In previous installments we built computer-assisted test instruments to measure voltage, resistance, and capacitance. In this final installment, we'll round out our component inspection system (CIS) with a PC-based IC tester. Our tester can test all 14- and 16-pin CMOS and TTL IC's. It can also be modified to test any IC that requires digital inputs and provides digital outputs.

Testing methods

There are two common ways of testing digital circuits and IC's: transition and state testing. In transition testing, you provide a series of signals to the circuit and note the output transitions. Transition testing is beneficial when you are interested in the AC characteristics (slew rate, propagation time, etc.) of a circuit. Transition testing is, however, expensive and time-consuming.

With state testing, you change one input at a time and note any changes in the outputs. State testing can't check the high-speed response of an IC, but it does allow you to exercise an IC and determine whether it's good or bad. Our tester uses the technique of state-testing.

Specifications

An IC may be fully specified by a connection diagram, a block (or function) diagram, and a truth table. The connection diagram shows the physical layout and name of each pin. The block diagram shows the internal functions of the IC. The truth table lists outputs provided by various input combinations.

For example, Fig. 1 shows a connection diagram (*a*), function diagram (*b*), and truth table (*c*) for a 4011 CMOS quad two-input NAND gate. The connection diagram shows that the 4011 is a 14-pin IC with ground (V_{SS}) at pin 7 and +V (V_{DD}) at pin 14.

The other pins are labelled as well, but to find out what the labels mean, we must refer to the functional diagram (Fig. 1-*b*). Here we see four NAND gates, each with two inputs (A and B) and one output (Y). To find out how each gate works, we must examine the truth table (Fig. 1-*c*). For each gate, the output is high unless both inputs are high, in which case, the output is low.

Testing steps

Using the connection and functional diagrams, as well as the truth table, we must follow four steps to test an IC.

1. Determine its pinout, its logic functions, and its truth table.
2. Specify a series of inputs to exercise all of the functions.
3. Determine the proper outputs for each input combination.
4. Perform physical testing.

You get pinout, functional, and truth-table information from data books, which can be purchased from IC distributors. (They are also available at larger local libraries). Using that information, you then plan a series of inputs. Using the truth table as a guide, step through each of the possible inputs (or as many as necessary to determine proper function) and determine the proper outputs. A programming form (like that shown in Fig. 2) can be helpful in setting up a test program.

After setting up a test program, it's time to perform a physical test. Physical testing includes these four steps:

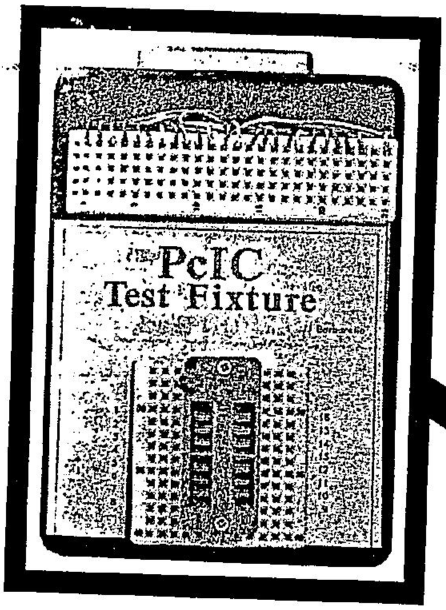
1. Initialize the IC to a predetermined state.
2. Apply inputs.
3. Monitor outputs.
4. Compare to what should occur; determine pass or fail.

For example, Fig. 2 shows a test program for the 4011. That program first initializes all inputs high. Actually, any initialization is fine (all inputs low, for example) as long as testing begins in a known, repeatable state. The actual test begins at step 9. One input on each gate is brought low, changing that gate's output from low to high. Then the input is returned low, making the output also return low.

The tester

For maximum flexibility, we designed a simple interface card, shown in Fig. 3. The heart of our card is a standard 8255 (IC2); the only other active component is a 74LS138 decoder (IC1) that allows you to choose an I/O port for communicating with the card. To avoid conflicts with other hardware devices, DIP switch S1 allows you to choose one of several different ports, ranging from 512 through 736 decimal.

The 8255 can function in several modes; we use Mode 0, in which each eight-bit register (PA, PB, and PC) can be programmed for input or output. A separate control register specifies how the data registers work.



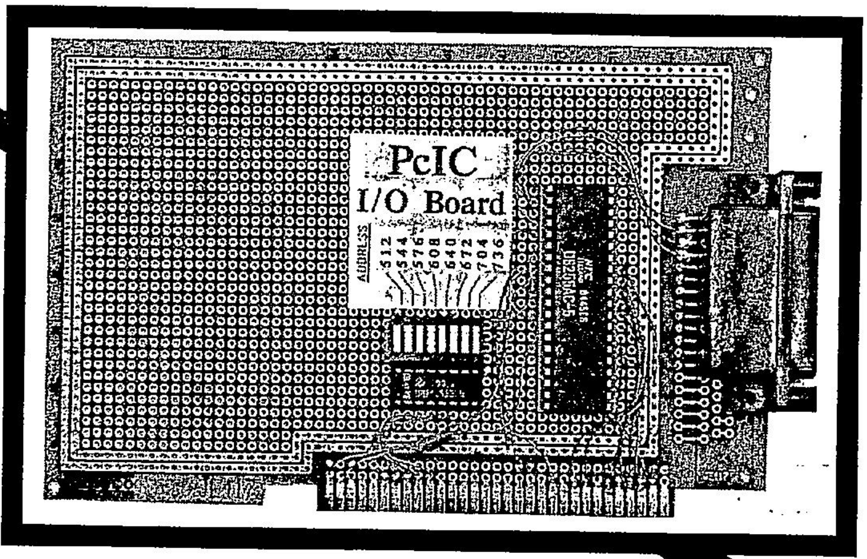
If we program PA and PC as outputs and PB as input, we have eight inputs and 14 outputs. Conversely, programming PA and PC as inputs and PB as outputs gives us eight outputs and 14 inputs. With these two possibilities, we can test any 14- or 16-pin IC with any combination of inputs and outputs.

A standard 25-pin cable connects J2 on the I/O board in the PC to the test fixture, shown in Fig. 4, which consists of J1, two solderless breadboards, and SO1, a 16-pin zero-insertion-force (ZIF) socket. That arrangement allows any of the 24 lines from J1 to be connected to any of the 16 pins on SO1 with jumper wires.

Software

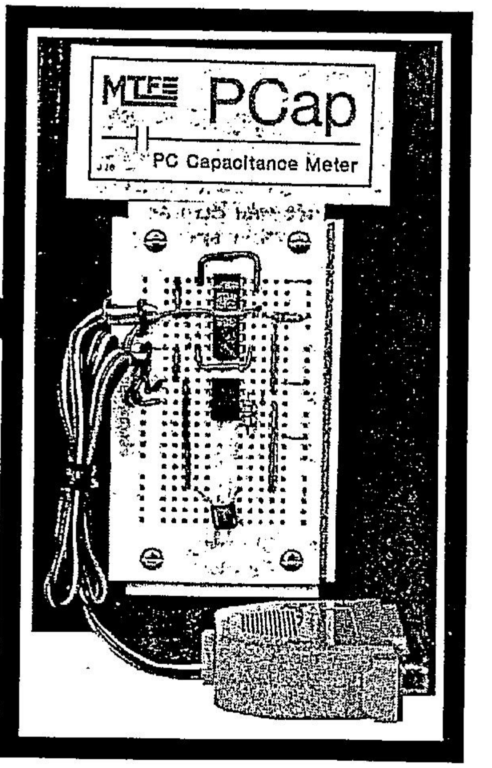
All our test and control programs are written in BASIC. Unfortunately, there is not enough space to present complete listings, but we will provide enough sample code so that you can understand the principles involved. In addition, we'll post compiled programs on the RE-BBS (516-293-2283, 1200/2400, 8N1; look for file PCTEST3.ZIP); software is also available on disk directly from the author, as discussed in the parts list.

EXPERIMENTING WITH PC-BASED TEST EQUIPMENT



Build this low-cost (under $100) PC-based IC tester!

JAMES J. BARBARELLO



The software needs to know the port address of the tester. If we define A as the address of PA (the decimal address selected by S1), then B=A+1, C=B+1, and D=C+1, where B is PB, C is PC, and D is the control register. For example, if S1 is set for address 640, then:

```
10 A=640: B=641: C=642: D=643
```

Now, to set up the 8255 PPI, we must send an appropriate value to the control register. For PA=output, PB=input, and PC=output, that value is 130. For PA=input, PB=output, PC=input, the value is 153. For example, to set PA and PC as outputs, and PB as input, then:

```
20 OUT D, 130
```

After setting up the 8255, we can send and receive information to and from the registers. For example, to bring PA lines PA0 and PA2 high, and then sense those lines to make sure everything is working correctly:

```
30 OUT A, 5
40 IF INP(A) 5 THEN PRINT "There's A
   Problem With PA":END
50 IF INP(A) = 5 THEN PRINT "Peripheral
   Register A is OK":END
```

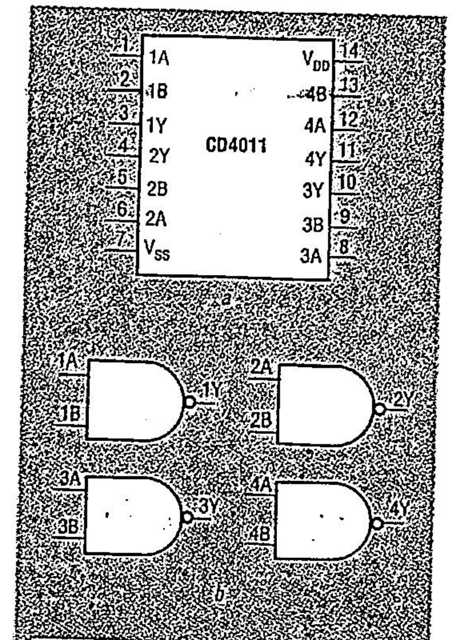
To sense the status of the input register, PB, use the same approach as in lines 40 and 50. For example, to sense the status of PB5 (J2-14):

```
60 X=INP(B): X = X AND 8 : PRINT "PB5
   is";
70 IF X=0 THEN PRINT "Low"
80 IF X=8 THEN PRINT "High"
```

The value 8 is the decimal number associated with PB5. ANDing the result of the input (INP) isolates its value from the other PB bits. The value 8 is called a "mask." Mask values for bits 0 through 7, respectively, are 1, 2, 4, 8, 16, 32, 64, 128.

Combining ideas, we can create a small test program to tell us whether the tester is working at the selected address. Refer to Listing 1.

It's important to remember that outputs remain constant until you change them—in other words, outputs are latched. Inputs, on the other hand, are not latched. The value returned is what exists at the instant you sense it. If a signal changes a short time before you sense it, you would not see the change. If you want to check for a change in input, you can create a loop that continually senses until a change occurs, or until the program has gone through the loop a specified number of times. For example, the fragment in Listing 2 checks input status on PB5 until it finds a high or has looped 500 times.



INPUTS		OUTPUT
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

c

FIG. 1—IC SPECIFICATIONS. It takes a pinout (*a*), functional diagram (*b*), and a truth table (*c*) to fully specify an IC.

Components
C1—0.1 µF, Mylar
IC1—8255 PPI
IC2—74LS138 3/8 decoder
J1—DB-25 female D connector, chassis mount
J2—DB-25 female D connector, right-angle PC mount
S1—8-position DIP switch
SO1—16-pin ZIF socket

Miscellaneous: 16-pin and 40-pin low-profile IC sockets, solderless breadboard, case, DB-25 male/male 10-foot cable, PC proto card, wire, solder, etc.

Note: The following items are available from JJ Barbarello, RD3, Box 241H, Tennent Road, Manalapan, NJ 07726: Double Sided I/O PC Board (PCIO), $20.00; Complete kit with all parts (no programs or libraries), $85.00; Programs on 5¼" or 3½" PC disk, $10.00; Program and libraries on 5¼" or 3½" PC disk, $35.00; Complete Kit with all parts, programs, and libraries, $99.95. Programs include both BASIC and compiled versions of CHNGADR, ICSETUP, IC, and ICAUTO. Libraries include both .DEF and .DAT files for the most common TTL (74xx series) and CMOS (4xxx series) IC's. NJ residents must add 7% sales tax.

PC IC PROGRAMMING SHEET   Sheet 1 of 1

IC TYPE: CD 4011 CMOS Quad 2-In NAND Gate

REMARKS: Inputs=1,2,5,6,8,9,12,13. Outputs=3,4,10,11.

INTERCONNECTS (From ICSETUP Program):

IC:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J1:	1	2	9	10	3	4	23	N/A	N/A	5	6	11	12	7	8	25

STEP No.	IC PINS																NOTES
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
1	1		X	X						X	X						Init,
2		1	X	X						X	X						X= Don't
3			X	X	1					X	X						Care
4			X	X		1				X	X						
5			X	X				1		X	X						
6			X	X					1	X	X						
7			X	X						X	X	1					
8			X	X						X	X		1				End Init
9	0		Hi	Lo						Lo	Lo						----- Test
10	1		Lo	Lo						Lo	Lo						Hi,Lo
11			Lo	Hi		0				Lo	Lo						are outputs
12			Lo	Lo		1				Lo	Lo						
13			Lo	Lo				0		Hi	Lo						
14			Lo	Lo				1		Lo	Lo						
15			Lo	Lo						Lo	Hi	0					
16			Lo	Lo						Lo	Lo	1					

FIG. 2—PROGRAMMING SHEET. Create a test program using a form like this for each IC you wish to test.

Tester programs

Four programs control the tester: CHNGADR, ICSETUP, IC, and ICAUTO. Starting with the simplest, CHNGADR is a short program that reads the file HWADDRES.DAT, displays it on the screen, and then asks you for a new hardware address. It stores your answer in a four-character data file (three characters for the value of the address, and a trailing "D" to indicate that the address is in decimal).

ICSETUP allows you to identify the number of pins on the IC under test, specify each pin as an input, output, ground, or +V, and label each pin. Then ICSETUP counts the number of inputs and outputs to determine the 8255's mode (PA = In, PB = Out, PC = In; or PA = Out, PB = In, PC = Out). Then ICSETUP assigns the available input and output lines to appropriate pins on the ZIF in the test fixture, and displays an interconnect list, like that shown in Table 1, which specifies which connections from the solderless breadboard go to which ZIF pins.

The program then writes your definitions and the 8255 setup information to a definition file. For example, if you specify file "4011," ICSETUP creates a file named 4011.DEF. You have to run ICSETUP only once for each type of IC; a library of definitions for common IC's is available from the author.

The program called IC uses the definition-file information to allow you to perform manual testing. You use the manual procedure to create a test file that ICAUTO can use to do all testing

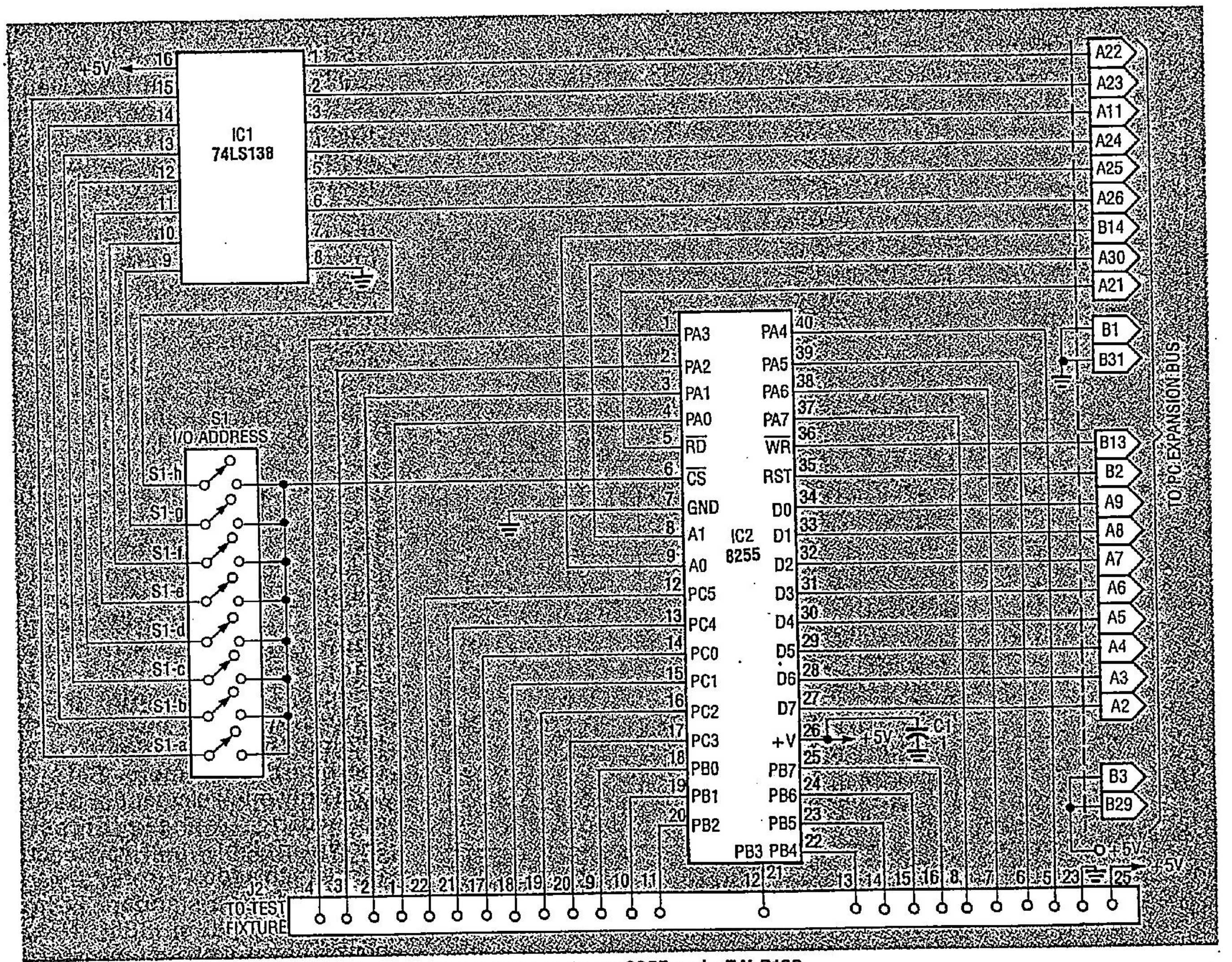


FIG. 3—THE I/O BOARD CONTAINS only two active components, an 8255 and a 74LS138. A bypass capacitor and a DIP switch round out the circuitry.

automatically. IC asks you for the name of the file, retrieves the definition information, displays an interconnect list, and then displays a graphical view of the IC under test.

Using the interconnect list, you make the connections and then insert the IC to be tested into the ZIF socket. Next you enter a pin number and a value (0 or 1). (For IC's that require a clock signal, you can specify a pulse by typing P instead of 0 or 1. A pulse reverses the state of a signal momentarily and then returns it to the original value. Only the IC's response at the end of the pulse will be stored as relevant.) The program outputs the specified value to the specified pin, reads the IC's outputs, saves the results, and displays the states of all pins on the screen.

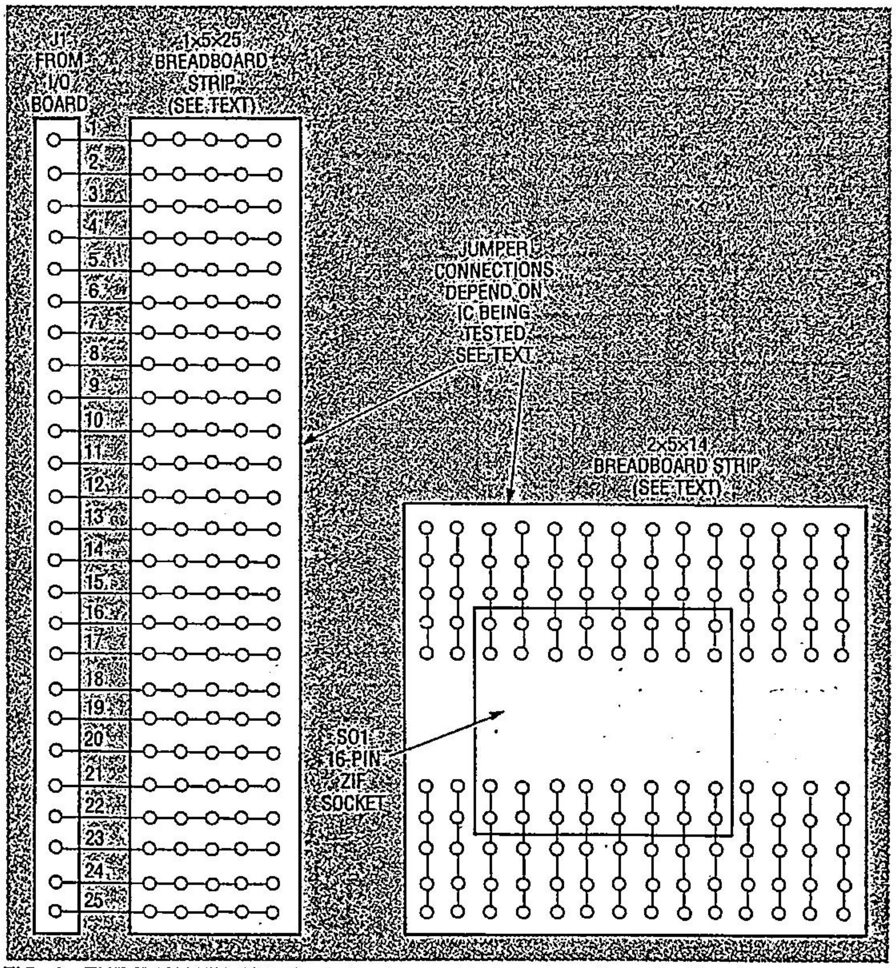
Actually, the IC program has two phases, Initialize and Test. During Initialize, all outputs are stored as "256," which indicates that the results are not to be considered in determining whether an IC is good or bad. When you finish initialization, you select the Test phase. Now both inputs and outputs are stored as they occur. When you finish the Test phase, you select End, at which point you can save the complete test procedure, along with the acceptable results, to a data file if you so desire. Examples of the data structure for the DEF and DAT files are shown in Tables 2 and 3, respectively.

The final program, ICAUTO, uses the DEF and DAT files to test an IC automatically. If a response is not what is specified in the DAT file, the program stops, indicates a failure, and identifies the pin where the failure occurred. Otherwise, after completing all the sequences in the DAT file, ICAUTO indicates Pass and requests the next IC.

Construction

The tester hardware consists of two major components: the PC-based I/O board and the test fixture. The two communicate via a standard ribbon cable with 25-pin male D connectors at both ends.

We'll begin with the I/O Board, which can be built with a prototyping PC "half-card," such as Jameco's JE417. This type of card contains an array of plated-through holes on a 0.1" grid, a mounting area for a DB-25 connector, and an edge-contact area suitable for an 8-bit PC expansion slot. Before beginning assembly, place the card in front of you with the DB-25 mounting area to your left, and the edge-contact area down. Note the 31 gold-plated edge contacts: the



J1 FROM I/O BOARD
1×5×25 BREADBOARD STRIP (SEE TEXT)
JUMPER CONNECTIONS DEPEND ON IC BEING TESTED SEE TEXT
2×5×14 BREADBOARD STRIP (SEE TEXT)
SO1 16-PIN ZIF SOCKET
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

FIG. 4—THE TEST FIXTURE includes two solderless breadboards. One has permanent connections from J1; the ZIF socket is inserted into the other. The software specifies how the two must be interconnected, depending on the type of IC to be tested.

LISTING 1

```
1 REM** I/OTEST.BAS
10 CLS : PRINT : PRINT : PRINT : DEF SEG = 64
20 INPUT "Address You've Selected (ex: 640)";A
15 B = A+1: C = B+1: D = C+1
30 OUT D, 130 : REM** PA=Output, PB=Input, PC=Output
40 OUT A, 255 : X1 = INP(A)
50 OUT A, 0 : X2 = INP(A)
60 IF (X1 <> 255) OR (X2 <> 0) THEN PRINT "Problem With PA"
70 OUT C, 255 : X1 = INP(C)
80 OUT C, 0 : X2 = INP(C)
90 IF (X1 <> 255) OR (X2 <> 0) THEN PRINT "Problem With PC"
100 OUT D, 153 : REM** PA=Input, PB=Output, PC=Input
110 OUT B, 255 : X1 = INP(B)
120 OUT B, 0 : X2 = INP(B)
130 IF (X1 <> 255) OR (X2 <> 0) THEN PRINT "Problem With PB"
135 GOTO 150
140 PRINT : PRINT TAB(25);"I/O BOARD TESTS OK AT ADDRESS";A
150 PRINT : PRINT : PRINT : END
```

LISTING 2

```
200 REM** This code checks for a high input on PB5 (J2-14)
210 REM** It assumes you've set up the 8255 by creating
215 REM** variables A,B,C,D and have performed an
220 REM** OUT D, 130 to set PB as an input register.
230 REM
240 IF (INP(B) AND 8)<>8 THEN X=X+1 : GOTO 260
250 PRINT "High Sensed On PB5." : END
260 IF X < 500 THEN GOTO 240 : REM** Try 500 times!
270 PRINT "PB5 Remained Low During 500 Sensings." : END
```

left-most one is B1. Find contacts B5 through B10, and, using a hobby knife, remove them. Doing so is a safety measure to ensure that the +12, −12, and −5-volt supplies from the PC do not get into our +5-volt-only circuitry.

Next, flip the card over so that the DB-25 area is to your right. Install a low-profile 40-pin socket vertically about an inch from the DB-25 mounting area, as shown in the opening photos of this article, and solder it in place. Locate a 16-pin low-profile socket horizontally about an inch above the edge contacts; the left end of the socket should be above edge-contact A31. Solder the socket to the board. Then mount S1 horizontally above the 16-pin socket and solder in place.

Using 22-gauge wire or smaller, make all interconnections, as shown in Fig. 2. Check for shorts, opens, and solder bridges. Then mount the DB-25 connector. Now insert IC1 and IC2 into their sockets, and check the board once more. Set one (and only one!) element of S1 on, according to the addresses shown in Table 4. If you're not sure which address to use, try address 544 for a standard PC, or 640 for an AT or 386). Next, power down your computer, insert the card in a free slot, and power back up. *Never fiddle with the board while the PC is on.*

Enter and run I/OTEST.BAS (shown in Listing 1) using GWBASIC or BASICA. Be sure to enter the correct address when the program prompts you. If you get the message "I/O BOARD TESTS OK AT ADDRESS XXX" (where "XXX" is the address you specified with S1), you're ready to proceed. Otherwise, power down the PC, set S1 to another address, and try again.

The test fixture consists of a plastic box (any size or type will do), a DB-25 female connector, two chunks of a solderless breadboard, and a 16-pin ZIF socket for test IC's. See the opening photos of this article.

The solderless breadboard should have at least 42 rows of contacts and no power buses. Starting from one end, count down 26 rows and carefully cut the breadboard with a fine-tooth hack saw. File the end of the piece you just cut so that you end up with a piece with 25 clean rows of contacts.

Typical breadboards have two unconnected columns of contacts separated by a "valley." Cut along the valley to create a piece with just a single column of 25 rows of five contacts. This is the block into which the 24 wires from J1 will be inserted.

From the remaining piece (with the valley), cut a section that contains 22 rows. Do not cut down the valley, because this piece will accept the ZIF socket. File the ends smooth.

Now machine the box to accept J1 and the breadboards, and drill some holes on top to pass the J1 wires. Solder 6" lengths of #22 solid wire to J1 pins 1 through 23, and pin 25. Mount J1 in the box. Locate the J1 connecting block (the one without the valley) on the box and secure it with glue or double-sided tape. Beginning with pin 1, feed each wire up through the hole and insert in row 1. Continue through pin 23. Last, feed the wire from J1, pin 25 and insert it in row 25. Note: Nothing is connected to row 24. Check continuity between J1 and the block to ensure you've connected all the leads correctly. Locate the other block (with the valley) perpendicular to the J1 block and secure it to the box. Insert the ZIF into the block. Mark the block to indicate which row is pin 1, which is pin 2, etc. Cut sixteen 6" lengths of #22 solid wire and strip ⅜" of insulation from each end of each wire. These will serve to connect the J1 block to the ZIF block.

Use

At this point you should have a board (with verified address) and a HWADDRES.DAT file indicating that address. You should also have a correctly-wired test fixture, and a cable to connect it to the board.

To create a data file for an IC, run ICSETUP. Identify the IC as either 14 or 16 pins. Then define each pin as an output or an input, and give each pin a name (six-character maximum length). Don't define the power pins yet. When finished, enter "99." Now you'll be asked to identify the GND and +V pins. When you're done, you'll be asked if you want to save this definition to a file.

TABLE 1
SAMPLE INTERCONNECT LIST

SO1/ZIF Socket	J1/Breadboard
1	1
2	2
3	9
4	10
5	3
6	4
7	23
8	
9	
10	5
11	6
12	11
13	12
14	7
15	8
16	25

TABLE 2—4011 DEF FILE

Rec. No.	Data	Decoded Mask
1	CD 4011, Quad	
2	2-In NAND Gate,	
3	CMOS	
4	Decoded	
5	Mask	
6	130	
7	OUT1 IN-A A**	1
8	OUT2 IN-A A**	2
9	OUT3 IN-A B**	1
10	OUT4 OUT-A B**	2
11	OUT5 OUT-B A**	4
12	OUT6 IN-B A**	8
13	N/A7 IN-B **	0
14	**	0
15	**	0
16	OUT 8IN-C A**	16
17	OUT 9IN-C A**	32
18	OUT10OUT-C B**	4
19	OUT11OUT-D B**	8
20	OUT12IN-D A**	64
21	OUT13IN-D A**	128
22	N/A14V+ **	

** Note: Encoded mask value

Answer yes and enter a descriptive name (the number of the IC, maximum eight characters), and don't include an extension.

Now get a clean programming sheet and create a test program for that IC. With the programming sheet in hand, run IC. Identify the previously specified file name. Connect the J1 and ZIF test blocks according to the interconnect list that appears on the screen. Insert a known good IC and press a key.

Execute the Initialize and Test procedures as outlined above.

TABLE 3—4011 DAT FILE

Rec. No.	PortA	PortB	PortC
1	1	256	0
2	3	256	0
3	7	256	0
4	15	256	0
5	31	256	0
6	63	256	0
7	127	256	0
8	255	256	0
9	254	1	0
10	250	3	0
11	234	7	0
12	170	15	0
13	171	14	0
14	175	12	0
15	191	8	0
16	255	0	0
17	253	1	0
18	245	3	0
19	213	7	0
20	85	15	0
21	87	14	0
22	95	12	0
23	127	8	0
24	255	0	0

TABLE 4—I/O ADDRESS

Switch Pos.	Address
S1-a	512
S1-b	544
S1-c	576
S1-d	608
S1-e	640
S1-f	672
S1-g	704
S1-h	736

When done, select End, and in response to the Save question, specify Yes. You now have the files necessary to test IC's of this type.

To test an IC, run ICAUTO. Specify a file name, connect the test blocks, insert the IC to be tested, and press a key.

As the IC is tested, you will hear a short tone as each step is performed. The bottom of the screen will report the relevant information, and the graphical representation of the IC on the screen will show how the inputs and outputs are changing. If the IC fails, the test will stop and the pin where failure occurred will be highlighted. Otherwise, the test will end and "Pass" will be indicated. Insert the next IC to be tested and press any key to continue. After testing all IC's of that type press Esc to end. R-E