire then connects pin 8 directly to ground on +5-volt regulator *IC11*.

The eight data lines on *IC12* (pins 11 through 18) connect directly to AD0 through AD7 on *IC2*. CHIP SELECT at pin 1 of *IC12* is generated by Y7 at pin 7 of *IC3*. Output Y7 is low when A13, A14, and A15 are all high. This places *IC12* at hexadecimal address E000. Reading and writing to *IC12* are controlled by RDHI at pin 3 of *IC7A* and WRITE at pin 16 of *IC2*.

Resistor *R17* and capacitor *C20* provide the ADC's clock at about 300,000 Hertz (300 kHz).

A low-to-high transition on pin 1 or pin 3 of *IC12* causes the ADC to begin converting its analog input (the temperature voltage) to an eight-bit word. The 8052 can be programmed to read this word on AD0 through AD7 and convert it to a temperature. Next we need a way to display the measurement.

Figure 6 shows the wiring for *M1*, an Amperex-Philips No. LTN111R-10 LCD module. This module has a 16-character display and contains a CMOS controller/driver and its own ROM and RAM. Each of the 16 characters in the display is created by turning on and off the LCD segments arranged in a 5 × 7 matrix. (A matrix display was chosen for this project because it allows much more flexibility in character—which can be letters, numbers or other symbols—design than seven-segment displays do.)

The module's ROM stores the patterns that create 160 different characters, and the RAM stores display data. As Fig. 7 shows, virtually all of the circuitry of *M1* is contained in one LSI surface-mount device. Figure 8 shows a sample display on *M1*.

Data and instructions are written

PARTS LIST

Semiconductors

IC12—ADC0804 analog-to-digital converter
IC13—LM34 precision Fahrenheit temperature sensor (Digi-Key)
IC14—LM385-1.2 voltage-reference diode (Digi-Key)
IC15—74LS00 quad NAND gate

Capacitors (25-WV)

C19,C21—1.0-µF tantalum electrolytic
C20—300-pF ceramic

Resistors (¼-watt, 5% tolerance)

R17—10,000 ohms
R18—75 ohms
R19—100,000 ohms
R20—5,000-ohm pc-mount potentiometer

Miscellaneous

M1—LTN111R-10 16-character LCD module (Digi-Key Cat. No. AMX116-ND)
S2,S3—Spst normally-open, momentary action pushbutton switch
Three-conductor shielded cable (60 inches long); 34-conductor ribbon cable with connectors at both ends (8 inch long—see text); 14-pin single-row header; 34-pin double-row header; Wire Wrap IC sockets and hardware; rubber grommet; small-diameter heat-shrinkable tubing; machine hardware; hookup wire; solder; etc.

Note: For a 2764 EPROM containing the 10 programs listed in this article, plus temperature-alarm program, send $19 to Lakeview Microdesigns, 2209 Winnebago St., Madison, WI 53704. Include $2 P&H per order. Wisconsin residents, please add 5% state sales tax.

Supplier Addresses

Digi-Key Corp.
701 Brooks Ave. S.
Thief River Falls, MN 56701-0677
1-800-344-4539

Jameco Electronics
1355 Shoreway Rd.
Belmont, CA 94002
415-592-8097

Sources of Information

Applications information for LTN111R-10 LCD module (and others)—Digi-Key (Cat. No. AMXDS)
National Semiconductor 1988 Linear Data Book #2; contains data sheets and applications notes for ADC0804, LM34, LM385-1.2—Digi-Key (Cat. No. 9052B) or Jameco (Cat. No. 400042)

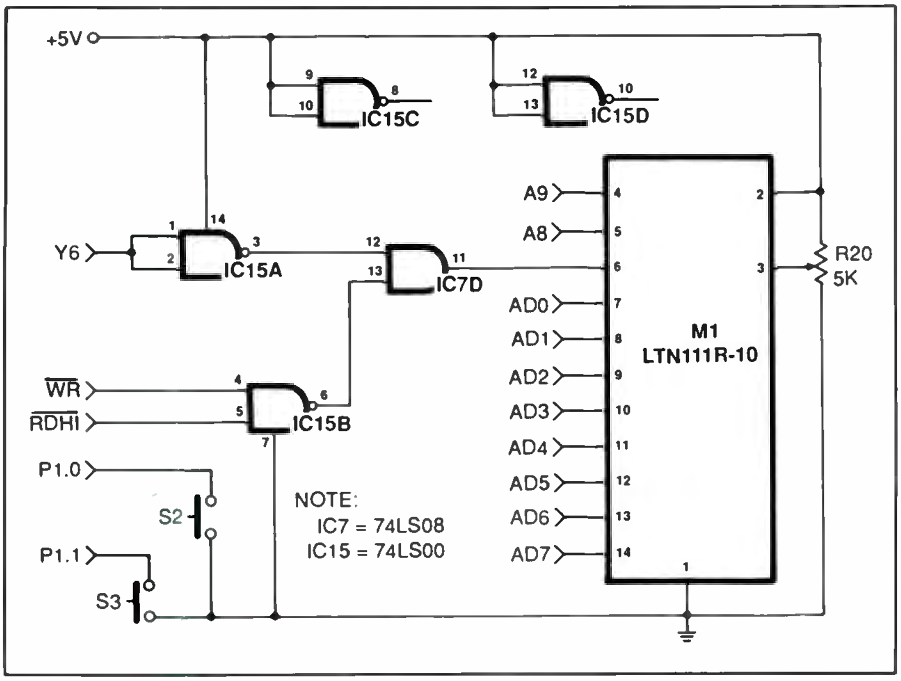


Fig. 6. Schematic diagram of LCD module and switches that connect to the Microsys.

to *M1* on AD0 through AD7 on pins 7 through 14. Three control lines at pins 4, 5 and 6 complete the interface with *IC2*. Pin 3 is a contrast-adjust input for the display, and connections to +5 volts and ground are at pins 2 and 1, respectively.

To display a character on *M1*, *IC2* writes the address of the position desired to *M1*.(Alternatively, *M1* increments or decrements the address automatically after a character is written to the display.) *IC2* next writes the code for the character desired to *M1*. The module then displays the character desired at the position requested.

Logic gates *IC15A*, *IC15B* and *IC7D* generate an enable signal at pin 6 of *M1*. When Y6 at pin 9 of *IC3* is low and either RDHI or WRITE is low, *M1* can be read or written to. Line Y6 is low when A14 and A15 are high and A13 is low, which places *M1* at hexadecimal address C000.

Address lines A8 and A9 select the operation to be performed when *M1* is enabled. When A8 and A9 are both low, instructions can be received by *M1* on AD0 through AD7. When A8 is low and A9 is high, character codes can be received.

The applications notes for *M1* describe the various instruction codes and their functions. They also illustrate the possible characters and their character codes.

Momentary-action pushbutton switches *S2* and *S3* permit user input to the system. Each switch connects to a line on Port 1, an I/O (input/output) Port of *IC2*. In the Tempwatch program, *S2* resets the maximum and minimum temperatures and times to the present temperature and time, and *S3* lets the user set the Tempwatch's clock.

Construction

Creating the Tempwatch from the Microsys you built last month requires you to add the circuitry for the ADC, sensor, LCD module and switches. The control program must also be written and saved in the EPROM. Figure 9 shows a completed Tempwatch mounted in its enclosure. Adding the hardware is the first step. Here is where the locking connectors we specified for the Microsys come in handy. Unplug both connectors and remove the circuit-board assembly from the Microsys' enclosure.

Wire Wrap the new components specified here on this circuit-board assembly. Begin by mounting the sockets for *IC12* and *IC15*. Do *not* install the ICs themselves in the sockets until after you have ascertained that your wiring is correct and power-distribution tests have been conducted. As you did last month, solder the pins of the sockets to the copper rings through which they pass on the board. Then wire the power-supply and ground pins to the +5-volt and ground buses on the circuit board.

Using Fig. 5 as a guide, wire the circuitry associated with *IC12*, adding *IC14*, *R17*, *R18*, *R19*, *C19*, *C20* and *C21* to the board as required. Mount *IC14*, *R18*, *R19* and *C21* as close to *IC12* as possible, to minimize noise in *IC12*'s analog circuitry. Wire the analog grounds (ground connections to *C21*, *IC13* and *IC14*) directly to pin 8 of *IC12*. Connect and solder one wire from pin 8 of *IC12* to a ground point near *IC11*.

To permit flexible use of the project, it is a good idea to mount temperature sensor *IC13* at the end of a 5-foot-long cable. Use three-conductor shielded cable for this to assure good immunity to noise.

Prepare the sensor cable as follows. First remove 2 inches of the outer plastic jacket from each end. If the shield is made up of wire mesh, separate the wires at both ends back to the cut-off plastic jacket and clip its conductors off ar only *one* end. Twist together the fine wires of the braid at the other end and sparingly tin with solder. If the shield is foil with a wire tracer, unwind it back to the cut-off plastic jacket, trim the foil at both ends and clip the wire tracer at only *one* end. Strip ½ inch of insulation from all conductors at both ends of the cable, tightly twist together the fine wires of each conductor and sparingly tin with solder.

At the end of the cable from which the shield (and tracer wire) was clipped, slide a 1-inch length of small-diameter heat-shrinkable tubing over and form a small hook in the end of each conductor. Next, trim the leads of IC13 to a length of about ¼ inch and also form small hooks at the ends of the remaining lead lengths. Crimp and solder a cable conductor to each lead on IC13. At the other end of the cable, identify conductors, for later use in wiring them. Do this either by recording the insulation colors for each conductor at *IC13*, or by using an ohmmeter to identify the wires and temporarily labeling each with masking tape.

Slide the tubing over the soldered connections until it is flush with the bottom of *IC13*'s case and shrink the tubing into place. Slide a 4-inch length of larger-diameter heat-shrinkable tubing over the entire cable so it overlaps slightly the bottom of the IC's case and shrink it into place.

Slide a 1½-inch length of tubing onto the shield at the other end of the cable and shrink it into place. Slide another larger-diameter piece of tubing onto the cable, placed so that about 1 inch of the cable conductors extends out the end of the tubing, and shrink into place.

Insert the three cable conductors in the circuit board and solder connections to pins 6, 7, and 20 of *IC12*, using Fig. 5 and your previous identification of the conductors as guides. Solder a connection between the shield and pin 8 of *IC12*.

Because module *M1* contains CMOS integrated circuits, be sure to exercise the normal safety precautions for handling devices that are subject to damage from electrostatic discharge. In addition, the display can be damaged by mechanical shock or pressure; so handle it with care.

The *M1* circuit board has a single row of 14 holes on 0.1-inch centers for mounting on it an interface-cable connector. Unfortunately, this configuration doesn't seem to match any readily available connectors; so some improvisation is in order.

For the sake of convenience, a 34-pin ribbon cable is used to connect *M1* to the circuit board, even though many of the cable's conductors will go unused. You can buy or make the cable, which should be about 5 inches long and have socket connectors at each end.

Solder a 14-pin single-row header to *M1*. On the Microsys circuit board, a 34-pin double-row header is recommended, to match the connector and minimize the chances of plugging it in incorrectly. Use an indelible marker to place an identifying dot near pin 1 of each header. On *M1*, pin 1 is nearest the corner of the module's circuit board. On the Microsys circuit board, pin 1 will be the pin that is nearest the pin 1 arrow when the connector is plugged in.

Plug the ribbon cable's connectors into the headers, matching pin 1 to pin 1 on the connector and header at

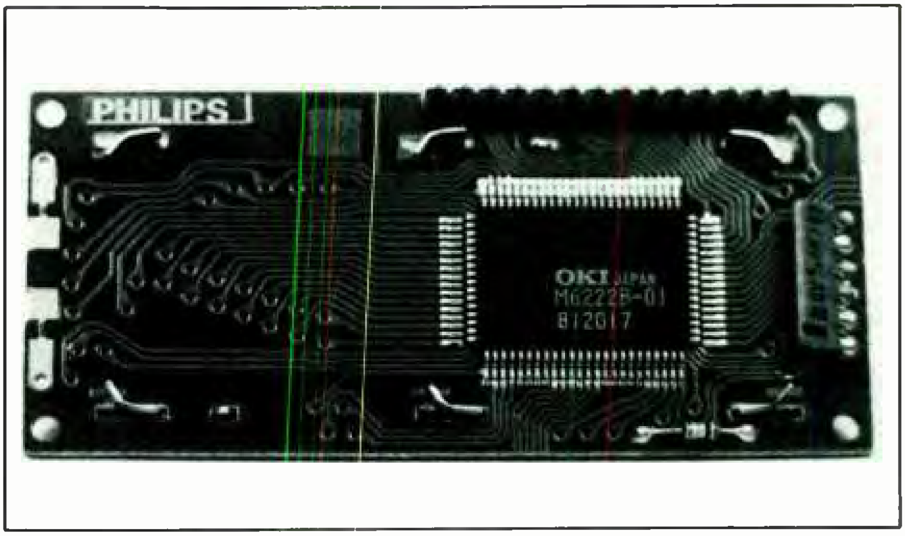


Fig. 7. LCD module has one IC that contains virtually all of the module's controller and driver circuitry.

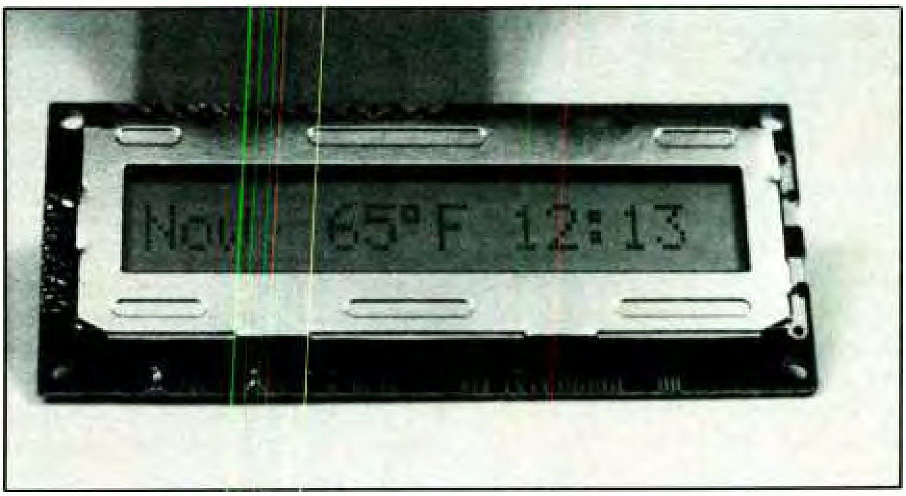


Fig. 8. LCD module used in this project can display message of up to 16 characters, using letters, numbers, or other symbols.

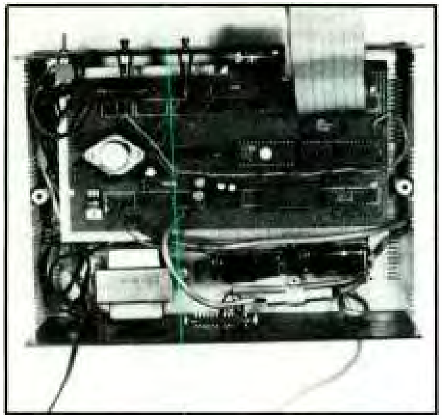


Fig. 9. Circuit board, transformer and batteries mount on the floor of the enclosure; serial-port connector and sensor cable exit through rear panel.

each end. On *M1*, plug the socket nearest the pin 1 arrow into pin 1 on *M1*'s header. One entire row of sockets on the connector, as well as the bottom three sockets of the other row, are not used in this project.

Once you are satisfied that everything physically meshes together, disconnect the cable. Then install *R20* on the circuit board, and wire the circuitry shown in Fig. 6. To wire *S2* and *S3*, prepare three 8-inch and one 4-inch lengths of hookup wire by stripping ½ inch of insulation from the ends of each. Twist together one end of the 4-inch wire and one end of one 8-inch wire and crimp and solder the twisted pair to one lug of *S2*. Crimp and solder the free end of the 4-inch wire to one lug on *S3*. Then crimp and solder the other two wires to the remaining switch lugs.

Connect the wire common to both switches to ground on the circuit board. Connect *S2*'s other wire to pin 1 of *IC2*, and *S3*'s other wire to pin 2 of *IC2*.

Route the sensor cable through a rubber-grommet-lined hole in the enclosure's rear panel.

Switches *S2* and *S3* and module *M1* mount on the front panel, the switches in suitable-size holes and the module in a 2⅞ by 1⁷⁄₁₆-inch slot you must cut into the panel. In addition to the slot, *M1* requires that you drill four mounting holes for it in the front panel. Cut the larger display window opening first, making it large enough that the module fits easily into it to obviate mechanical stress on the display.

When the opening is ready, insert *M1* in the front panel and mark the locations for the four mounting holes. Remove and set aside the module and drill appropriate size holes in the marked locations.

Temporarily remove S1 from the front panel and label *S1*'s positions ON and OFF, *S2* RESET, and *S3* SET CLOCK on the front panel. Use a dry-transfer lettering kit and protect the legends with two light coats of clear spray acrylic. Allow the first coat to dry before spraying on the next.

Mount M1, using spacers of sufficient length to assure that fit between *M1* and the front panel is correct. Install the circuit board in its enclosure and connect the two sets of locking connectors. Tie a knot in the sensor cable to act as a strain relief, and route this cable through the rubber grommet in the back panel. Do *not* connect the ribbon cable until preliminary power supply checks have been made and you are certain that the circuit has been wired correctly. Mount *S1*, *S2* and *S3* in their respective holes on the front panel.

Checkout

Power up the Microsys and use a dc voltmeter or multimeter set to the dc volts function for the presence of +5 volts, ±0.2 volt, between output of *IC11* and circuit ground. Connect the meter's common lead to any convenient circuit-ground point in the circuit and use the "hot" lead to probe the circuit for this measurement. Also measure for +5 volts from pin 2 to

Program 6. Write Temperature to Screen

```
1      REM Program 6, write temperature to screen
10     DO
20    XBY(0E000H)=255 :  REM write to adc to start conversion
30    ADC=XBY(0E000H) :  REM read adc output
40    TEMP=ADC*2.47*100/255 :  REM calculate temperature
50     PRINT "The temperature is ",TEMP," degrees Fahrenheit"
60     WHILE 1=1
70     END
```

Program 7. Write Temperature to LCD Module

```
1      REM Program 7, write to LCD module
10    XBY(0C000H)=1 :  REM clear display
20    XBY(0C000H)=38H :  REM select 8-bit data, 2 logical lines
30    XBY(0C000H)=0CH :  REM turn display on, cursor off
40    XBY(0C000H)=6 :  REM increment address counter after each write
50    XBY(0C000H)=80H :  REM select position 1 on display
60     FOR I=1 TO 4 :  REM write "*****" to display
70    XBY(0C200H)=2AH
80     NEXT I
90    XBY(0C200H)=ASC(T) :  REM write "TEMP" to display
100   XBY(0C200H)=ASC(E)
120   XBY(0C200H)=ASC(M)
130   XBY(0C200H)=ASC(P)
140   XBY(0C000H)=0C0H :  REM select position 9 on display
150   XBY(0C200H)=ASC(W) :  REM write "WATCH" to display
160   XBY(0C200H)=ASC(A)
170   XBY(0C200H)=ASC(T)
180   XBY(0C200H)=ASC(C)
190   XBY(0C200H)=ASC(H)
200    FOR I=1 TO 3 :  REM write "****" to display
210   XBY(0C200H)=2AH
220    NEXT I
230    END
```

pin 1 on the double-row header, and from pin 20 to pin 10 of *IC12*.

If at any point the meter doesn't register the proper voltage, power down the circuit and locate the source of and rectify the problem before continuing.

When everything looks okay, power down the circuit and plug *IC12* and *IC15* into their respective sockets, making sure you properly orient each IC and that no pins overhang the sockets or fold under between the ICs and the sockets. Then connect *M1*'s ribbon cable, being sure to observe proper orientation for pin 1. If possible, insert a new or freshly erased EPROM into *IC9*'s socket. This will allow you to take advantage of the 8052's special PROG2 command, which will be described later.

Boot the Microsys to MCS BASIC-52 as before: configure your terminal for communication with eight bits, one stop bit, and no parity. Connect the Microsys to the serial port of your terminal, power up the Microsys and press the SPACE bar at the terminal.

If you don't see the READY prompt, power down and locate and fix the problem before continuing. Re-examine the added wiring on the Microsys. Be sure your terminal is configured properly, as described last month.

When you do obtain the READY prompt, you can begin measuring temperature. The listing given in Program 6 "reads" the output of *IC12*, converts it to a temperature and displays the result on the terminal's video display screen. Key in Program 6 at your terminal and follow this by typing RUN and a carriage return (ENTER key) to run it. The terminal screen will now display any measurements, in degrees Fahrenheit, taken by the sensor.

Program 6 uses the special operator XBY in MCS BASIC-52 to read and write to external memory. Line 20 writes the value 255 to *IC12*, which is located at hex address E000. Writing to *IC12* causes this IC to

Program 8. Test Operation of Switches

```
1      REM Program 8, switch test
10     PRINT "(CONTROL/C to quit)"
20     DO
30    X=PORT1 :  REM read port 1
40    Y=X-4*INT(X/4) :  REM calculate (bit 1 + bit 2) of port 1
50     IF Y=2 THEN  PRINT "Bit 0 of port 1=0; S2 pressed"
60     IF Y=1 THEN  PRINT "Bit 1 of port 1=0; S3 pressed"
70     WHILE 1=1
80     END
```

begin converting its analog input to a digital word. (The actual value written to the ADC is unimportant; any write command serves the purpose.)

Line 30 reads the byte at AD0 through AD7 of *IC12* and stores it in the variable ADC. Line 40 converts the byte to a Fahrenheit temperature, and line 50 displays the temperature on the terminal screen.

The number 2.47 in line 40 represents twice the reference voltage at pin 9 of *IC12*. Although the voltage across the reference diode is stable, it can vary among individual ICs from 1.223 to 1.247 volts. For greatest accuracy in the temperature-measuring circuit, measure the voltage from pin 9 to pin 8 of *IC14* on your system and enter twice this value in line 40, in place of 2.47.

To exit Program 6's infinite loop, use CONTROL/C.

Program 7 writes the message "****TEMPWATCH***" (minus the quotation marks) in the LCD module's display. Type in and run this program as you did before. After running Program 2, adjust *R20* to give the desired contrast on the display.

Because *M1* is quite powerful in itself, programming it is relatively straightforward. Lines 10 through 40 in Program 7 initialize and configure the module by writing values to hex address C000. Address C000 enables *M1* by bringing line Y6 low. The initialization and configuration codes are described in the applications notes for *M1*.

Line 50 selects position 1 on *M1*, at hex location 80, and lines 60 through 130 cause the message "****TEMP" (again, without the quotation marks) to appear in the first eight character positions of *M1*. The character codes for the letters in the message are the standard ASCII codes for the letters, specified by the ASC operator in BASIC.

Characters are written to *M1* at hex address C200. The "2" in C200 sets A9 high, to program *M1* to receive the character codes. The module increments its address counter after each character is displayed, so the eight-character message fills the first eight positions on the module.

Although the 16 character positions of *M1* are physically arranged in a single line, they are addressed in two logical lines. Positions 1 through 8 are at hex 80 through 87, while positions 9 through 16 are at C0 through C7. The jump in addresses between the two lines has to be taken into account when programming the module.

This is why line 140 sets *M1*'s address counter to C0, to specify position 9. Lines 150 through 220 write "WATCH***," to positions 9 through 16 to finish the message.

Program 8 tests the operation of switches *S2* and *S3* by continuously monitoring Port 1 and giving a message on the terminal screen when a switch is pressed. Type in and run this program. Then press *S2* and *S3* and watch for the appropriate messages.

The switches connect to otherwise unused lines on Port 1, an eight-bit I/O port of *IC2*. Line 30 reads the value of Port 1, and line 40 finds the sum of bit 0 and bit 1. If the sum equals 2, bit 0 is low, which means *S2* has been pressed. If the sum equals 1,

Program 9. Combine Features of Programs 6 & 7

```
1       REM Program 9, display temperature on LCD module
10      PRINT "(CONTROL/C to quit)"
20      XBY(0C000H)=1 :  REM clear display
30      XBY(0C000H)=38H :  REM select 8-bit data, 2 logical lines
40      XBY(0C000H)=0CH :  REM turn display on, cursor off
50      XBY(0C000H)=6 :  REM increment address counter after each write
60      DO
70      XBY(0E000H)=255 :  REM write to adc to start conversion
80      TEMP=XBY(0E000H)*2.47*100/255 :  REM get & calculate temperature
90      IF TEMP-INT(TEMP)>=.5 THEN TEMP=INT(TEMP)+1 ELSE TEMP=INT(TEMP)
100     DIGIT1=INT(TEMP/100) :  REM divide temperature into 3 digits
110     DIGIT2=INT(TEMP/10)-DIGIT1*10
120     DIGIT3=TEMP-DIGIT1*100-DIGIT2*10
130     XBY(0C000H)=80H :  REM select position 1 on display
140     XBY(0C200H)=DIGIT1+48 :  REM write ASCII codes of digits to display
150     XBY(0C200H)=DIGIT2+48
160     XBY(0C200H)=DIGIT3+48
170     XBY(0C200H)=0DFH :  REM write degree symbol to display
180     XBY(0C200H)=ASC(F) :  REM write "F" to display
190     WHILE 1=1
200     END
```

bit 1 is low, which means *S3* has been pressed.

Listing 9 combines features of Programs 6 and 7. It reads the temperature at *IC12*'s output, then displays it on *M1*. Lines 20 through 50 initialize *M1*. Lines 70 through 80 get and calculate the temperature at *IC12*. Lines 100 through 120 divide the temperature reading into three digits, and lines 130 through 180 write the three digits to the display. To give the correct character codes, 48 is added to each digit's value.

Finally, the listing in Program 10 gives the full-featured Tempwatch. With this program, the Tempwatch monitors temperature and displays the present, maximum, and minimum temperatures and the times they occurred.

Carefully key in this program, just as you did the previous ones and run it. You'll see three alternating displays, giving the three temperatures and times. If you don't see these displays, LIST the program and carefully check it against the one shown here. Rekey any portions of the program that contain errors in them. Use CONTROL/S and CONTROL/Q to stop and start the listing for this program on the monitor's screen.

On power-up, the time is set to 24:00. To set the correct time, momentarily press and release *S3*. The display will now show a rotating display of numbers from 1 to 24 (for a 24-hour clock). When the correct hour is displayed, once again press and release *S3*. You'll then see numbers from 0 to 59 rotate on the display. When the correct minute is displayed, again press and release *S3* and the program will return to normal operation, with its clock set to the correct time.

To test the capability of the Tempwatch to store maximum and minimum temperatures, heat and cool the sensor and monitor the results on the display. Heat the sensor by holding it near (not on) a heated soldering iron. When the sensed temperature has risen several degrees, move the sensor away from the soldering iron and allow it to return to room temperature. The maximum-temperature display should now show the highest value measured and the time it occurred.

Similarly, you can test the minimum-temperature display by spraying the sensor with an aerosol component cooler. The minimum-temperature display should show the coldest temperature measured, along with the time it occurred.

To reset the maximum and minimum temperatures to the current temperature, momentarily press and release *S2*.

If you have any problems in running this program, use the LIST com-

Program 10. Full-Featured Tempwatch Program

```
10      REM ****************Tempwatch program*************************
20      STRING 25,5
30      XTAL=4915200 :  CLOCK 1 : TIME=0
40      TEMP=0 : MIN=255 : MAX=0 : HOUR=24 : MINUTE=0
50      VREF=1.225 :  REM reference-diode voltage
60      $(1)="Now " : $(2)="Max " : $(3)="Min "
70      XBY(0C000H)=1 :  REM clear display
80      XBY(0C000H)=38H :  REM select 8-bit data, 2 logical lines
90      XBY(0C000H)=6 :  REM automatic increment of display address counter
100     XBY(0C000H)=0CH :  REM turn display on, cursor off
110     DO
120     ONTIME 60,770
130     XBY(0E000H)=255 :  REM write to adc to start convert
140     TEMP=XBY(0E000H)*VREF*200/255 :  REM get temperature
150     IF TEMP-INT(TEMP)<.5 THEN TEMP=INT(TEMP) ELSE TEMP=INT(TEMP)+1
160     IF TEMP>=MAX THEN MAX=TEMP : XHR=HOUR : XMIN=MINUTE
170     IF TEMP<=MIN THEN MIN=TEMP : NHR=HOUR : NMIN=MINUTE
180     N=1 : DTEMP=TEMP : DHR=HOUR : DMIN=MINUTE :  GOSUB 230
190     N=2 : DTEMP=MAX : DHR=XHR : DMIN=XMIN :  GOSUB 230
200     N=3 : DTEMP=MIN : DHR=NHR : DMIN=NMIN :  GOSUB 230
210     WHILE 1=1
220     END
230     REM ***********write-to-display subroutine*********************
240     XBY(0C000H)=80H :  REM select position 1 on display
250     FOR I=1 TO 4 : XBY(0C200H)=ASC($(N),I) :  NEXT  :  REM display message
260     REM display temperature (3 digits)
270     D1=INT(DTEMP/100) : D2=INT(DTEMP/10)-D1*10 : D3=DTEMP-D1*100-D2*10
280     IF D1=0 THEN XBY(0C200H)=32 ELSE XBY(0C200H)=D1+48
290     IF D1.AND.D2=0 THEN XBY(0C200H)=32 ELSE XBY(0C200H)=D2+48
300     XBY(0C200H)=D3+48
310     XBY(0C200H)=0DFH :  REM display degree symbol
320     XBY(0C000H)=0C0H :  REM select position 9 on display
330     XBY(0C200H)=ASC(F) : XBY(0C200H)=32
340     REM display time
350     IF DHR>9 THEN XBY(0C200H)=INT(DHR/10)+48 ELSE XBY(0C200H)=32
360     XBY(0C200H)=DHR-10*INT(DHR/10)+48 : XBY(0C200H)=ASC( : )
370     XBY(0C200H)=INT(DMIN/10)+48
380     XBY(0C200H)=DMIN-10*INT(DMIN/10)+48 : XBY(0C200H)=32
390     FOR I=1 TO 50
```

mand to review it and verify that all lines are entered correctly. (CONTROL/S and CONTROL/Q will stop and start the listing of a long program on the screen.)

MCS BASIC-52 is usually quite helpful in generating error messages, so these may also help you in tracking down the source of problems.

To save the program in EPROM, snap *B1* and *B2* into their battery connectors type "PROG" (do *not* type the quotation marks in this or any other instruction to key in a command), and the Microsys will program the EPROM with the program currently in RAM. An additional command allows you to configure Tempwatch to begin running immediately upon power-up, which frees you from having to connect to the terminal at all to use the Tempwatch.

To provide this capability, type "PROG2." This command is required in addition to the PROG command that saved the program. It causes the 8052 to run Program 1 in the EPROM immediately on power-up.

Now you can power down the Tempwatch, disconnect it from the terminal, and move it to a new location. When you power up again, the Tempwatch program runs automatically.

There are plenty of enhancements, in both the hardware and the software areas, that you can add to the Tempwatch. For example, you could add a temperature alarm or a program that measures and stores the temperatures at a particular time of day. Or, you can turn your Microsys into something else, such as an instrument to measure sound or light, to control relays, or to monitor or control whatever real-world conditions or devices you wish.

The great appeal of the Microsys is that you are ultimately in control. By adding appropriate inputs, outputs, and programs, you can tailor the Microsys to fit your own requirements, whatever they may be. ME

```
400    X=PORT1-4*INT(PORT1/4) :  REM check for switch press
410     IF X=1 THEN  GOSUB 450
420     IF X=2 THEN  GOSUB 730
430     NEXT I
440     RETURN
450     REM ****************set-clock subroutine************************
460    XBY(0C000H)=1 :  REM clear display
470    COUNT=24 : POS=0C2H : CHAR=58 : UNIT=HOUR :  GOSUB 510
480    COUNT=60 : POS=0C5H : CHAR=32 : UNIT=MINUTE :  GOSUB 510
490     GOSUB 730
500     RETURN
510     REM ************set hours, minutes subroutine******************
520     DO  : X=PORT1-4*INT(PORT1/4) :  UNTIL X>1 :  REM wait for switch open
530    SELECT=0
540     DO
550     IF COUNT=24 THEN J=1 ELSE J=0
560     DO
570    DELAY=0
580     DO
590    DELAY=DELAY+1
600    XBY(0C000H)=POS :  REM set display postion
610    XBY(0C200H)=INT(J/10)+48 :  REM display digit 1 of hour or minute
620    XBY(0C200H)=J-INT(J/10)*10+48 :  REM display digit 2 of hour or minute
630    XBY(0C200H)=CHAR :  REM display colon or space
640    X=PORT1-4*INT(PORT1/4) :  REM check for switch press
650     UNTIL DELAY=6.OR.X<2
660     IF X=1 THEN SELECT=1 : TIME=0
670    J=J+1
680     UNTIL J=COUNT+1.OR.SELECT=1
690     UNTIL SELECT=1
700     IF UNIT=HOUR THEN HOUR=J-1 ELSE MINUTE=J-1
710     DO  : X=PORT1-4*INT(PORT1/4) :  UNTIL X=3 :  REM wait for switch open
720     RETURN
730     REM ***********reset max, min subroutine************************
740    MAX=TEMP : XHR=HOUR : XMIN=MINUTE
750    MIN=TEMP : NHR=HOUR : NMIN=MINUTE
760     RETURN
770     REM ***************update-time subroutine**********************
780    TIME=TIME-60 : MINUTE=MINUTE+1
790     IF MINUTE=60 THEN HOUR=HOUR+1 : MINUTE=0
800     IF HOUR=25 THEN HOUR=1
810     RETI
```