DIGITAL IC TESTER

An Accessory For Your Commodore 64

Your Commodore C64 computer can pretest digital components.

JIM BARBARELLO

An important part of building a digital project is to make sure that all the active components work properly before they're installed. That is even more critical when you deal with sensitive devices like CMOS integrated circuits. Unfortunately, few of us do pre-installation tests because it's somewhat difficult to routinely test digital devices of any kind. We simply solder the parts into the circuit, cross our fingers, and apply power. If the circuit doesn't work, we poke around until we find the bad part. Then we replace it, hoping we don't damage the PC board, or install another defective component. Clearly, a better way is to test each part (especially IC's) before using it.

If you're fortunate enough to own a Commodore *C64* computer you can easily and inexpensively assemble a special kind of tester that will automatically perform up to 100 user-programmed tests on most TTL and CMOS 14- and 16-pin digital IC's; and it is also a simple matter to use the tester to check the switching action of transistors and diodes.

The tester, whose schematic is shown in Fig. 1, is a simple device that actually depends on special software to run the tests. That software allows you to specify how you want a specific TTL or CMOS device tested, and then tests as many units of that device as you like, one after the other, with each complete test taking less than five seconds. If the device doesn't function properly, the software stops the test and tells you specifically how the device failed.

To avoid creating damage where none existed before, the DUT (*D*evice *U*nder *T*est) is installed in a ZIF (*Z*ero *I*nsertion *F*orce) socket; a special kind of IC socket that applies virtually no strain or force to the pins of the DUT.

As an added bonus, the tester includes a hardware reset switch (S1) that resets the computer at the push of a button (no more powering down and then up again). No external power supply is required to use the unit because the tester gets its power directly from the computer's expansion port.

The circuit

As shown in Fig. 1, the tester consists mainly of IC1, a 6821 PIA (*P*eripheral *I*nterface *A*daptor), two resistors, and ZIF-socket SO3. In addition, two 16-pin DIP sockets (SO1 and SO2) are used as terminal blocks to apply power to the DUT. Switch S1 connects between the computer's RESET line and ground: pressing S1 brings the RESET line low and restarts the system without actually having to re-power the computer.

PIA IC1 receives its power, timing, addressing, and data from the computer's expansion port. The sixteen individual peripheral lines are programmed as inputs or outputs by software. IC1 is addressed at decimal memory locations 57100 to 57103. Output lines remain in the previously set state until changed. Input lines are not latched, and therefore show the momentary status of any device connected to them.

Two halves

Since the computer has an 8-bit data bus and the PIA has 16 data lines, the PIA is divided into two halves: side A with PA0–PA7, and side B with PB0–PB7. There are individual control, data direction, and peripheral registers for each side. The control registers allow you to select either the data-direction register (for configuring the peripheral lines as input or output), or the peripheral register (for sending and receiving data). The peripheral registers (when selected as data- direction registers) are located at decimal memory locations 57100 (side A) and 57102 (side B). The control registers are located at 57101 (side A) and 57103 (side B).

Construction

The tester is assembled on a Jameco *JE413* or similar wiring board using wire-wrap connections. Because the edge of the board also serves as plug P1, Fig. 1 shows the connections to both sides of P1 as they actually appear on the board. The half of P1 indicated as TOP are the gold-plated finger connections on the component side of the

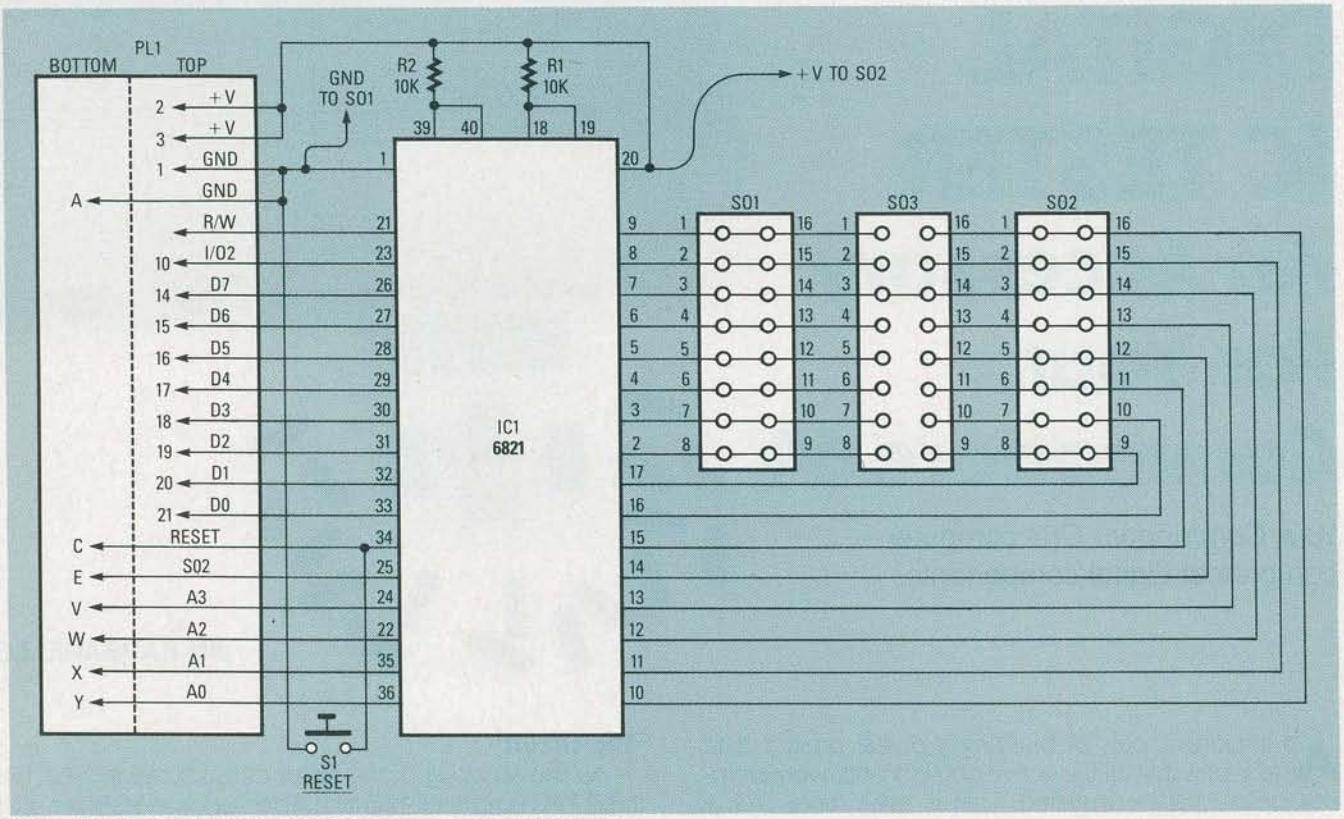

FIG.1—THE SOCKET TERMINALS are specially wired to provide almost automatic selection of the ground and power terminals.

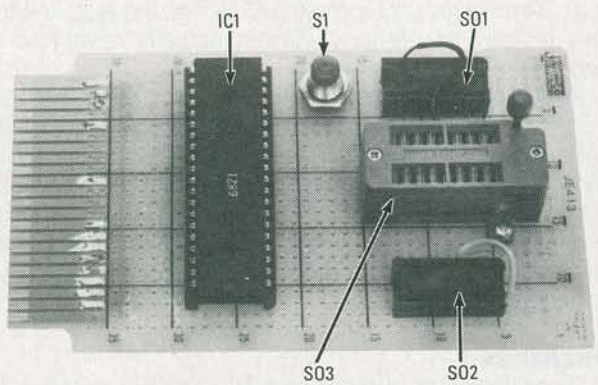

FIG. 2—THE PARTS WILL FIT WITHOUT CROWDING on a Jameco *JE413* board. Wire-wrap is used for all connections except those soldered to the fingers.

board. The half of P1 indicated as BOTTOM are the gold-plated finger connections on the wiring side of the board.

However, the connections to both sides of the board are made from the bottom side. The bottom connections have obvious solder pads that extend from the fingers. The connections to the top fingers are made by passing wires through the appropriate holes on the board. Although wire-wrap connections are used to interconnect the components, those to the fingers should be soldered; don't use stake-on or wire-wrap terminals at the fingers.

The general parts layout isn't critical, although operation will be somewhat more convenient if you follow the one shown in Fig. 2. Make certain you install SO3 so that its lever faces the back edge of the board. Drill two ⅛-inch holes near SO1 and SO2 for the power leads.

Wire across

As shown in Fig. 1, the pins on SO1 and SO2 are wired straight across: pin 1 connects to pin 16, pin 2 connects to pin 15, etc. That is done so you don't have to figure out the convolution of the pins when inserting the +V and ground wires in SO1 and SO2. To determine the power connections to the DUT, simply count from 0–14 or 0–16 starting on either row of pins and you'll automatically end up on the correct connections. (It might appear confusing

PARTS LIST

R1, R2—10,000-ohms, ¼-watt, 10%
IC1—6821 PIA (Jameco 68B21)
SO1, SO2—16 pin wire-wrap socket
SO3—16 Pin ZIF socket
S1—Momentary N.O. pushbutton switch
Miscellaneous: Wiring board Jameco JE413 , No. 24, solid wire (black and red), wire-wrap wire, wire-wrap tool, solder, etc.

NOTE: The following are available from J. J. Barbarello, RD#3, Box 241H, Tennent Road, Manalapan, NJ 07726. IC Tester program (No. IC64) on Commodore *C64* disk (IC64): $5.00. Enhanced program (No. (TTLDB) with TTL library of the most commonly used TTL IC's on a Commodore *C64* disk, along with instructions for use: $20.00. Enhanced program (No. CMOSDB) with CMOS library of the most commonly used CMOS IC's on disk, along with instructions for use (CMOSDB): $20.00. Enhanced program with both CMOS and TTL libraries and instructions: $30.00. Please include $2.00 shipping with each order. New Jersey residents must add appropriate sales tax.

```
1 REM ************************************
2 REM **  IC64 TESTING SOFTWARE       **
3 REM **  NAME:  IC64    V87Ø6Ø5      **
4 REM **  (C) 1987, J.J. BARBARELLO  ** 5 REM **  MANALAPAN, NJ  Ø7726       **
6 REM ************************************
8 GOSUB 5ØØØ:PRINTCHR$(147):RC$=CHR$(18)+CHR$(158):NC$=CHR$(146)+CHR$(5)
1Ø DIM I1(1ØØ),I2(1ØØ),O1(1ØØ),O2(1ØØ),DO$(16),LS$(16):RC$=CHR$(18)+CHR$(158)
15 PRINTCHR$(147):RO=4:CO=4:FORI=1TO8:T(I)=2^(I-1):NEXT
16 T1$="    "+RC$+"  THE IC64 INTEGRATED CIRCUIT  "+NC$
17 T2$="    "+RC$+"      AUTOMATED TEST SYSTEM     "+NC$:PRINTT1$:PRINTT2$
18 GOSUB85Ø5Ø:PRINTBL$:GOSUB5Ø5Ø:INPUT"WHICH SIZE IC (14 OR 16 PINS)";IC
19 IF IC<>14 AND IC<>16 THEN 18
2Ø PRINT"    WAIT...":CM$="+-NIDE"+CHR$(29)+CHR$(157)
25 L9$="SETUP PHASE: DEFINE SOCKET PINS   "
3Ø IFIC=16THENL1$="    Ø Ø Ø Ø Ø Ø Ø Ø Ø 1 1 1 1 1 1 1 1    ":GOTO 4Ø
35 L1$="    Ø Ø Ø Ø Ø Ø Ø     Ø Ø 1 1 1 1 1 1    ":GOTO 4Ø
4Ø IFIC=16THENL2$="    1 2 3 4 5 6 7 8 9 Ø 1 2 3 4 5 6    ":GOTO 5Ø
45 L2$="    1 2 3 4 5 6 7 N N 8 9 Ø 1 2 3 4    "
5Ø L3$="    C C C C C C C C C C C C C C C C    "
6Ø GOSUB85Ø5Ø:PRINTL0$:PRINTL1$:PRINTL2$:PRINTL3$
7Ø RO=16:CO=4:GOSUB5Ø5Ø:PRINT RC$;" SETUP COMMANDS: ";NC$:PRINT
8Ø PRINT"    I = INPUT PIN";TAB(22);"O = OUTPUT PIN"
9Ø PRINT"    ";"- = VSS (GND)"TAB(22);"+ = VDD (PWR)
1ØØ PRINT"    N = NOT USED"TAB(22)"E = END DEFINE":PRINT"    "CHR$(6Ø)"-";
11Ø PRINT"CRSR = LEFT"TAB(22)"-"CHR$(62)"CRSR = RIGHT"
12Ø RO=9:CO=4:GOSUB 5Ø5Ø:SS$=CM$:PT=1
13Ø GOSUB 61ØØ:GOSUB6ØØØ
14Ø ON SS GOSUB 15Ø,16Ø,17Ø,18Ø,19Ø,23Ø,21Ø,22Ø:GOSUB5Ø5Ø:GOTO13Ø
15Ø PRINT"+":DO$(PT)="+":RETURN
16Ø PRINT"-":DO$(PT)="-":RETURN
17Ø PRINT"N":DO$(PT)="N":RETURN
18Ø PRINT"I":DO$(PT)="I":RETURN
19Ø PRINT"O":DO$(PT)="O":RETURN
21Ø IF PT=16 THEN RETURN
212 IF IC=14 AND PT=7 THEN PT=9:CO=CO+4
215 PRINTCHR$(157);"{5S}":PT=PT+1:CO=CO+2:RETURN
22Ø IF PT=1 THEN RETURN
222 IF IC=14 AND PT=1Ø THEN PT=8:CO=CO-4
225 PRINTCHR$(157);"{9S}":PT=PT-1:CO=CO-2:RETURN
23Ø PRINTCHR$(157)"{5S}"
235 RO=1Ø:CO=14:GOSUB5Ø5Ø:PRINTRC$" W A I T ..."NC$
236 FORI=1TO16
237 IFDO$(I)<>"I"ANDDO$(I)<>"O"ANDDO$(I)<>"+"ANDDO$(I)<>"-"THENDO$(I)="N"
238 NEXT
24Ø FORI=1 TO 8:IF DO$(I)="I" THEN PA=PA+T(I)
245 IF DO$(I)="N" THEN MA=MA+T(I)
25Ø NEXT:MA=255-MA
26Ø FORI=1 TO 8:IF DO$(I+8)="I" THEN PB=PB+T(I)
265 IF DO$(I+8)="N" THEN MB=MB+T(I)
27Ø NEXT:MD=255-MB
28Ø POKE SA+1,Ø:POKE SA,PA:POKE SA+1,4:POKE SB+1,Ø:POKE SB,PB:POKE SB+1,4
285 POKE SA,Ø:POKE SB,Ø
29Ø RO=4:CO=4:GOSUB5Ø5Ø:PRINT"    I.C.  INITIALIZATION          "
3ØØ RO=16:CO=2:GOSUB5Ø5Ø:PRINT"DIRECTIONS:  MOVE CURSOR WITH LEFT/"
31Ø PRINT"    RIGHT CURSOR KEY.  CHANGE INPUTS    "
32Ø PRINT"    BY TYPING 1 OR Ø.  BRING IC TO      "
33Ø PRINT"    INITIALIZED STATE, THEN PRESS ";RC$;" E ";NC$:PRINTBL$:PRINTBL$
34Ø PRINT"    STEP: Ø"TAB(22)"(MAX STEPS=1ØØ)":NS=1:PH=Ø
345 RO=9:CO=4:GOSUB5Ø5Ø:FORI=1TO16:PRINTDO$(I)"{5S}";:NEXT:PRINT:PRINT"    ";
35Ø FOR I=1TO16:IF DO$(I)="I" THEN PRINT"Z ";:PU=I:GOTO 37Ø
36Ø PRINT"{5S}{5B}";
37Ø NEXT:CM$="Ø1E"+CHR$(29)+CHR$(157):RO=11:CO=3:GOSUB5Ø5Ø:PH=Ø
38Ø SS$=CM$:PT=1:N=PEEK(SA) AND MA:GOSUB3ØØØ:PRINTA$;:N=PEEK(SB) AND MB
385 GOSUB 3ØØØ:PRINTA$
39Ø FORPT=1TO16:IFDO$(PT)<>"I"THEN NEXT:GOTO 7ØØØ
4ØØ CO=4+(PT-1)*2:GOSUB 5Ø5Ø:PX=PT
41Ø GOSUB61ØØ:GOSUB6ØØØ:IF SS=Ø THEN 41Ø
42Ø ON SS GOSUB 43Ø,44Ø,53Ø,45Ø,46Ø:GOSUB5Ø5Ø:GOTO41Ø
43Ø PRINT"Ø";:LS$(PT)="Ø":GOTO 47Ø
44Ø PRINT"1";:LS$(PT)="1":GOTO 47Ø
45Ø IF PT=PUTHENRETURN
453 PT=PT+1:IFDO$(PT)<>"I" THEN 45Ø
456 PRINTCHR$(157);"{5S}":CO=4+(PT-1)*2:RETURN
46Ø IFPT=PXTHENRETURN
463 PT=PT-1:IFDO$(PT)<>"I"THEN46Ø
466 PRINTCHR$(157);"{5S}":CO=4+(PT-1)*2:RETURN
47Ø PRINTCHR$(157);CHR$(157);RC$">"NC$:C=CO
475 N1=Ø:FORI=1TO8:IFLS$(I)="1"THENN1=N1+T(I)
48Ø NEXT:N2=Ø:FORI=9TO16:IFLS$(I)="1"THENN2=N2+T(I-8)
49Ø NEXT:O1(NS)=N1:O2(NS)=N2:POKE SA,N1:POKE SB,N2:RO=11:CO=3
495 IF PH=Ø THEN I1(NS)=999:I2(NS)=999
5ØØ IF PH=1 THEN I1(NS)=PEEK(SA) AND MA:I2(NS)=PEEK(SB) AND MB
5Ø5 GOSUB5Ø5Ø:N=PEEK(SA) AND MA:GOSUB3ØØØ:PRINTA$;:N=PEEK(SB) AND MB
5Ø7 GOSUB 3ØØØ:PRINTA$
51Ø IF NS=1ØØ THEN SS$="YN":GOTO 57Ø
52Ø RO=22:CO=9:GOSUB5Ø5Ø:PRINTNS:RO=11:CO=C:GOSUB5Ø5Ø:NS=NS+1:RETURN
53Ø IF PH=1 THEN SS$="YN":NS=NS-1:GOTO 57Ø
535 PH=1:C=CO:RO=4:CO=4:GOSUB5Ø5Ø:PRINT"       I.C. TESTING PHASE
54Ø RO=15:CO=Ø:GOSUB65Ø5Ø:PRINTRC$;BL$;NC$:RO=18:CO=23:GOSUB5Ø5Ø
55Ø PRINT"PRESS "RC$" E "NC$" TO END ";RC$;BL$;NC$
56Ø RO=11:CO=C:RETURN
57Ø PRINTCHR$(147):PRINTT1$:PRINTT2$:PRINT:PRINT"    TEST ANOTHER IC (Y/N)..";
58Ø GET A$:IF A$="" THEN 58Ø
59Ø GOSUB 6ØØØ:IF SS=Ø THENS8Ø
6ØØ IF SS=2 THEN 9ØØ
61Ø RO=4:CO=4:GOSUB5Ø5Ø:PRINT"     <<< TESTING PHASE >>>"
62Ø PRINT:PRINT"   INSERT NEXT DEVICE TO BE TESTED."
63Ø PRINT:PRINT"   PRESS "RC$" F1 "NC$" WHEN READY...";
64Ø GET A$:IF A$="" THEN 64Ø
65Ø IF ASC(A$)<133 THEN 64Ø
66Ø RO=4:CO=4:GOSUB5Ø5Ø:PRINTBL$;BL$;BL$;BL$:RO=4:CO=4:GOSUB5Ø5Ø
67Ø PRINT"TESTING. STEP:";CO=19
68Ø FORI=1TO NS:IF I1(I)<>999 THEN 69Ø
685 POKE SA,O1(I):POKE SB,O2(I):GOSUB 5Ø5Ø:PRINTI:NEXT
69Ø S=I:FORI=S TO NS:GOSUB5Ø5Ø:PRINTI:POKE SA,O1(I):POKE SB,O2(I)
7ØØ IF(PEEK(SA) AND MA)<>I1(I) OR(PEEK(SB) AND MB)<>I2(I) THEN 73Ø
71Ø NEXT
72Ø PRINT:PRINT"    PASS. IC PERFORMS AS EXPECTED"
723 PRINT"    PRESS ANY KEY..."
725 GET A$:IF A$="" THEN 725
726 GOTO 57Ø
73Ø CO=4:GOSUB5Ø5Ø:F=I:PRINT"       IC FAILS AT STEP";F
735 PRINT"       PIN #{5S}(I=INPUT, O=OUTPUT)"
74Ø PRINTL1$:PRINTL2$:PRINTL3$:PRINT"    ";:FOR Q=1TO16:PRINTDO$(Q)"{5S}";:NEXT:PRINT
75Ø PRINT:PRINT"          EXPECTED RESPONSE"
76Ø N=I1(F):GOSUB 3ØØØ:PRINT"   ";A$;:N=I2(F):GOSUB 3ØØØ:PRINT A$
765 IF IC=14 THEN RO=14:CO=18:GOSUB5Ø5Ø:PRINT"   "
77Ø PRINT:PRINT"          OBSERVED RESPONSE"
78Ø N=PEEK(SA) AND MA:GOSUB 3ØØØ:PRINT"   ";A$;:N=PEEK(SB) AND MB
79Ø GOSUB3ØØØ:PRINTA$:PRINT:IF IC=14 THENRO=17:CO=18:GOSUB5Ø5Ø:PRINT"   ":PRINT
795 GOTO 723
9ØØ PRINTA$:RO=12:CO=12:GOSUB5Ø5Ø:PRINT"PROGRAM ENDED.":PRINT:PRINT:END
2999 END
3ØØØ REM** CONVERT TO BASE2
3Ø1Ø A$="":FOR I=8 TO 1STEP-1:IF T(I)>N THENA$=" Ø"+A$:GOTO 3Ø3Ø
3Ø2Ø N=N-T(I):A$=" 1"+A$
3Ø3Ø NEXT:RETURN
5ØØØ REM* CURSOR CONTROL USING PLOT              KERNEL ($FFFØ)
5ØØ5 SA=571ØØ:SB=SA+2:POKE SA+1,Ø:POKE SA,Ø:POKE SA+1,4:POKE SB+1,Ø:POKE SB,Ø
5ØØ8 POKE SB+1,4
5Ø1Ø DATA 162,Ø,16Ø,Ø,24,32,24Ø,255,96,999
5Ø2Ø A=49З99:SC=A
5Ø3Ø READ B:IF B<>999 THEN POKE A,B:A=A+1:GOTO 5Ø3Ø
5Ø4Ø POKE53280,6:POKE53281,6:RETURN
5Ø5Ø POKE SC+3,COL:POKE SC+1,ROW:SYS SC
5Ø6Ø BL$="                                        ":RETURN
6ØØØ REM** INSTR ,SEARCH=SS$,IN=A$
6Ø1Ø FOR SS=1 TO LEN(SS$)
6Ø2Ø IF A$=MID$(SS$,SS,1) THEN RETURN
6Ø3Ø NEXT SS:SS=Ø:RETURN
61ØØ REM** CURSOR BLINK
6111 PRINTCHR$(157)">";
6120 GET A$:IF A$="" THEN 614Ø
6130 RETURN
614Ø PRINTCHR$(157);" ";:GOTO 6110
7ØØØ REM** NO DEFINITIONS
7Ø1Ø PRINTCHR$(147):PRINT:PRINT
7ØZØ PRINT"  NO DEFINITIONS MADE. PROGRAM ABORTED":PRINT:PRINT:END

READY.
```

at first glance, but if you study Fig. 1 a few times you'll understand how it works out automatically.)

Connect R1 and R2 after all the wire-wrap connections are made. Solder R1 to IC1 pins 18 and 20. Similarly, solder R2 to pins 20 and 39. Cut 6-inch lengths of red and black No. 24 solid wire. Solder one end of the black wire to S1. Solder one end of the red wire to the junction of R1 and R2. Bring each wire through the board to the component side through drilled holes after first making a knot in each wire to serve as a mechanical stop. Make two additional stop-knots on the component side of the board, then strip ¼-inch of insulation from the end of each wire. The uninsulated ends will remain free for later use in applying power (red) and ground (black) to the DUT.

As with all wire-wrapped projects, check for wire nicks or other faults. You may want to do a continuity check on each connection to ensure proper wiring.

Using the tester

Insert the tester into the computer's expansion port before applying power to the computer. Enter the program shown in Table 1 (the program is available on disk—see the Parts List) and save it using the filename "IC64." Run the program by typing LOAD "IC64", 8 (notice that the comma is after the close-quotation mark), and then press the RETURN key. Then type RUN and press RETURN. Then install the DUT in the ZIF socket.

As shown in Fig. 3, 16-pin IC's just fit the ZIF socket. However, 14-pin IC's must be positioned towards the front of the socket because the rear two ZIF pins aren't used when testing 14-pin IC's. Take particular note from Fig. 3 how the pin numbering of the ZIF socket is modified for 14-pin IC's.

The program begins by asking you to specify whether the IC you'll be testing has 14 or 16 pins. Next, the program enters the Setup Phase and displays the screen shown in Fig. 4. Under the pin labeled 01 (meaning pin 1), is a blinking cursor that resembles the greater-than (>) symbol. (You can move the cursor with the right or left arrow keys). Note from Fig. 4 that when a 14-pin test is selected the two center positions are marked "N," meaning that the two lower pins in the ZIF socket (SO3) aren't used. The

cursor will automatically skip over the unused pins as the data is entered.

The bottom of the Fig. 4 screen defines the setup commands. As shown in Fig. 5, you identify each pin as an input (I) to the IC, an output (O) from the IC, a ground pin (−), or a voltage-supply pin (+). A pin that will not be used, or a pin where you do not particularly care what happens, is identified with an N or left blank. As an example, the pin identification for a 7408 TTL quad 2-input AND gate would be:

IIOII0− 0II0II+

For pins 1 through 7, pins 1 and 2 are identified as gate inputs with pin 3 being an output; pins 4 and 5 are inputs with pin 6 an output; pin 7 is ground (−). Similarly, we see that pins 8 and 11 are outputs, pin 14 is where the power

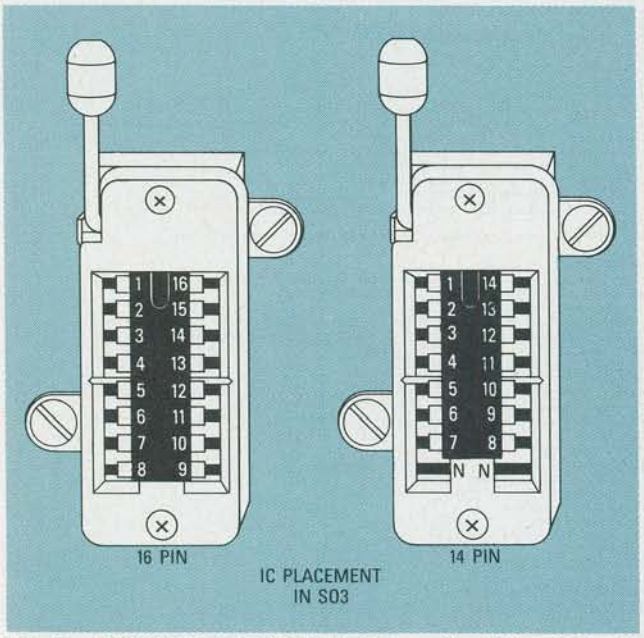

FIG. 3—ALTHOUGH A 16-PIN IC FITS the entire ZIF socket, a 14-pin IC must be positioned forward, so that the rear two ZIF pins aren't used.

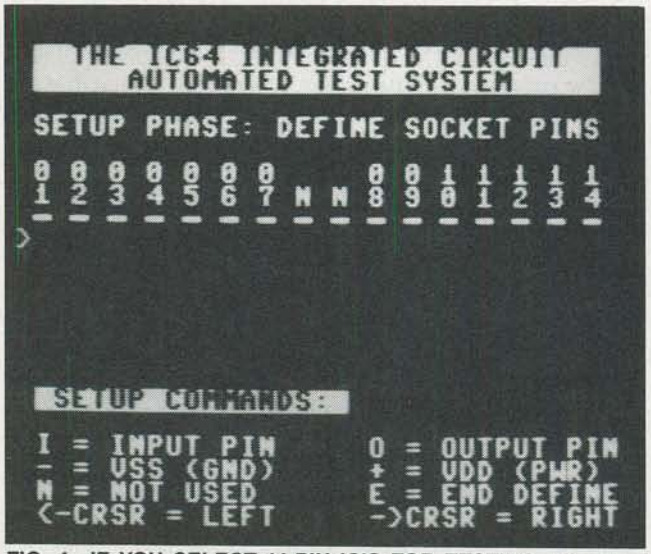

FIG. 4—IF YOU SELECT 14-PIN IC'S FOR TESTING the setup screen will indicate an N for the two unused ZIF socket terminals. The actual IC terminals are indicated as 01–14.

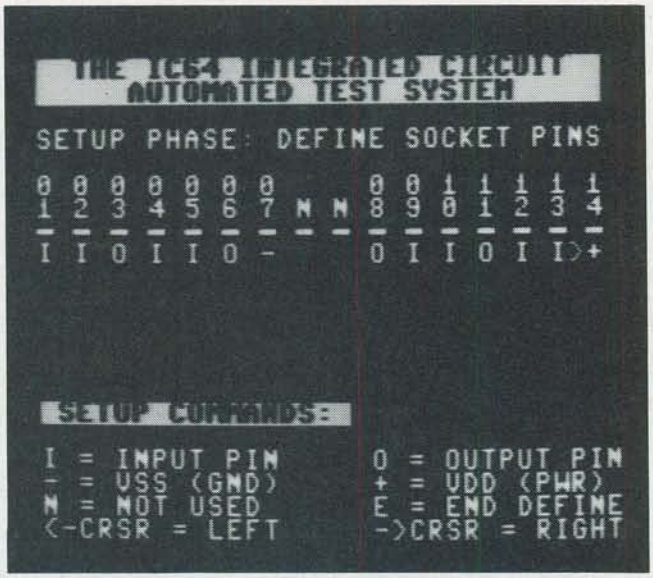

FIG. 5—FIRST STEP IN ANY TEST is to define the function of each socket pin, including the ground and +V connections.

FIG. 6—THE FIRST TEST INITIALIZES each IC to a standard setup. The N shown for pin 11 indicates we don't care what the result will be on that particular pin.

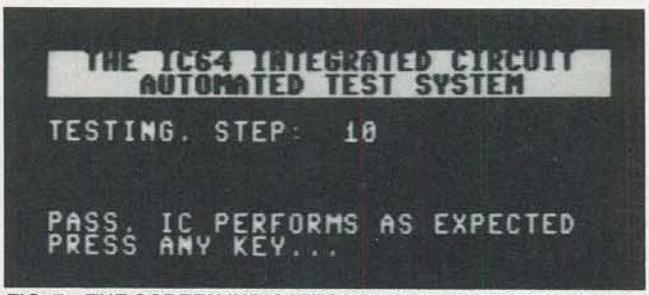

FIG. 7—THE SCREEN INDICATES WHEN the IC PASSES all tests that were established by the user.

(+V) is applied, and the remaining pins are inputs. The space between the − and the O indicate that the two lower pins in the ZIF socket are not used for the 14 pin device. To make any corrections, position the cursor and enter the desired data. Once the IC's I/O definition has been completed, press E to end the setup phase.

The next screen, IC Initialization, shows the definitions

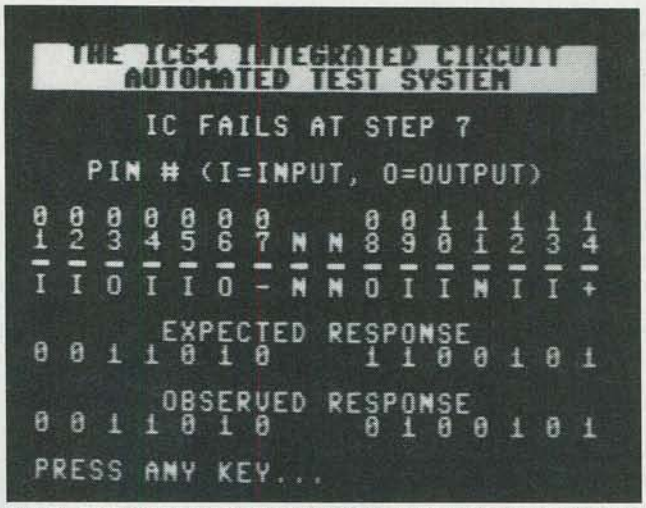

FIG. 8—THE SCREEN NOT ONLY INDICATES AN IC HAS FAILED, but it also shows what test (step) it failed and the and the manner in which it failed.

with each input pin marked with a diamond. During initialization the cursor will only move to the defined input pins. To illustrate the power of the tester, we will deliberately attempt to fool it by entering false data. As shown in Fig. 6, the definition for a NAND gate, we have defined output pin 11 with an N; meaning that we aren't interested in its result. It automatically displays a 0 even though its inputs, pins 12 and 13, should produce a high output. If you haven't done so yet, insert a known-good NAND GATE IC into the ZIF socket. Insert the black ground wire into position 7 or 10 of SO1 to apply ground to pin 7 of the DUT. Insert the red power wire into position 1 or 16 of SO2 to apply power to pin 14 of the DUT.

You want to initialize the IC to insure that all DUT's start out in the same state. As shown in Fig. 6, we'll change all the inputs to a low (0): Start by positioning the cursor at pin 1 and enter 0; then move the cursor to pin 2 and press 0. Since both inputs to the AND gate are now a 0, note that pin 3 changes to 1.

Continue the procedure, changing all inputs to a 0. It is important to note that you can initialize the IC in any way you want. For instance, you could have made pin 1 a 0, pin 2 a 1, pin 4 a 1, pin 5 a 0, and so on. The only rule for a valid test is to make sure your initialization values allow the test for each IC to begin in the same condition. But notice that pin 11, which we defined with an N, incorrectly indicates 0 because it's a "don't care" value.

When you press E to end the initialization phase, the title changes to the IC Testing Phase. As you did during the initialization phase, change the outputs one at a time to exercise all of the functions of the DUT. The number of test steps that you have defined are shown at the bottom left of the screen (you can define up to 100 steps, including initialization). As you change each input you will see its output change as per the IC's logic.

After performing all the steps necessary to exercise the IC, press E to end the testing phase. The program has now stored all conditions and responses of how you expect the IC to perform. The next screen, Auto Testing Phase, allows you to test as many IC's of this type as you desire. In answer to the question TEST ANOTHER DEVICE (Y/N)?, press the Y (Yes) key. As requested by the next message to appear, insert the next device to be tested into the ZIF socket and press any key to begin testing. The program will begin counting through the test steps. If the device is good, you'll see the screen display shown in Fig. 7. If the device does not perform as expected, the sequence will stop, and as shown in Fig. 8, the display will show what was expected and what was actually observed. You can then press a key to return to the auto-testing phase.

Additional capabilities

You can leave the tester connected when it's not being used. During that time, if you need to perform a complete (cold) reset of the computer, simply press the RESET button momentarily and then release it. As shown in Fig. 9, the computer will reset as if you had powered down and then up again.

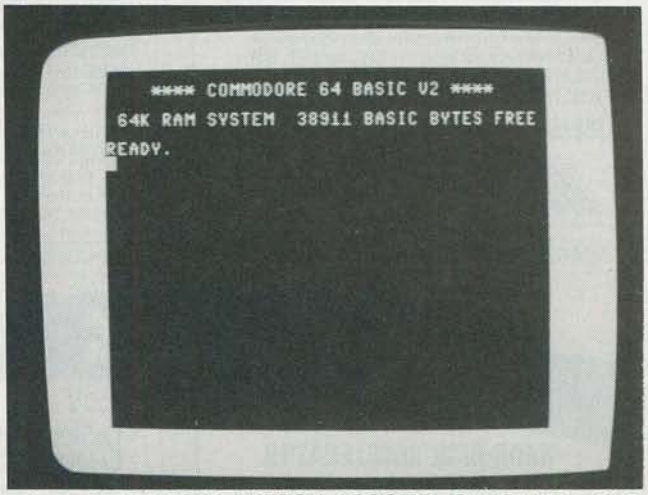

FIG. 9—THE RESET BUTTON ACCOMPLISHES THE SAME THING as if the computer was powered down up again. It makes a reset easier.

Since the tester can determine digital transitions of any device, you can also test the switching action of diodes and transistors. For example, place a 1,000-ohm resistor between pins 1 and 3, a 10,000-ohm resistor between pins 2 and the base of an NPN transistor, the collector of the transistor to pin 3, and the base of the transistor to pin 4. To accomplish that, you may wish to use a 16 pin DIP jumper cable with one end inserted into the ZIF socket and the other end attached to a solderless breadboard (Radio Shack parts 276-1976 and 276-175 respectively, or their equivalents). You can then make the connections on the breadboard.

Back on the tester, position the +V wire to pin 1 and the ground lead to pin 4. Using the program, define pin 1 as +, pin 2 as I (Input), pin 3 as O (Output) and pin 4 as − (ground). Initialize by changing pin 2 to 0. For the test, change pin 2 to 1, noting that pin 3 will change to 0. Change pin 2 back to 0 and note that pin 3 changes back to 1. That tests the digital switching action under a reasonable load and ensures the device will switch properly in a circuit. A similar setup can be used for a switching diodes such as the 1N914 and 1N4148.

You can also expand the program to allow saving of device setups and responses in a disk file, so you would not have to set them up manually each time. You could then build a library of files for devices. (A CMOS and a TTL library, along with an enhanced program is available. See the Parts List).