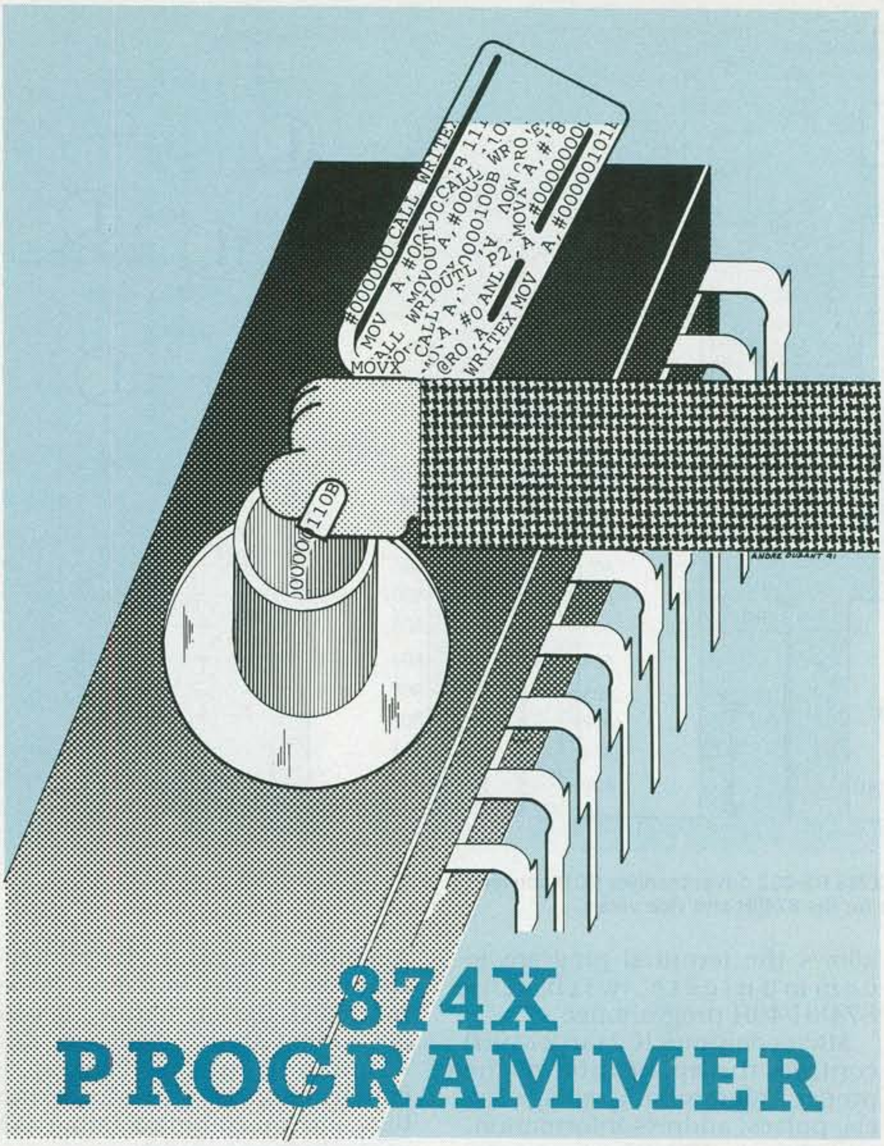


874X PROGRAMMER

USING MICROPROCESSORS IN YOUR home projects often turns out to be a complex ordeal. The typical microprocessor project consists of the processor, EPROM, RAM, address decoder circuitry, clock circuitry, input/output ports, and the ever present but essential "glue" components. Things can get a little more complicated if a special-purpose IC like an analog-to-digital converter is thrown in. Then comes the pleasure of putting everything on a circuit board—and most of the time the complexity of the circuit necessitates the use of a double-sided board!

But don't despair. There is an easier and more enjoyable way to exercise your hobby. This article will open the door to a more efficient hobbyist approach to designing microprocessor-based projects. Imagine how much design time could be saved if you had a processor, EPROM, RAM, clock, and input/output ports already integrated into a standard 40-pin package.

Such devices already exist, of course—they're called microcontrollers. Several different varieties of these microcontrollers are now readily and inexpensively available. All you really need to use them is a microcontroller programmer, and we're going to show you how to build one in just one evening for under $50. The programmer is good for the 8748H and 8749H series of microcontrollers made by Intel.

The 8748/49H

The 8748/49H is commonly referred to as a single-component 8-bit microcomputer. The instruction sets for the 8748H and 8749H are identical. The 8749H contains 2K of EPROM and 128 bytes of RAM, while the 8748H contains 1K of EPROM and 64 bytes of RAM. Although that doesn't sound like a lot of memory, you'll find the amount of EPROM and RAM to be more than adequate for most controller applications. And if you do require extra RAM, you can hang it outboard just as you would with any other processor IC.

Both IC's include an interval timer/event counter, two single-level interrupts, an internal oscillator, a true bi-directional bus, two latched quasi-bidirectional I/O ports, two testable input pins, and an 8-bit processor that executes over 96 instructions with most of them consisting of a single byte. If you're short on I/O or memory, the 8748H/49H will accommodate most common peripheral circuitry available for other microprocessors. A minimum circuit configuration consists of the 8748H/49H, a crystal, two 27-pF disk capacitors, a 5-volt DC power source, and a 1-μF reset capacitor.

Software

There are many cross assemblers for the 8748H/49H available in the public domain, and many more advertised by reputable electronics distributors. Just choose one that fits your needs and budget. One cross-assembler software package that can be used with PC-compatible computers is contained in a ZIP file (TASM.ZIP) that's included as part of a larger ZIP file containing all software relevant to this article. The larger ZIP file (874XPGR.ZIP) is available on the RE-BBS (516) 293-2283, 1200/2400, 8N1.

The 8748H/49H lends itself well to applications that require I/O port activity and serial communications (RS-232) with a terminal or supervisory program. This project was designed to take advantage of both. Most of the data storage and screen information are maintained and presented to the user by the terminal program, PROG.EXE, which runs on a PC-compatible computer. The terminal program is DOS based, so you may have to modify the source code (PROG.BAS) to run on a different computer. The software listing

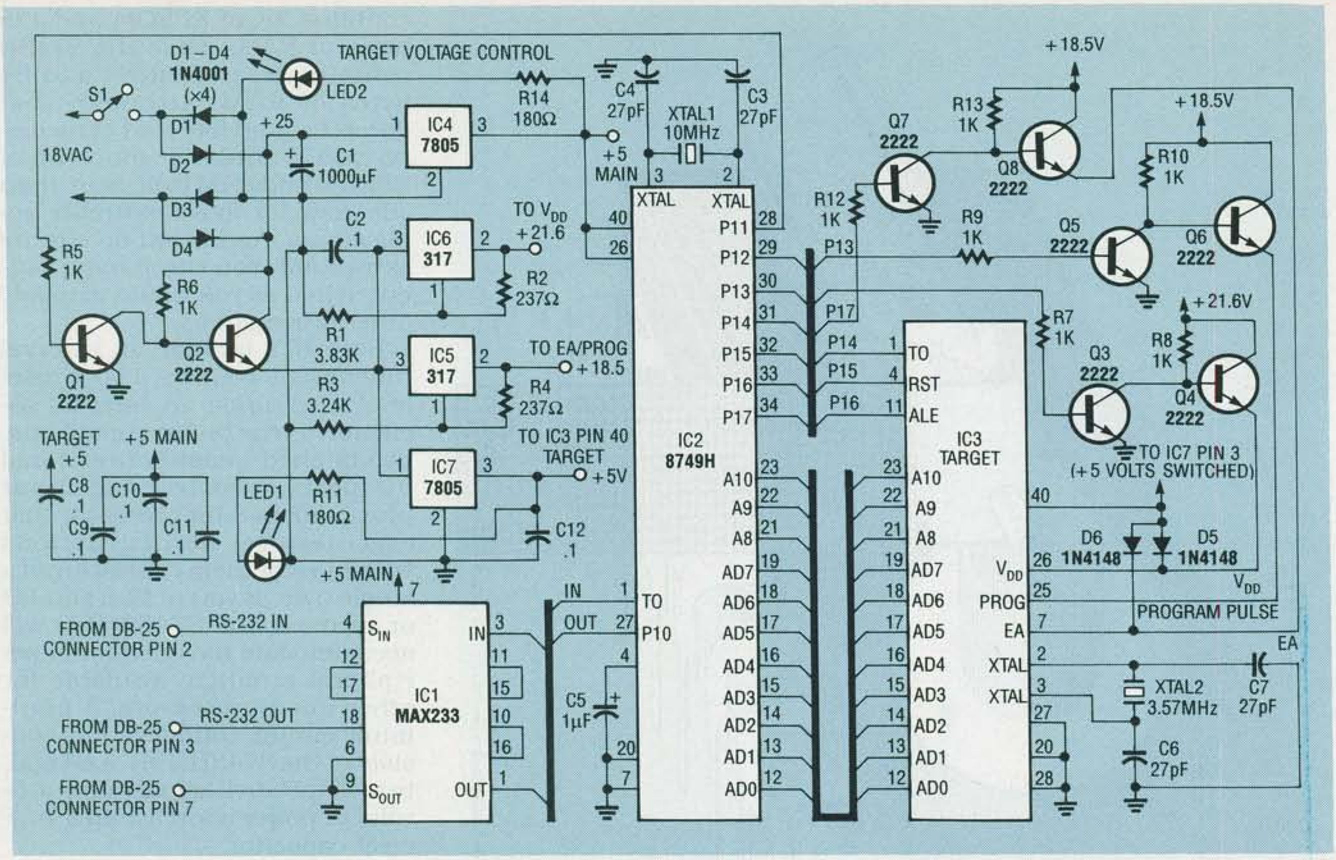


FIG. 1—PROGRAMMER SCHEMATIC. The MAX233 RS-232 driver/receiver (IC1) converts the signals from your serial port to TTL levels for the 8749H and vice versa.

for the terminal program is unfortunately too large to print here, but it is contained in the main ZIP file (874XPGR.ZIP).

Any data or commands are sent serially at 2400 bits per second from the terminal program via serial port to the programmer. The programmer's processor acts on the received data and returns any necessary data to the terminal program. This eliminates the housekeeping functions that would normally be performed by the programmer's processor, and thus simplifies both the hardware and the software of the programmer module.

Circuitry

Looking at the schematic in Fig. 1, IC1 is a MAX233 RS-232 driver/receiver. Its purpose is to convert the signals from your serial port to TTL levels for the 8749H and vice versa. The input serial data stream is fed into the TO input of the 8749H programmer while the output data stream is fed from I/O Port 1.0. The combination of the driver/receiver IC, the built-in hardware of the 8749H, and firmware in EPROM allows the terminal program to communicate with the 8748H/49H programmer.

Microcomputer IC2 (an 8749H) controls the application of the proper programming voltage levels, pulses, address information, and data to IC3 (the target device), which is installed in a ZIF (zero insertion force) socket when programming. The code for IC2 (874XCODE.HEX) is also contained in the ZIP file 874XPGR.ZIP. (There is also a binary version of the code, 874XCODE.BIN, which is also contained in the ZIP file.) The bidirectional ports contained in IC2 latch output data and read input data that is latched onto an external port by another device. Traditionally that would be handled by both a 74LS373 octal latch used as an output port and a companion 74LS244 used as an input port. Our programmer contains no external latches or address decode circuitry in either the data bus or control ports.

Data and address information are multiplexed on the bus pins AD0–AD7. The bus pins behave in a similar fashion to the bidirectional port pins but tend to be more TTL-like in nature. If you get a data sheet, study the differences in internal hardware construction as it pertains to the bus and quasi-bidirectional I/O ports. In the case of the programmer hardware, the target, IC3, and main processor, IC2, alternately latch output data on their respective busses to be read by the opposing processor's bus. Address pins A8–A10 are actually IC2's I/O port pins P.0–P.2, and are used as latched output pins. I/O port P1 is used to control the transistor pairs that supply the correct programming voltages to the target device. Since all of the bidirectional I/O pins can drive one TTL load, port P1 is also used to set up TTL logic levels on IC2's port pins P1.4–P1.6 that connect to the target device directly.

Power for IC1 and IC2 is supplied via voltage regulator IC4 and associated circuitry. Crystal XTAL1 along with the two 27-pF capacitors supply the feedback path for the on-chip oscillator. Since precise clock periods are required to generate timing for

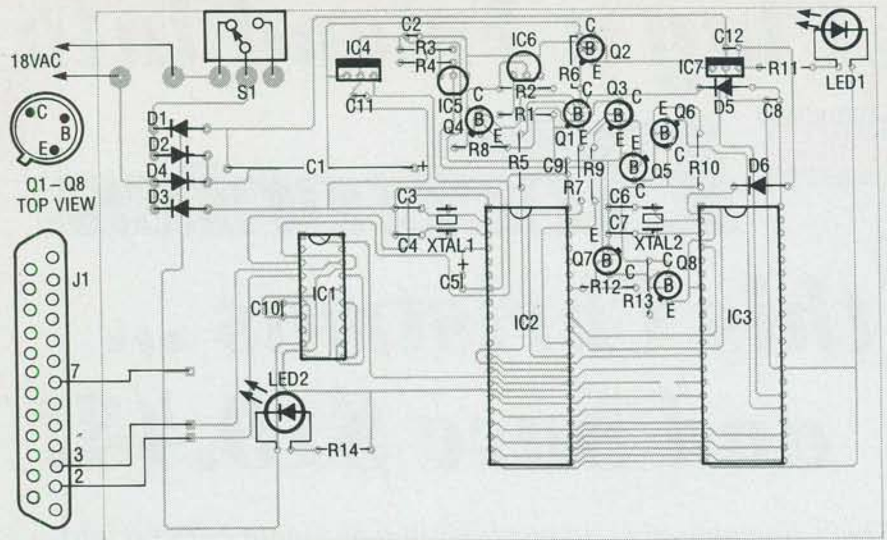


FIG. 2—FIRST INSTALL THE POWER SUPPLY DIODES D1–D4, filter capacitor C1, switch S1, and connect the 18-VAC transformer. With no other parts installed, you should have a full-wave bridge circuit with a +25-volt DC output across C1.

the serial data stream and programming pulses, a crystal-controlled oscillator is essential.

The 1-μF tantalum capacitor, C5, resets the microcomputer. Note the absence of the reset switch, and don't be tempted to add one. If the power is toggled or the processor is reset while a target device is socketed, permanent damage will result to the target due to transient voltages on the transistor pairs generated by a main processor reset. Therefore, NEVER apply or terminate power while a target is socketed. You may add a reset switch across the 1-μF capacitor as shown in the example circuits we'll look at later.

The device to be programmed, IC3, can be either a 8748H or 8749H. You select the type when you run the terminal program. The target device needs +5-, +18-, and +21-volt DC power sources to effect the programming/verify process. The voltages are derived from voltage regulators IC5–IC7; IC5 and IC6 are standard configurations of the low-power "LZ" version of the LM317.

Transistor pairs Q3-Q4, Q5-Q6, and Q7-Q8 provide the high-voltage switching functions necessary for the programming and verification of the target device. Voltage regulator IC7, a 7805T, supplies +5-volts DC to the target during programming and verification. Light-emitting diode LED1 is active when power is applied to the target device. Transistor pair Q1-Q2 is used to switch all operating power to the target device (IC3).

The 8748H/49H needs a clock signal to move data internally. Crystal XTAL2 along with its 27-pF capacitors are used to supply a clock signal for the target device. Any crystal between 3 and 4 MHz will suffice. The target clock period is not critical to the programming process.

The sequence used to program IC3 is similar to programming an ordinary EPROM; the target device is powered up in program mode. Address information is passed to the target by IC2. Then, data information is latched out of IC2 to the target. A pulse is applied to the target's PROG pin and the verification process follows. If verification is good, then the process is repeated for the next byte, and so on.

To sum it up, IC1 converts RS-232 voltage levels to TTL voltage levels and vice versa. Controller IC2 provides communication with the terminal program via a 2400 bits-per-second serial link, provides address and data information to the target, provides precisely timed pulses to the target, and provides voltage-switching information to the transistor pairs that interface with the target. Target IC3 is programmed with the data you specify using the terminal program in conjunction with IC2. All of this is done with a single-component microcomputer on a single sided board!

PARTS LIST

All resistors are ¼-watt.
R1—3830 ohms, 1%
R2, R4—237 ohms, 1%
R3—3240 ohms, 1%
R5–R10, R12, R13—1000 ohms, 5%
R11, R14—180 ohms, 5%

Capacitors
C1—1000 μF, 35V, electrolytic
C2, C8–C12—0.1 μF, 50V, Mylar
C3, C4, C6, C7—27 pF, disk
C5—1 μF, 35V, tantalum

Semiconductors
IC1—MAX233 RS-232 driver/receiver
IC2—8749H microcontroller (programmed)
IC3—8748H or 8749H microcontroller
IC4, IC7—7805T 5-volt regulator
IC5, IC6—LM317LZ low-power adjustable regulator
D1–D4—1N4001 diode
D5, D6—1N4148 diode
LED1, LED2—light-emitting diodes, choose color to suit taste
Q1–Q8—2N2222A NPN transistor

Other components
S1—SPST toggle switch
XTAL1—10-MHz crystal
XTAL2—3.57-MHz crystal

Miscellaneous: 18VAC 1.35A transformer, heatsink for IC4, 40-pin ZIF socket (for target IC3), 40-pin IC socket (for IC2), 20-pin IC socket (for IC1), 25-pin right-angle female DB-25 connector (optional), serial cable, PC board, wire, solder, etc.

Note: The following items are available from F. Eady, PO Box 541222, Merritt Island, FL 32954:
- **A kit of parts including a preprogrammed microcontroller (not including the transformer, ZIF socket, serial cable, or 25-pin connector)—$49.95 + $5.00 S&H.**
- **PC board only—$15.00 + $5.00 S&H.**
- **Software on floppy disk—$5.00 postpaid.**

Check or money orders only.
For technical assistance call (407) 454-9905.

Construction

The first thing you must do is etch and drill a PC board from the pattern we've provided—or purchase a ready-to-use PC board from the source mentioned in the parts list. As shown in Fig. 2, start assembly by installing power supply diodes D1–D4 and filter capacitor C1. Mount the switch S1 and connect the 18-VAC transformer. At this point, with no other components mounted yet, you should have a full-wave bridge circuit that outputs +25-volts DC measured across C1.

Once you are satisfied with the 25-volts DC across C1, install voltage regulator IC4 and bypass capacitor C11. Be sure to install a heatsink on IC4. Apply power and measure the output of IC4; you should have +5 volts DC at the output (pin 3). If so, install the rest of the power supply components: R1–R4, C2, C12, IC5, IC6, and IC7. To check the voltage levels from those regulators you must also install transistor pair Q1-Q2, since this pair supplies power to the regulators. Install LED1 and LED2, along with current-limiting resistors R11 and R14.

Once all of the power components are installed (with no IC's yet installed), apply power and both LED's should light. You should be able to read the voltages on the outputs of the voltage regulators (IC5–IC7). You can now jumper R5 (that goes to the base of Q1) to +5 volts; that should turn off power to the target device and extinguish LED1. That simulates a high TTL level that would normally come from the main processor, IC2, and verifies that the target power-shut off circuitry is working properly.

Finish the assembly by installing the remaining transistor pairs. You can test the transistor pairs and their switching by jumping the base input resistors to +5 volts and noting the change in output voltage at the pair's open-emitter output. The V_{DD} pair should toggle between +21 and +5 volts. The Program Pulse pair should toggle between +18 volts and floating. The EA pair should toggle between +18 and +5 volts. If not, make sure that you have installed blocking diodes D5 and D6 and also recheck the rest of your work.

The prototype used a modified right-angle DB-25 connector for J1, mounted directly to the board using the appropriate nuts and bolts. You do not have to use a connector; you can solder your cable directly to the PC pads if you wish. If you do decide to add the DB-25 connector, cut off all of the pins except 2, 3, and 7. Note that no holes are provided for the

LISTING 1

```
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;                                                                      ;
                    ;         EXAMPLE 1 - INTELLIGENT DISPLAY DRIVER                       ;
                    ;                                                                      ;
                    ;         THIS ROUTINE WRITES "8748" TO THE NSM 1416 4 DIGIT DISPLAY   ;
                    ;                                                                      ;
                    ;         LABEL A CODE WRITES THE LETTER "X" TO ALL 4 DIGITS           ;
                    ;                                                                      ;
                    ;         LABEL B CLEARS THE CURSOR WITHIN THE NSM 1416                ;
                    ;                                                                      ;
                    ;         LABEL C WRITES "8748" INTO DIGITS 0-3                        ;
                    ;                                                                      ;
                    ;         LABEL D HALTS THE PROGRAM BY LOOPING ON ITSELF               ;
                    ;                                                                      ;
                    ;         SUBROUTINE WRITEX PERFORMS THE WRITE FUNCTION                ;
                    ;                                                                      ;
                    ;                                                                      ;
                    ; NOTE:   CE- PIN ON THE NSM 1416 IS GROUNDED                          ;
                    ;                                                                      ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
ADDRESS
  |
  |   CODE      LABEL
  |    |          |
0000 23 07        A         MOV  A,#00000111B       ;WRITE LETTER X TO ALL 4 DIGITS
0002 14 2F                  CALL WRITEX             ;WRITE TO DIGIT 3
0004 23 06                  MOV  A,#00000110B
0006 14 2F                  CALL WRITEX             ;WRITE TO DIGIT 2
0008 23 05                  MOV  A,#00000101B
000A 14 2F                  CALL WRITEX             ;WRITE TO DIGIT 1
000C 23 04                  MOV  A,#00000100B
000E 14 2F                  CALL WRITEX             ;WRITE TO DIGIT 0

0010 9A 00        B         ANL  P2,#00000000B      ;CLEAR THE CURSOR
0012 23 00                  MOV  A,#0
0014 90                     MOVX @R0,A

0015 23 07        C         MOV  A,#000000111B      ;WRITE LETTER 8 TO DIGIT 3
0017 3A                     OUTL P2,A
0018 23 38                  MOV  A,#'8'
001A 90                     MOVX @R0,A

001B 23 06                  MOV  A,#000000110B      ;WRITE LETTER 7 TO DIGIT 2
001D 3A                     OUTL P2,A
001E 23 37                  MOV  A,#'7'
0020 90                     MOVX @R0,A

0021 23 05                  MOV  A,#000000101B      ;WRITE LETTER 4 TO DIGIT 1
0023 3A                     OUTL P2,A
0024 23 34                  MOV  A,#'4'
0026 90                     MOVX @R0,A

0027 23 04                  MOV  A,#000000100B      ;WRITE LETTER 8 TO DIGIT 0
0029 3A                     OUTL P2,A
002A 23 38                  MOV  A,#'8'
002C 90                     MOVX @R0,A

002D 04 2D        D         JMP  $                  ;LOOP HERE FOREVER

002F 3A         WRITEX      OUTL P2,A               ;WRITE LETTER X SUBROUTINE
0030 23 58                  MOV  A,#'X'
0032 90                     MOVX @R0,A
0033 83                     RET
```

LISTING 2

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;                                                                          ;
;         EXAMPLE 2 - EXPENSIVE LED BLINKER                                ;
;                                                                          ;
;         THIS ROUTINE BLINKS AN LED                                       ;
;                                                                          ;
;         LABEL A WRITES 01 HEX TO THE BUS - TURNS ON LED                  ;
;                                                                          ;
;         LABEL B WRITES 00 HEX TO THE BUS - TURNS OFF LED                 ;
;                                                                          ;
;         LABEL C JUMPS TO THE BEGINNING LABEL A                           ;
;                                                                          ;
;         SUBROUTING KILLTIME DECREMENTS 2 REGISTERS TO CREATE A DELAY     ;
;                                                                          ;
;                                                                          ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ADDRESS
 |
 |   CODE    LABEL
 |    |        |
0000 23 01     A         MOV  A,#00000001B    ;WRITE A BINARY 00000001 TO THE DATA BUS
0002 02                  OUTL BUS,A           ;TURN ON TRANSISTOR AND LED

0003 14 0C               CALL KILLTIME        ;KILL SOME TIME

0005 23 00     B         MOV  A,#00000000B    ;WRITE A BINARY 00000000 TO DATA BUS
0007 02                  OUTL BUS,A           ;TURN OFF TRANSISTOR AND LED

0008 14 0C               CALL KILLTIME        ;KILL SOME TIME

000A 04 00     C         JMP  A               ;GO DO IT ALL AGAIN

000C B8 FF     KILLTIME  MOV  R0,#0FFH        ;LOAD REGISTER R0 WITH FF HEX

000E B9 FF     INNERLOOP MOV  R1,#0FFH        ;LOAD REGISTER R1 WITH FF HEX
0010 E9 10               DJNZ R1,$            ;DECREMENT R1 TO 00

0012 E8 0E               DJNZ R0,INNERLOOP    ;DECREMENT R0 - IF R0 NOT EQUAL 0 THEN DO INNERLOOP

0014 83                  RET                  ;DONE KILLING TIME
```

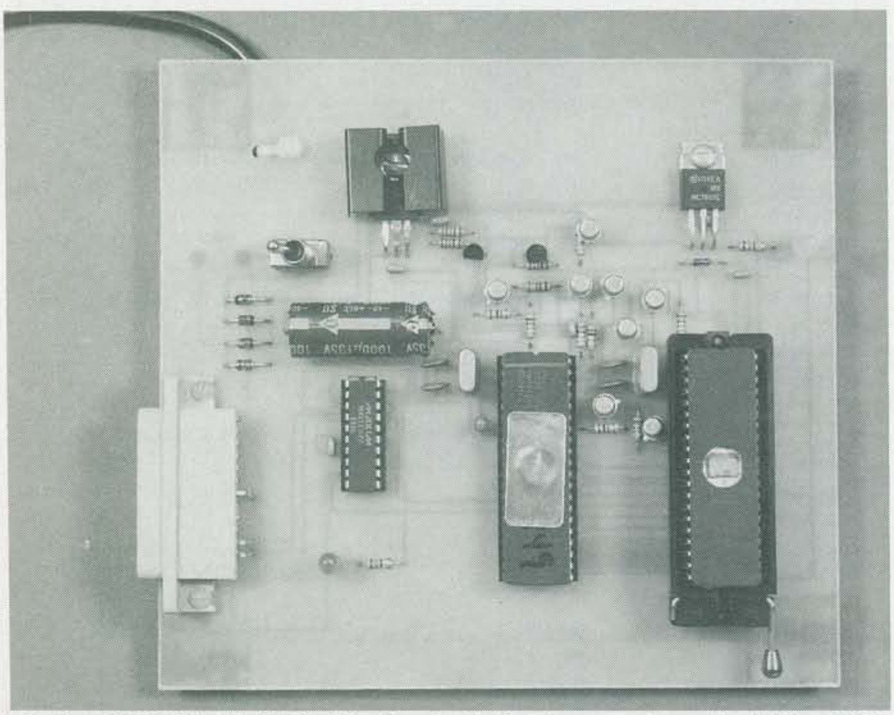


FIG. 3—THE COMPLETED UNIT. Double check all voltages on the pads of IC1 and IC2 before installing them in their sockets. When you apply power, LED2 should illuminate.

DB-25 mounting hardware because of the many different styles of connectors. Install your DB-25 connector and drill mounting holes accordingly.

It's a good idea to double check all voltages on the pads of IC1 and IC2 before installing them. When you are satisfied that all is well, install the IC's. Apply power and LED2, the main power LED, should illuminate. LED1 should not illuminate indicating that IC2 has initialized transistor pair Q1-Q2 properly and no voltage is present at the target ZIF socket (IC3). Figure 3 shows the completed unit.

Using the programmer

Connect the serial port from your computer to the programmer's serial connections and execute the terminal program at this time. If all is well, "READY FOR COMMAND" should appear on your screen. This indicates that the terminal program has established communications with the programmer. If "UNABLE TO COMMUNICATE WITH PROGRAMMER" appears, something is not right with the programmer or your serial port connections.

Power up the programmer and start the terminal program (PROG.EXE)—if you haven't done so already. You should get "READY FOR COMMAND" on the screen before beginning. You may socket the target device in the IC3 ZIF socket any time after you power up and any time LED1 is not on.

Note the list of commands. You may execute a command by typing the letter contained in parentheses preceding the command. The terminal program has been written so that its use will be obvious to the user. For those of you not familiar with programming any sort of programmable device, the basic steps are:

1. Make sure the target device is blank.
2. Load the binary image of the file you want to program into the terminal program.
3. Program the device.

Some sample programs and circuits have been included that use the 8748H in a minimum mode configuration. The intent here is to allow you to enter the machine code into a file using a

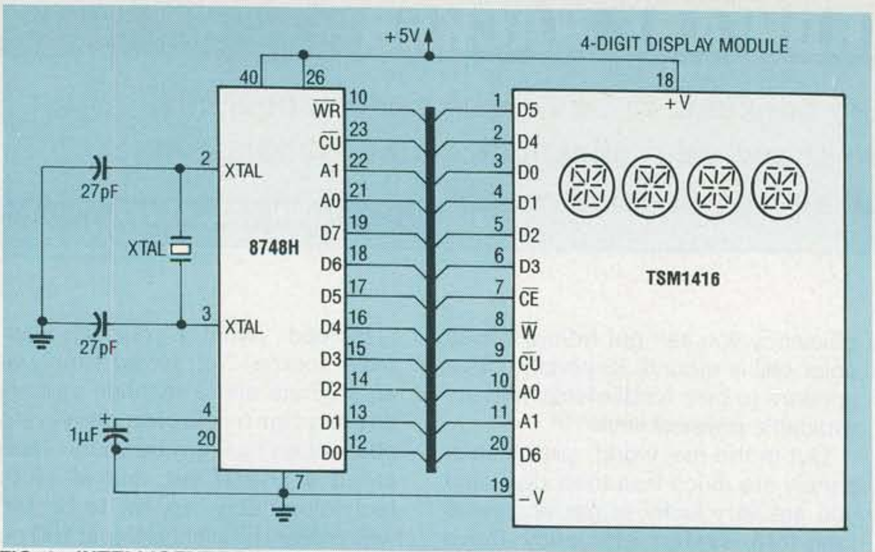
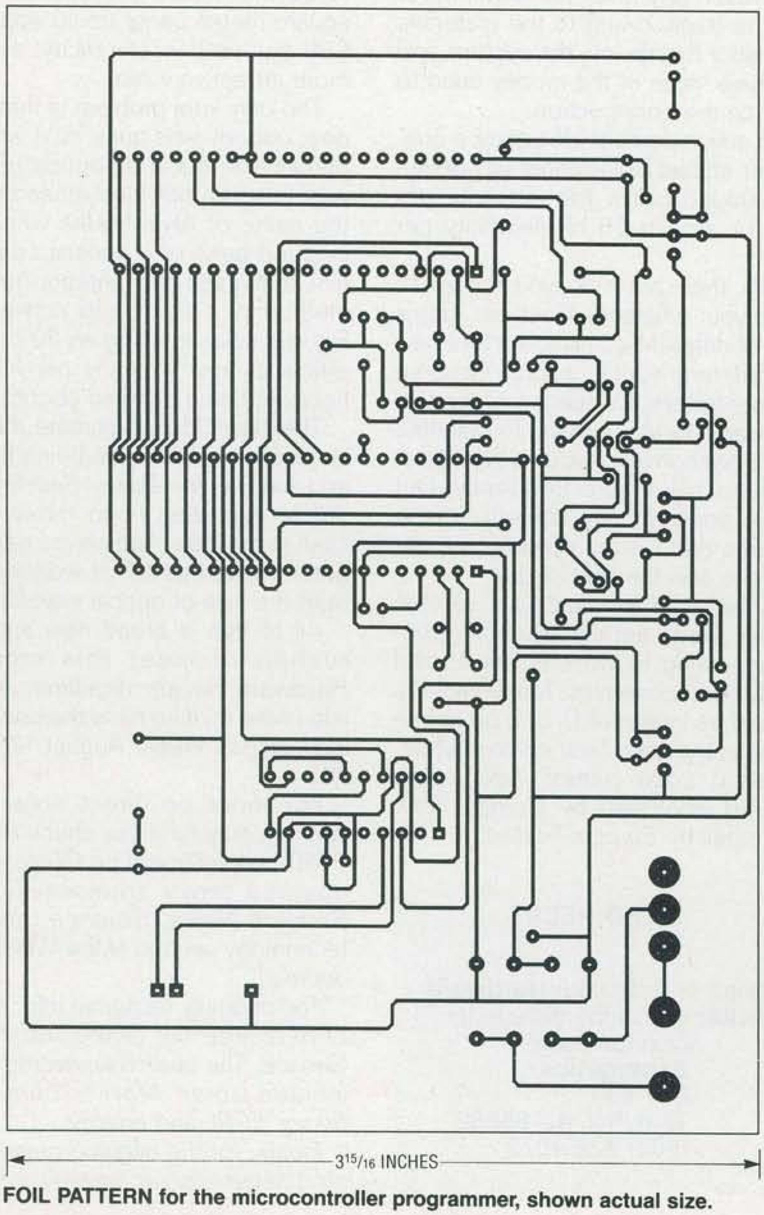


FIG. 4—INTELLIGENT DRIVER for a 4-digit display module. The circuit will first put an "X" in all four digits and then display "8748." The accompanying software is shown in Listing 1. The TSM1416 4-digit display module is manufactured by Three-Five Systems, Inc.



FOIL PATTERN for the microcontroller programmer, shown actual size.

binary editor and then program that file into a target 8748H. This eliminates the initial need for a cross assembler and gives you the opportunity to experiment with minimum cost and effort. The circuits presented illustrate the advantages of using an integrated microcomputer like the 8748H—and they're simple enough to be built on an experimenter's breadboard.

Figure 4 shows an intelligent driver for a 4-digit display module, and Listing 1 shows the accompanying software. The circuit will first put an "X" in all four digits and then display "8748." While it's nothing fancy, it does show you how to make the display work. The TSM1416 4-digit display module is manufactured by Three-Five Systems, Inc.

Our second example circuit, shown in Fig. 5, is our "expensive LED blinker circuit." We say "expensive" because you certainly

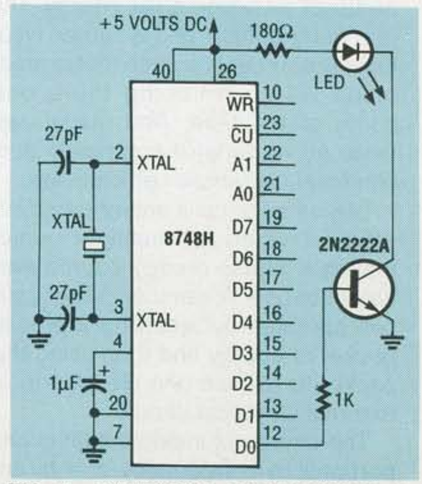


FIG. 5—EXPENSIVE LED BLINKER circuit. While it is somewhat overkill, it is very useful as a teaching tool. The software for this circuit is shown in Listing 2.

don't need a microcontroller to turn an LED on and off—the circuit is somewhat of an overkill. However, the circuit's simplicity becomes extremely beneficial when it's being used as a teaching tool. That way you can concentrate on the microcontroller's operation. The software for the Fig. 5 example is shown in Listing 2.

For the two example circuits we've provided, you can copy the software routines directly from the listings or download them from the RE-BBS as part of the main ZIP file (874XPGR.ZIP). **R-E**