I C TESTER

TECHNICAL AND ASSEMBLY MANUAL

CIRCUIT CELLAR
4 Park Street, Suite 12 – Vernon, CT 06066

I C TESTER
Technical and Assembly Manual

Release 1.0
December 1987

Circuit Cellar Inc.
4 Park Street
Suite 12
Vernon, CT 06066

* *

WARRANTY

The Circuit Cellar Inc. extends the following warranty:

A factory manufactured circuit board or assembly carries with it a one year warranty covering both parts and labor. Any unit which is found to have a defect in materials or workmanship will be repaired or replaced at the option of Circuit Cellar Inc.

No credit will be given for units which show damage due to user modification or neglect.

Units returned for repair must have prior authorization from Circuit Cellar Inc. A return authorization number may be obtained by phone or letter. Please retain a record of the return authorization number as most subsequent correspondence will refer to that number. Under no circumstances is any product to be returned to the Circuit Cellar without prior authorization. Circuit Cellar Inc. will assume no responsibility for unauthorized returns.

All returns must be shipped prepaid. Insurance is recommended as losses by a shipping carrier are not the responsibility of The Circuit Cellar, Inc. Repaired units will be returned with postage paid.

For repair of units which have expired their warranty, a minimum inspection fee must be prepaid. Contact Circuit Cellar Inc. for information on current minimum charges.

NO WARRANTY is extended on USER ASSEMBLED systems or kits. However, assembled kits will be inspected and repaired with charges based on the current minimum one hour charge. However, in the event that repair charges would exceed a reasonable amount, the user may be consulted for a determination. Circuit Cellar Inc. retains the right to refuse to repair any USER ASSEMBLED item. This right is at the sole discretion of Circuit Cellar Inc.

Repairs on USER ASSEMBLED items must be prepaid.

Return authorization must be obtained prior to any return.

Circuit Cellar Inc. reserves the right to change any feature or specification at any time.

* *

Copyright Notice

Disclaimer

The Circuit Cellar Inc. makes no representations or warranties with respect to the contents hereof. Further, changes are periodically made to the information contained herein. The Circuit Cellar Inc. reserves the right to incorporate these changes in new editions of this publication without obligation to notify any person of such revisions or changes. Mention in this document of specific product(s) compatible with the FX-2MME does not constitute an en-dorsement of the product(s); rather the information regarding specific product(s) is given for illustrative purposes.

Trademarks

IBM, IBM PC, and IBM Personal Computer are trademarks of the International Business Machines Corporation. CCI is a copyright of Circuit Cellar Inc.

TABLE OF CONTENTS

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                                     *
*         NOTE: WHILE EVERY PRECAUTION HAS BEEN MADE TO               *
*         ASSURE DEVICE SAFETY, DEVICES INSERTED IN THE               *
*         CCI IC TESTER RUN THE RISK OF BEING DAMAGED.                *
*         DEVICES SHOULD ONLY BE TESTED IF SUCH A RISK                *
*         IS ACCEPTABLE.                                              *
*                                                                     *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

AN INTRODUCTION TO THE IC TESTER

The Circuit Cellar IC Tester is a device capable of testing many digital ICs to verify proper operation. The unit comes with an extensive built-in library supporting all common 74xx00-series devices, and can be customized by the user to support other devices; it can even test custom devices, such as programmable logic devices (e.g., PALs).

The tester includes a universal ZIF (Zero Insertion Force) socket, and can support 14-, 16-, 20-, and 24-pin devices with a variety of power/ground pinouts, and having package widths of 0.3-inch, 0.4-inch, or 0.6-inch. Devices from all standard logic families are supported, including 74, 74L, 74H, 74C, 74LS, 74ALS (Advanced Low-power Schottky), 74AS (Advanced Schottky), 74HC (High-speed CMOS), 74HCT (High-speed CMOS, TTL-compatible), 74AC (Advanced CMOS), 74ACT/74AHCT (Advanced CMOS, TTL-compatible), and 74F (Fairchild Advanced-Schottky). Many 4000-series CMOS devices are also supported, although special considerations apply. Note, however, that the new 74AC and 74ACT family -- having non-corner power and ground pinouts -- are not supported by this tester.

The tester supports three modes of operation: Stand-Alone Mode, Terminal Mode, and PC-Host Mode (in order of increasing flexibility). The reason the tester supports such a wide variety of operating modes is because each mode has its unique advantages and disadvantages. The variety of modes gives you maximum benefit from the tester, regardless of your particular requirements.

Stand-Alone Mode requires the optional two-line by 20-character liquid crystal display (LCD), but does not require any connection to the tester's RS-232C port. This mode can identify devices inserted in the ZIF socket, as well as "retest" other devices of the same type as the last device identified.

Terminal Mode operation does not require the optional LCD, but requires the RS-232C port to be connected to a dumb terminal or a computer capable of emulating a terminal. This mode supports the same functions as stand-alone operation, but also allows devices to be tested by explicitly specifying their names. In Terminal Mode the IC Tester can also test 2716 and 2732 EPROMs, verifying that they are blank and checking for shorted inputs. PC-Host Mode provides the user with the greatest level of flexibility and power. Like the Terminal Mode, this mode does not require the optional LCD, but does require a connection between the Tester's RS-232C port and that of an IBM PC, XT, or AT, or compatible. PC-Host Mode supports all Terminal Mode functions, but also supports multiple device libraries (including user-generated device libraries), and includes a special debugging feature for debugging user-generated device tests.

The IC Tester's circuit board also supports a standard +5v three-port voltage regulator, allowing operation from 6 C-cell batteries.

HARDWARE OVERVIEW

The IC Tester is controlled by an 8031 single-chip microprocessor (U1), which contains I/O ports, an on-chip UART (serial port), and 128 bytes of RAM (read/write memory). An external EPROM is used to hold the operating program, as well as the test information for the standard device library. The EPROM socket (U6) can support 2764, 27128, 27256 and 27512 EPROMs (by configuring jumper block, JP1), with the 27256 being standard.

The Tester has three connectors, a 25-pin (DB-25S) connector (J1) for its RS-232C serial port interface; a 4-pin power connector (J2); and a 14-pin connector (J3) for connecting the optional LCD. A "pot" (potentiometer, P1) is included to adjust the viewing angle of the LCD, and a power on/off switch (SW2) is also included on the board.

Two pushbuttons ("identify" [PB1] and "retest" [PB2]) are included to facilitate Stand-Alone Mode operation, and a third pushbutton (PB3) allows the Tester to be reset. A four-position DIP switch (SW1) allows the user to select one of four standard baud rates, the desired serial (Terminal or PC-Host) mode, and a special '74Cx' operating mode (described later).

Two LEDs are included on the IC Tester for device status. One LED (D1) is a power-on indicator. The other LED (D2) is a mode status indicator, indicating Stand-Alone Mode when off, and a serial (Terminal or PC-Host) mode when lit.

Six 74HCT244's (U11-U16), three 74LS374's (U8-U10), and numerous resistors provide the circuitry that directly interfaces between the CPU and the universal ZIF socket for testing the ICs. A separate 74HCT374 (U19) is used to control the transistor power and ground switches (Q1-Q6) for the ZIF socket, as well as the mode status LED (D2).

POWER REQUIREMENTS AND CONNECTION

The four-pin power connector (J2) is used to bring +5v (+/-5%) to the IC Tester. The power supply for the standard tester (including the LCD) should be able to supply +5v @650 mA. The power connector pinout is shown in Fig. 1; notice that only two of the four pins are used.

Optionally, a 9v adapter can be used to power the IC Tester. To do this, the 7805 three-port voltage regulator (VR1) must be installed, and the +9v @650 mA supply of power must be provided at the J6 power connector, adjacent to the DIP switch (see Fig. 1). **The +5v and +9v supplies must NOT be connected to the board at the same time.**

If the IC Tester is to be powered from batteries, special considerations are in order. It is recommended that standard MOS and bipolar parts be replaced with equivalent CMOS devices to minimize power consumption. Thus, the following substitutions should be made:

Substitute	With
8031 (U1)	80C31
74LS374 (U8-U10)	74HCT374
74LS373 (U5)	74HCT373
74LS244 (U4)	74HCT244
74LS139 (U7)	74HCT139

Six C-cell batteries supply +9v. They can power the the board through the 7805 three-port regulator. Use the auxiliary power input connector, J6.

IMPORTANT NOTE: If the 7805 regulator is going to be used with a high DC voltage at J6, instead of +5v at J2, the +5v output of the 7805 regulator should be checked and verified before the board's ICs are installed. Be sure that +5v appears at the IC power pins.

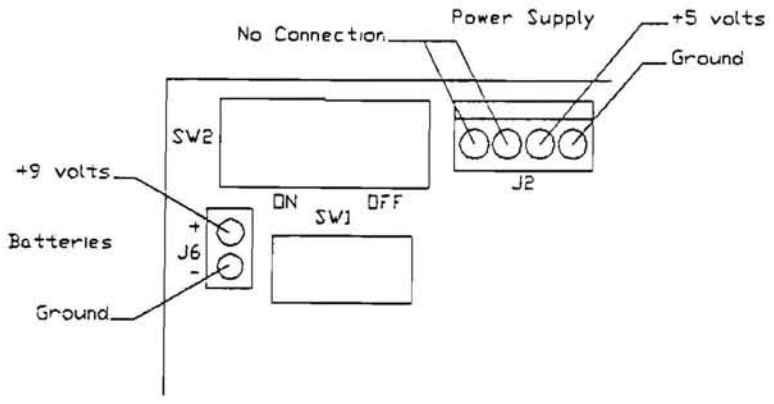


Figure 1 IC Tester J2 & J6 Power Connection Pinout

THEORY OF OPERATION

The Circuit Cellar IC Tester uses a series of "test vectors" to test the device under test (DUT). An "initial" or "output" test vector is first applied to the DUT inputs (with the DUT outputs under load), and a "readback" vector is read back from the DUT indicating the states of all inputs and outputs. This readback vector is then masked with a "don't care" mask and compared to an "expected readback" vector; if they match, the test vector passed, otherwise it failed. This process is repeated for all of the test vectors needed to test the DUT.

The IC Tester uses current-limiting resistors between the digital tester circuitry and the device under test (DUT) to prevent damage to the DUT. The test technique employs a combinatorial latching technique using feedback, which causes a loading effect on each DUT output for a period of 500 ns (nanoseconds), after which the load is effectively removed. The readback vector reads back both the DUT's outputs and its inputs, verifying not only that the outputs are correct, but also that the inputs are not shorted or "stuck" at a particular logic level.

Because the outputs of the DUT are loaded (using a 390 ohm resistor) the outputs of the DUT must be able to supply enough current to overcome the loading and "switch" the state of the combinatorial feedback latch. Most logic families can do this with ease. The notable exceptions are the 74C00-series CMOS devices and the 4000-series CMOS devices. These two families provide only very-low-current outputs, which often have difficulty "pulling down" a load resistor connected to +5v. For these devices, then, a "pull-down" load must always be applied to the outputs of these devices. For the 4000-series devices, this is simply done by specifying pull-down loading in the test vector definitions. For the 74C00-series devices, however, it is desirable to use the standard 7400-series test vectors, without writing special test vectors for the CMOS devices. To this end, the IC Tester supports a '74Cx' mode, which ensures that a pull-down load is always applied to the DUT outputs, regardless of what is specified (pull up load or pull down load) in the test vector definition.

A few 4000-series devices (listed in Appendix A) have difficulty even with pull-down loading. For extensive testing of low-current output devices (i.e., 74C-series and 4000-series devices), the load resistors should be changed from 390 ohms to around 1000 ohms; this would allow support for all low-current output devices, including those listed in Appendix A. Changing to 1000 ohm resistors would, however, disallow testing of many of the standard bipolar devices.

A different problem exists with certain devices with high current inputs. Since the 390 ohm resistors between the IC Tester's test circuitry and the DUT act as current-limiting resistors, there is a voltage drop across the resistors, the magnitude of which is determined by the input current of the DUT inputs. If a DUT input draws an excessive amount of current, the voltage across the current-limiting resistor could be too large, causing the wrong logic level to appear at the DUT input. For this reason, a handful of devices (in particular, a few select 74S-series and 74H-series devices with unbuffered inputs) cannot be tested by the IC Tester.

USING THE IC TESTER

The instructions on operating the IC Tester depend on which mode will be used. This discussion will be broken down into four primary sections. The first section will describe information applicable to all operating modes, while each of the three other main sections will describe one of the Tester's operating modes.

Mode-Independent User Information

The first requirement to operate the IC Tester is an acceptable supply of power. Refer to the "Power Requirements and Connection" section for power supply requirements and connection to the Tester. Whenever power is applied to the IC Tester and the power switch is ON, the power LED (D1) should be lit. If not, recheck the power connection and voltage.

When the IC Tester is first powered up, the mode status LED (D2) should also be lit, indicating that a serial operating mode is selected. This LED remains lit unless Stand-Alone Mode operation is selected. If Stand-Alone Mode operation is attempted without the LCD being present, the mode status LED will flash to indicate the error; the Reset pushbutton must be pressed to return to the selected serial mode of operation.

The IC Tester is capable of supporting six different "Device Types." Device Types are determined by number of pins and the power and ground pin numbers. The six Device Types supported by the IC Tester are given in Table 1.

Device Type	Number of Pins	GND	Power
1	14	7	14
2	14	11	4
3	16	8	16
4	16	12	5
5	20	10	20
6	24	12	24

Table 1. Device Types supported by the IC Tester.

Devices that do not fit into one of these Device Types cannot be tested directly. It is possible, however, to test these devices by creating an adapter socket which switches a few lines around to place the power and ground pins in acceptable positions. The adapter can then be inserted into the ZIF socket, and the device to be tested can be inserted into the adapter.

When devices are placed in the ZIF socket, they must be "bottom justified." The bottom of the ZIF socket is the end furthest from the handle. The bottom of the device to be tested is the end furthest from (opposite of) pin 1. The device to be tested must be placed in the ZIF socket so that its bottom is inserted into the bottom of the ZIF socket (see Fig. 2).

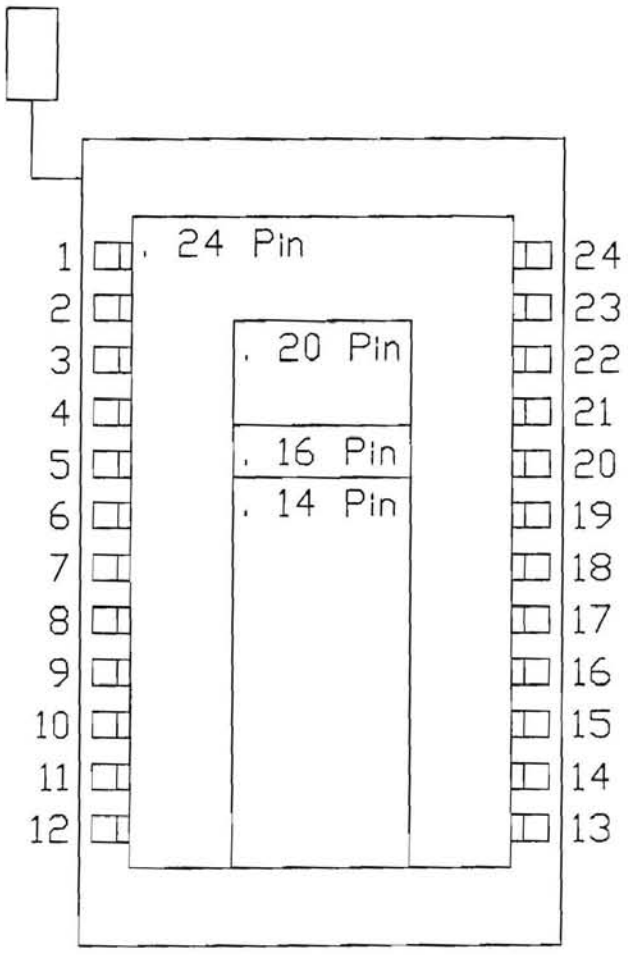


Figure 2 ZIF Socket IC Insertion Orientation Guide

Figure 3 shows a summary of the DIP switch (SW1) functional breakdown. When the IC Tester powers up, it selects one of two operating modes -- Terminal Mode or PC-Host Mode -- depending on the setting of switch SW1-4. If SW1-4 is ON (closed), Terminal Mode is selected; if SW1-4 is OFF (open), PC-Host Mode is selected.

The baud rate is selected by switches SW1-1 and SW1-2. One of four standard baud rates is selected (300, 1200, 2400, 9600), depending on the combination of these two switches, as shown in Fig. 3. The remaining SW1 switch, SW1-3, is used to select or deselect '74Cx' mode operation when operating is Stand-Alone Mode. Its function is described more completely below.

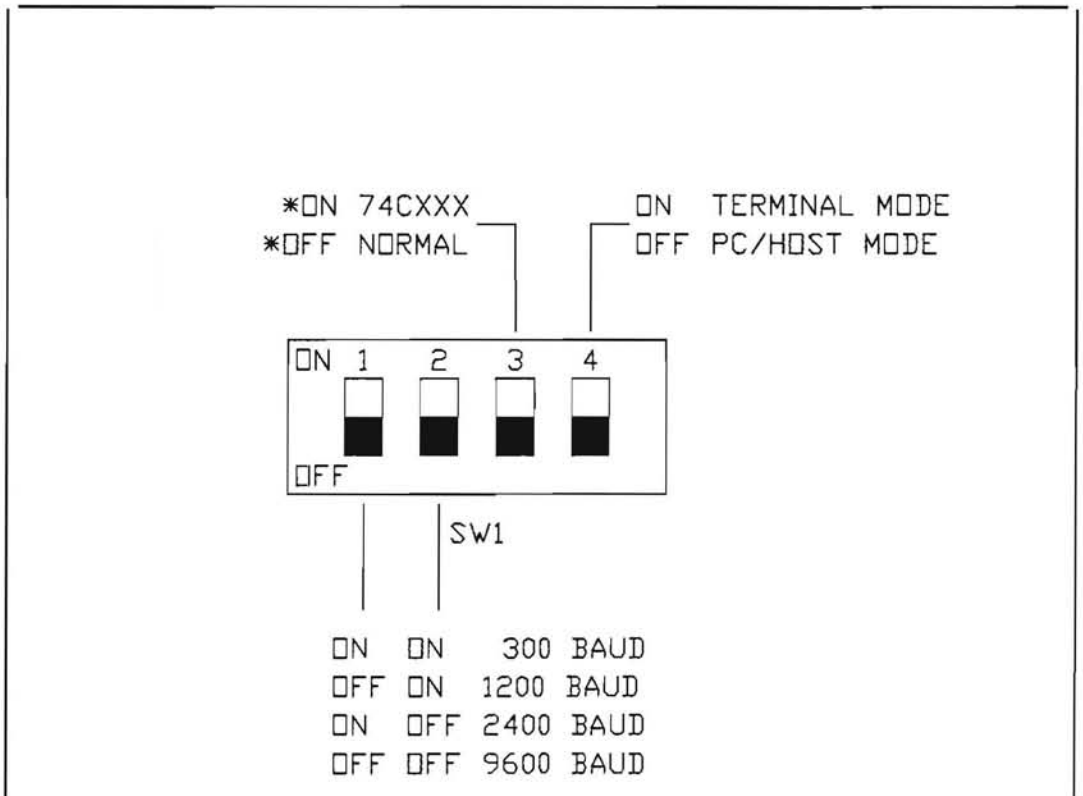


Figure 3 DIP SWITCH Functional Breakdown

Jumper block JP1 is used to configure the EPROM socket (U6) for the type of EPROM to be used. Figure 4 shows the various jumper block configurations for the supported EPROMs; the 27256 is the standard EPROM supplied with the unit.

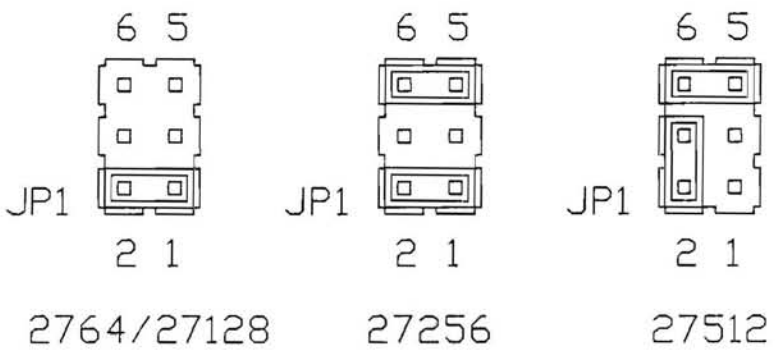


Figure 4 EPROM Type Selection Jumpering og JP1

Stand-Alone Mode Operation

The IC Tester always powers up in one of the two serial operating modes (Terminal Mode or PC-Host Mode). To enter Stand-Alone Mode, simply press the "identify" pushbutton (PB1). If the LCD is not installed, the mode status LED (D2) will flash to indicate the error, and the Reset pushbutton must be pressed to return to the selected serial operating mode. If the LCD is installed, the mode status LED will turn off, and the LCD will display a "Stand-Alone Mode" message. (The baud rate and serial mode settings determined by SW1-1,3,4 are irrelevant in Stand-Alone Mode.)

Stand-Alone Mode operation requires no connection to the IC Tester's RS-232C port, and is the Tester's simplest operating mode. To test a device, merely insert the device into the ZIF socket (bottom justified, as described above) and press the "identify" pushbutton. Assuming the device is supported by the IC Tester, if the device is identified, it passed its test vectors, indicating that it is a good part. If the "CANNOT IDENTIFY" message is displayed, the tester was unable to identify the device, indicating that it is a bad part.

The "identify" operation is also helpful for users wanting to identify unmarked or improperly marked devices, such as those often found in IC "grab bags." If an unknown device is being identified and the tester is unable to identify it, the device may either be defective, or it may be a device not currently supported by the IC Tester.

In some instances it may be desirable to determine which pins failed on the DUT. The IC Tester can only identify good parts, however, so pin failures of defective devices cannot be determined using the Tester's identify operation. To allow pin failures to be displayed, a second pushbutton, "retest," is included. The "retest" pushbutton causes the test vectors for the last identified device to be applied to the current DUT. If any of the test vectors fail, the pin failures will be displayed on the LCD.

To take advantage of the Tester's retest capability, a good device must first be inserted in the ZIF socket and identified. Once identified, one or more "suspect" devices of the same type can be inserted in the ZIF socket (one at a time, of course) and "retested," using the same test vectors as the first device. If any of the subsequent devices fail, the detected pin failures will be displayed.

In some cases, devices having different part numbers are functionally identical to the IC Tester. In such cases, one device is used as the "master" device name, and other devices are considered "clones." If an attempt is made to identify a clone device, the name of the corresponding master device will be displayed. For example, the 74LS04 and 74LS14 are functionally identical parts, except that the 74LS14 has Schmitt-trigger (hysteresis) inputs. Since the 74LS14's inputs do not affect operation from the IC Tester's standpoint, it is considered a clone of the 74LS04. Therefore, if a 74LS14 is placed in the Tester for identification, the Tester will identify it as a "7404".

To keep from potentially damaging ICs, the algorithm used by the Tester to identify devices supports only devices having corner power and ground pins (that is, Device Types 1, 3, 5, and 6). Devices with other power and ground connections, such as the 7475, cannot be identified by the IC Tester (many such devices can, however, be tested in Terminal or Stand-Alone Mode).

When identifying or retesting 74C00-series devices, the operation of the IC Tester needs to be modified slightly. Since these devices lack the high-current output capability of other device families, the Tester must always supply pull-down loads to all of the outputs of these devices. (Normally, device outputs are pulled in the direction opposite the expected output value, to create a loading effect). This special mode is called simply '74Cx' mode, and is controlled by switch SW1-3 during Stand-Alone Mode operation. When SW1-3 is ON (closed), '74Cx' mode is selected; when OFF (open), normal mode is selected.

Stand-Alone Mode can only be exited by pressing the Reset pushbutton.

Terminal Mode Operation

Terminal Mode operation includes all of the functionality of Stand-Alone Mode, as well as some additional features. For Terminal Mode operation, the IC Tester's RS-232C port must be connected to a dumb terminal or to microcomputer capable of emulating a terminal.

The computer connection varies from one machine to another. The IC Tester transmits on pin 3 of its DB-25S connector (J1), and receives on pin 2 (pin 7 is signal ground). Since most terminals and microcomputers transmit on pin 2 and receive on pin 3, a simple three-wire "straight-through" cable will generally work (see Fig. 5). In many cases, however, the terminal or computer will require active "handshake" lines for proper operation. To support these devices, pin 6 on the IC Tester's DB-25S connector provides an always-active handshake signal. This signal can be tied to the handshake inputs (typically pins 5, 6 and 8) of the terminal or computer, as needed. Figure 6 shows a common cable with tied handshake lines.

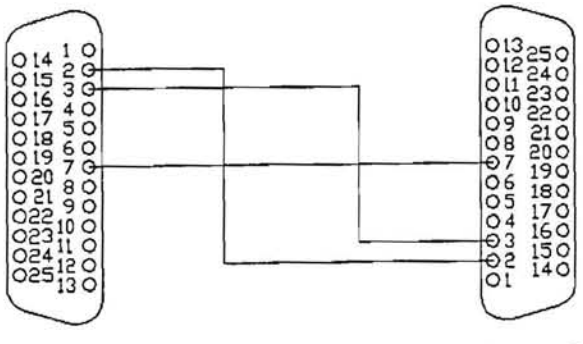


Figure 5 Straight-through cable w/no handshaking

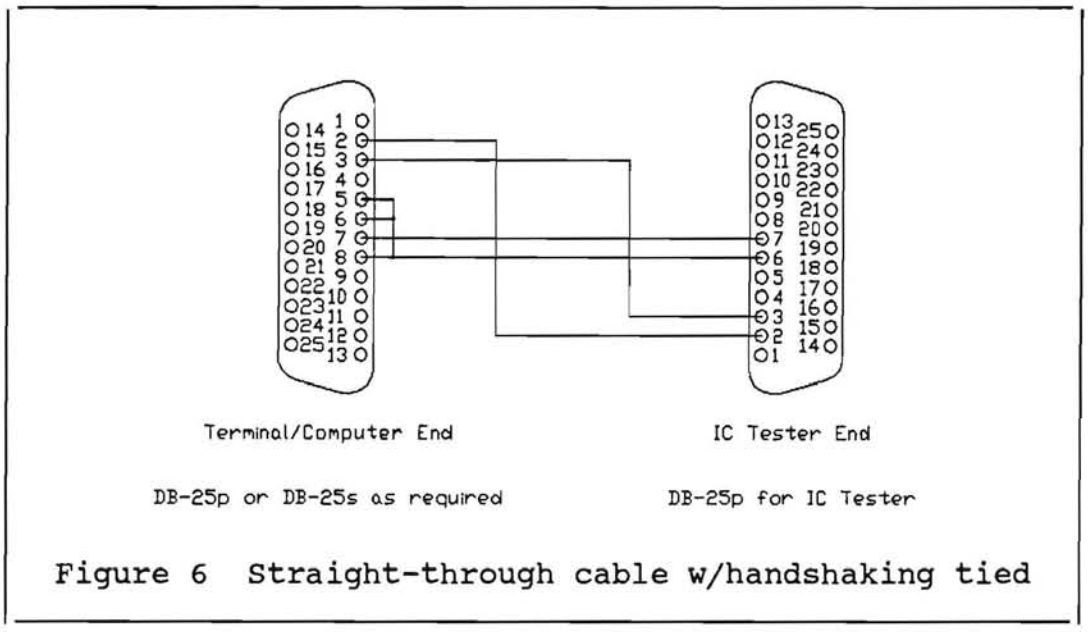


Figure 6 Straight-through cable w/handshaking tied

Terminal Mode is selected when SW1-4 is ON (closed). The setting of this switch is only checked at power-up or during Reset; thus, changing the switch setting when operating in another mode has no effect until Reset is pressed.

The communication baud rate is selected by SW1-1 and SW1-2, as described above. The desired baud rate should be selected using these switches, and must match the baud rate of the attached terminal. Most users will, no doubt, select 9600 baud operation for maximum performance. Other speeds can be chosen for special circumstances. As with the mode select switch, the baud rate switches are only checked at power-up or during Reset.

NOTE: Terminal Mode operating problems are almost always a result of a cabling problem, a baud rate problem, or an improper SW1-4 setting. Make sure these are properly configured before operating the Tester.

When Terminal Mode is selected and the Tester is powered up (or reset), the Tester displays a signon (copyright) message followed by a user menu on the terminal (see Fig. 7). Each menu item has an associated number that can be used to invoke the function. Some of the menu items also have single-letter alternatives to the numbers; such menu items can be invoked by pressing either the corresponding number or the corresponding letter. The letter 'M' can be pressed at any time to redisplay the menu.

```
*****************************************
***       CIRCUIT CELLAR IC TESTER      ***
***  Terminal Mode Operation    Ver 1.0  ***
***     (C) 1987 Circuit Cellar Inc.     ***
*****************************************

       *** MENU ***

I (1) IDENTIFY DEVICE
T (2) TEST SPECIFIED DEVICE
  (3) 2716 BLANK CHECK
  (4) 2732 BLANK CHECK
C (5) SET 74Cx CMOS MODE
M (6) DISPLAY MENU


Enter Selection:
```

Figure 7 Initial Terminal Mode Screen

The "Identify" menu option is essentially identical to the similar Stand-Alone Mode function. It causes the Tester to attempt to identify the device in the ZIF socket. Only parts with corner pinouts (Device Types 1, 3, 5 and 6) can be identified.

Another menu item, "Test Specified Device," allows the user to enter the name of the device to be tested. When this option is selected, the name of the last device identified or specified is displayed on the terminal. Pressing RETURN will "retest" that device, applying the test vectors for the specified device to the DUT (the same as pressing the "retest" pushbutton in Stand-Alone Mode). If a different device is desired, the user needs simply to enter the new device name; upon pressing the first character of a new device name, the previously-displayed device name is erased and the new device name is displayed (echoed) on the terminal as it is typed.

When specifying a device name, the device's "generic" name should normally be used. Thus, for example, when testing a 74LS04, specify the device name, "7404". There can, however, be exceptions to this rule. In some cases, a part in one family may operate differently than a similarly-numbered part in a different family. In such cases, the "generic" number should be used for the majority of cases (if applicable), and a "specific" number should be used for exceptional cases. For example, the 7495 and 74LS95 devices have corner power and ground pins (pins 7 and 14), while the 74L95 has center power and ground pins (pins 4 and 11). The generic name "7495" applies to the 7495 and 74LS95 parts, whereas the specific name "74L95" has been assigned to the 74L95.

The ability to specify the device being tested has two particular advantages. First, if the device is bad, the Tester can determine and display the failed pins. Second, devices with non-corner power and ground pins can be tested. For example, the Tester cannot identify a 7475 because it does not have corner power and ground pins. The Tester can, however, test the 7475 by specifying its name explicitly, using the "Test Specified Device" menu option.

As described in the "Stand-Alone Mode Operation" section, devices sometimes have functionally-identical "clones." As mentioned, when identifying a device that is a clone, the name of the master device is displayed. When explicitly specifying a device name, however, clone names are acceptable. The IC Tester will automatically look up and use the test vectors of its master device. For example, if the device name "7414" is entered to test a 74LS14, the IC Tester will know to apply the "7404" test vectors to the device.

As in Stand-Alone Mode, '74Cx' mode can be selected in Terminal Mode. This is done by selecting the proper menu item. When this is done, the menu automatically changes to reflect the new mode; the "Set 74Cx CMOS Mode" menu item is replaced with the menu item, "Clear 74Cx CMOS Mode". For a description of '74Cx' mode, see the discussions in the sections entitled "Theory of Operation" and "Stand-Alone Mode Operation."

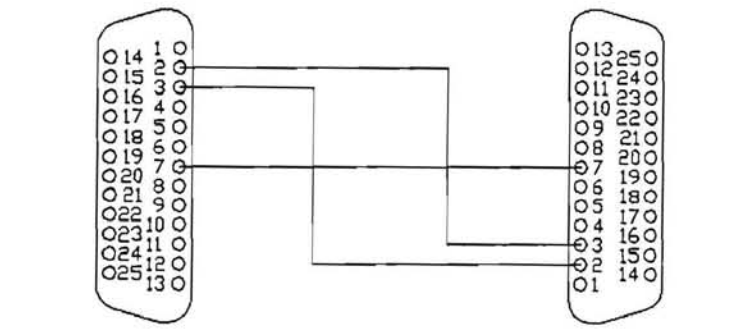
In addition to the features already described, Terminal Mode operation also supports 2716 and 2732 EPROM blank verification. When an EPROM is inserted in the ZIF socket and the appropriate menu item is selected, the Tester will determine whether or not the EPROM is blank. If it isn't, the first non-blank address is displayed. The Tester also checks for shorted EPROM inputs, and displays an appropriate error message, including pin numbers, if bad pins are detected.

PC-Host Mode Operation

PC-Host Mode is the most flexible of the IC Tester's operating modes. It includes all of the features of Stand-Alone Mode and Terminal Mode, along with additional features that allow you to create your own device libraries.

PC Serial Port Connection

For PC-Host Mode operation, the IC Tester must connect to the RS-232C port of an IBM PC, XT, AT or compatible computer. The cable required to connect to systems having a 25-pin RS-232C connector (DB-25P) is shown in Fig. 8. For systems with IBM's 9-pin RS-232C connector, use the cable configuration shown in Fig. 9. Note that these cables have only three wires. Standard "straight through" cables that have additional wires are also acceptable; the additional wires merely remain unused.



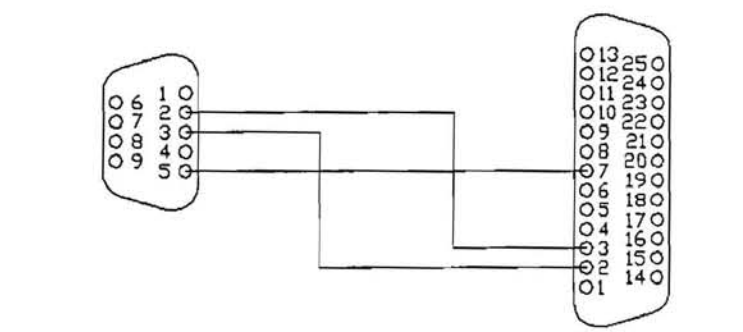
IBM PC or XT or AT End

DB-25s for IBM

IC Tester End

DB-25p for IC Tester

Figure 8 DB25 IBM Cable



IBM PC or XT or AT End

DB-9s for IBM

IC Tester End

DB-25p for IC Tester

Figure 10 DB9 IBM Cable

The power-up mode of the IC Tester is determined by switch 4 of DIP switch SW1. When this switch is OFF (open), the Tester powers up in PC-Host Mode. As described earlier, switches SW1-1 and SW1-2 determine the baud rate of the IC Tester's RS-232C serial port. The current PC software requires the Tester to be set for 9600 baud for PC-Host Mode operation. The PC software automatically sets the PC's baud rate to 9600 baud. The software also permits either COM1 or COM2 (i.e., PC serial port 1 or serial port 2) to be chosen for IC Tester communications.

Programs and Files

The PC-Host Mode software comes on two 360K double-sided, double-density diskettes. The first diskette contains these files:

veccpt.com	vector compaction program (executable).
ict.com	PC IC Tester operating program (executable).
ictctl.hex	IC Tester (resident) operating program with standard device library (Hex/ASCII format).
vector.cpt	standard device library in compacted format.

The second diskette contains the following files:

ictvec__.tst	vector definition files for devices in the standard device library.
vector.vdr	vector directory file for standard device library.

Source files with a '.pas' extension are for Turbo Pascal 3.0. (Note: Turbo Pascal 4.x requires modifications.) Source files with a '.asm' extension are 8086 assembly language files. Similarly, source files with a '.a51' extension are 8051 assembler files.

The file 'ictctl.hex' is a Hex/ASCII image of the compiled IC Tester operating program combined with the standard device library -- the code that resides in the Tester's on-board EPROM. Being in Hex/ASCII format allows the file to be easily downloaded to an EPROM programmer.

The remaining programs and files will be described later in this section, as appropriate.

Basic Operation

The most basic PC-Host Mode operation requires only two files: 'ict.com' and 'vector.cpt'. To initiate PC-Host mode operation, insert the diskette containing these files into drive a: of your PC (users that have a hard disk can copy the files to the hard disk if desired) and enter the command:

A: <RETURN>

to set the system default to drive a: (hard disk users will need to use 'c:'or 'd:' instead). Now execute the operating program, 'ict.com', by simply typing its name, "ict", followed by a RETURN (making sure that both programs reside in the current DOS directory if using a hard disk). The program will prompt you for the name of the vector compaction file, with a default of 'vector.cpt' (normally the standard device library); press RETURN to accept the default. You will then be prompted for which serial port is to be used, with a default to COM1. If you are using COM1, accept the default by pressing RETURN; otherwise, enter "2" (to specify COM2) and press RETURN.

Once this information has been accepted, ict attempts to initialize the IC Tester and ask for its current software version level. If a communication problem is detected, ict retries up to two more times before considering the communication link a hard failure. If this occurs a "COMMUNICATION ERROR" message is displayed, and the program exits to DOS.

If communication is established with the IC Tester, the Tester's software version number is displayed, and ict proceeds to read the device vector information from the vector compaction file (normally 'vector.cpt'). After reading the entire file, ict displays the number of devices in the library. The program then displays a menu, yielding a screen like that shown in Fig. 10.

The PC-Host mode menu is similar to that displayed by the IC Tester during Terminal Mode operation, with two primary exceptions. First, PC-Host mode doesn't support the single-letter menu item selection capability present in Terminal Mode; all PC-Host Mode menu items must be selected using the menu item number (except for 'Quit'). Second, the PC-Host Mode menu offers a 'Quit' option to return to DOS, and supports an additional feature called "debugging mode," which will be described later.

Parts can be identified by merely placing them into the Tester's ZIF socket and selecting the "Identify Device" menu item (1). To specify the name of the part to be tested, select menu item 2, "Test Specified Device". The '74Cx' mode is also supported as a menu-selected option, as described in the sections entitled "Theory of Operation" and "Stand-Alone Mode Operation."

```
     ******************************************
     ***       CIRCUIT CELLAR IC TESTER       ***
     ***  Terminal Mode Operation    Ver 1.0  ***
     ***     (C) 1987 Circuit Cellar Inc.     ***
     ******************************************

Enter compaction filename [VECTOR.cpt]:
IC Tester using software version: V1.0
Reading device vector info... done -- 237 devices read.

           *** MENU ***

    I (1) IDENTIFY DEVICE
    T (2) TEST SPECIFIED DEVICE
      (3) 2716 BLANK CHECK
      (4) 2732 BLANK CHECK
    C (5) SET 74Cx CMOS MODE
    M (6) DISPLAY MENU



    Enter Selection:

Device:
Message:
Pin Failures:
```

Figure 10 ICT Menu Display

As in Terminal Mode, 2716 and 2732 EPROMs can be tested for shorted inputs, and to verify that they're blank. See "Terminal Mode Operation" for more details.

Unlike Terminal Mode operation, PC-Host Mode operates with a fixed display format. The menu is always present, indicating your device test options, and three fields are always displayed, showing information about the most recent operation. The three fields are Device, Message, and Pin Failures.

The Device line shows the name of the device most-recently tested. It is also where a new device name is entered if a specified device is to be tested. As in Terminal Mode, when the menu option "Test Specified Device" is selected, the name of the most-recently identified or tested device is displayed (on the Device line in PC-Host Mode); simply pressing RETURN will select that device name. A new device name can be entered by merely entering its name. When the first character of the new device name is entered, the previously-displayed name is erased, and the new device name is displayed as it is typed.

The Message line is where status messages are displayed; messages such as "Device Identified", "*** DEVICE FAILED ***", and "74Cx Mode Selected" are displayed on the Message line. In general, the Message line indicates the outcome of the most recently selected menu item.

The Pin Failures line is where failed device pins are indicated, for devices that fail testing. Pin failures can only be determined (and thus displayed) when testing a specified device or an EPROM.

The ict program supports an alternate approach to specifying the device vector compaction filename and communication port. Parameters can be included as part of the command line; the parameters must follow "ict" on the command line, with spaces used for separation. If one parameter is specified, it is assumed to be the name of the vector compaction file (the '.cpt' extension is still assumed, if no extension is specified explicitly). If two parameters are specified, the first is assumed to be the vector compaction filename and the second is assumed to be the communication port (1 or 2). For example, if this line is entered,

```
ict vector 2
```

the ict program will be invoked, and will automatically select 'vector.cpt' as the device vector compaction file, and COM2 as the serial port for IC Tester communications.

One of the primary benefits of allowing the parameters to be included on the command line is that DOS batch files can be used to invoke the program, without any prompting required. This is particularly beneficial if multiple device libraries are available on a system -- a separate batch file can be used to invoke the program for each device library.

Communication Errors

The IC Tester and its associated ict software were designed considering potential communication problems. If a communication problem between the PC and the IC Tester occurs at any time, a "COMMUNICATION ERROR" is displayed to notify you of the problem. If a gross communication failure -- such as the communication cable getting unplugged -- occurs during an IC test, the IC Tester automatically turns off power to the DUT to prevent any possible damage. Once communication is properly re-established, the test can be restarted.

ADVANCED OPERATION -- CREATING YOUR OWN TEST VECTORS

The basic operation of the IC Tester will provide all of the functionality you're likely to need, as long as the devices you want to test exist in the Tester's standard device library. But what about other devices -- new devices or semi-custom devices, such as PALs? The IC Tester can be used for testing these devices as well! With the included IBM PC software, you can create test vectors for most digital devices. The basic requirements are that the device must fit into one of the six Device Type categories supported by the tester, and must be capable of supplying a "reasonable" amount of current at its outputs, without "extreme" input current consumption. Most IC families fit easily into these requirements, including MOS, high-speed CMOS, and virtually all TTL families (such as LS-TTL), with the exception of only a few problem parts.

To explain how to develop test vectors for new parts, it is best to begin with an explanation of test vectors and how they are used in the IC Tester. Examples will then be used to illustrate the process of creating and compiling sets of test vectors for custom devices and device libraries.

In order to define test vectors for many devices, it is important to first develop a straightforward means of describing the vector information. A compiler can then be written to convert the test vector definitions into a format usable by the computer for actually testing the devices. The complete definition of the test vectors and other information for a single device is called a device test vector definition module, or simply a vector definition module.

What information is needed to define a device and its test vectors? Clearly, the device must have a name. Eight characters seemed like a reasonable name length limit, so this is what was implemented. The tester must also know the size of the device, and the locations of the power and ground pins. Other than these items, the only other obvious information required is the test vectors themselves. In order to perform a fast device identification, however, the tester must also know which pins are inputs, which are outputs and which are tri-state (that is, having a "high-impedance" state).

A test vector merely specifies the high (1) and low (0) logic levels that are to be written to the pins of the device under test (DUT); some of the pins will be inputs and others will be outputs. A test vector that is written to the DUT pins is referred to as an output vector (or initial vector), since it is written out to the DUT. To determine if the DUT responded properly to the output vector -- that is, to make sure outputs switched as expected and to verify that no inputs are shorted --a corresponding read-back vector must be read back from the DUT and compared to an expected read-back vector. Thus as you develop test vectors for a device, each complete test vector consists of two parts, an output vector and an expected read-back vector.

It may seem that a value written to a DUT output pin is meaningless (and could "short" the pin), but all logic values reach the DUT through load and current-limiting resistors; thus a value written to a DUT output merely indicates which direction (pull up or pull down) the loading effect is to be realized. For most devices, it is desirable to load each output in the direction opposite its expected output value to make sure that the device works under load conditions, and to make sure that the value read back from the DUT is caused by the DUT itself, and not by the value written out to the DUT pin by the Tester.

To aid in the documentation of the device test vector definition modules, it is also desirable to allow comments; this feature has also been implemented. All characters on a line following an asterisk (*) are ignored by the test vector compiler. Comments may appear anywhere within the vector definition file. A whole line may be commented by placing an asterisk in the first character position of the line.

The final format chosen for specifying the vector definition modules is shown in Fig. 11. The order of the different line types is important, though comment lines may be freely interspersed.

```
# DeviceName                * DEVICE NAME RECORD
* COMMENT LINES MAY BE INTERSPERSED
* FOR DOCUMENTATION AND CLARIFICATION PURPOSES.
S #pins Gpin Ppin           * DEVICE SETUP RECORD
F  I   O  ...  T  I  I      * PIN FUNCTION RECORD
P  p#  p# ...  p# p# p#     * PIN NUMBER RECORD
I  0   1  ...  1  1  0      * INITIAL (OUTPUT) VECTOR
R  1   0  ...  1  X  0      * EXPECTED READ-BACK VECTOR
I  ...
   ...
.
.
.
E                           * END OF DEFINITION RECORD
```

Notes:

DeviceName = Name of device
* = start of comment area
#pins = Number of pins on the IC
Gpin = IC ground pin number
Ppin = IC power pin number
p# = IC pin number

Figure 11 Device Test Vector Definition Module Format

The best way to understand the vector definition module format is by example. Figure 12 shows the test vector definition module for the 7400 quad two-input NAND gate. As its name implies, the 7400 contains four two-input NAND gates; its pinout is shown in Fig. 13.

Referring to Fig. 12, the first line is the Device Name line (having a pound sign (#) in the first column); this line must be the first non-comment line in a vector definition module. The name of the device may appear anywhere on the line after the pound sign; preceding and following spaces are ignored. As a general rule, a device name should be kept as "generic" as possible; instead of using the name "74LS00" use "7400", and so on. There are, of course, cases where exceptions must be made (as described earlier), so use your best judgement.

```
# 7400        *QUAD 2-INPUT NAND.
S 14 7 14
*   NAND 1      NAND 2      NAND 3      NAND 4
F   I  I  O     I  I  O     I  I  O     I  I  O
P   1  2  3     4  5  6     9 10  8    12 13 11
I   0  0  0     0  1  0     1  0  0     1  1  1
R         1           1           1           0
I   0  1  0     1  0  0     1  1  1     0  0  0
R         1           1           0           1
I   1  0  0     1  1  1     0  0  0     0  1  0
R         1           0           1           1
I   1  1  1     0  0  0     0  1  0     1  0  0
R         0           1           1           1
E   * OF 7400
```

Figure 12 Device Test Vector Definition Module For 7400

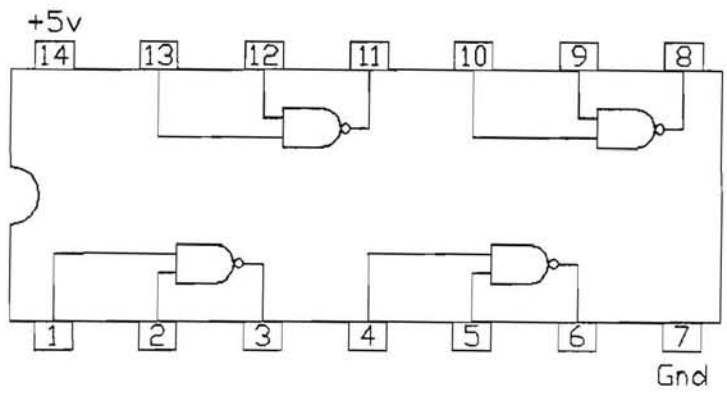


Figure 13 7400 Pinout

The second non-comment line of the vector definition module is the Set-up line, which has an 'S' in the first column. Three numbers with delimiting spaces must follow the 'S'; the first number indicates the number of pins the device has, the second indicates the ground pin number, and the third indicates the power pin number. These numbers tell the compiler (and the tester) what the chip's Device Type is. As described earlier, six Device Types are supported by the IC tester. The 7400 is a Type 1 device, having 14 pins with its ground on pin 7 and its power (+5v) on pin 14.

The Set-up line is followed by a comment line indicating how the chip's four independent NAND gates are grouped. The next line is the Pin Function line, having an 'F' in the first column, indicating the pin functions for the device pins involved in the test. This line specifies a pin function identifier for each pin, with the identifiers being separated by one or more spaces. Valid identifiers are 'I' for input pins, 'O' for output pins, and 'T' for tri-state pins.

Besides specifying the pin functions, the Pin Function line also determines the columnization for the remainder of the vector definition module. All 1's and 0's in the test vectors must be aligned under these columns, and the pin numbers in the Pin Number line (the next line in the definition) must also be aligned under these columns.

As just mentioned, the next line in the vector definition module is the Pin Number line; it has the letter 'P' in the first column. This line specifies the device pin numbers for the pins used in testing the device. These must, of course, correspond to the pin function identifiers specified in the Pin Function line, and must fall in the columns defined by the pin function identifiers. If the pin number for a column has two digits (e.g., pin 13), either of the two digits may fall in the column.

The next several lines in the vector definition module are the actual test vectors. The lines beginning with 'I' are the "Initial" vectors (output vectors), and the lines beginning with 'R' are the expected read-back vectors. For 'I' vectors, the acceptable identifiers are '1' and '0', corresponding to high and low digital values, respectively. For 'R' vectors, acceptable identifiers are '1', '0' and 'X', with the 'X' identifier indicating a "don't care" condition; that is, 'X' indicates that the specified pin should be ignored when comparing the actual read-back vector to the expected read-back vector. For maximum testing benefit, the 'X' identifier should be used sparingly --only in situations when a device output truly cannot be known.

You may notice in the Fig. 12 example that several of the expected read-back vectors are missing 1/0 information in some of the columns. This is because of a feature that was included in the vector compiler to enhance readability and minimize typing. If the 1 or 0 bit value of a column does not change from one line to the next, leaving the column blank in the subsequent line(s) implies that the value should be the same as the last value explicitly stated for that column. For example, the second 'I' vector for the 7400 definition module specifies the bit values 0 and 1 in the first two pin columns, respectively. The corresponding 'R' (expected read-back) vector leaves those columns blank, implying that the expected read-back values for the corresponding pins (pins 1 and 2) are 0 and 1, respectively --the same values that appeared in those column positions in the previous 'I' vector. This feature is particularly helpful for device input pins, since the expected read-back value of an input pin should always be the same value as that specified in the corresponding output vector.

The last line in the vector definition module is the End line, which begins with an 'E'. The 'E' is the only letter required to specify the end of the vector definition module, although comments are permitted (indeed, recommended), as they are on all other lines.

Clones

Another feature has also been incorporated to save effort as well as memory space: cloning. As described earlier, there are many instances where an IC is functionally identical to another device having a different part number. If you wish to support a device that is functionally identical to a previously-defined device (that is, the same test vectors would pass on both devices), you can declare the new device to be a clone of the other (the master). An example of this is shown in Fig. 14. The Device Name and End lines are the same as the standard vector definition module, but only the 'C' ("Clone") line is found between them, indicating the name of the master device.

```
# 7437          * QUAD 2-IN NAND BUFFERS
C 7400          * CLONE OF 7400 (DEF. TYPE = 1)
E               * END OF 7437 DEVICE DEFINITION
```

Figure 14 Clone Definition Example

When a clone is specified, the vector compiler automatically looks through the six Device Type tables to find the specified device name. The clone device is then assigned the same Device Type.

When creating device libraries, any number of device vector definition modules may be included in a file, and any number of vector definition files (within the limits of the system) may be grouped to form a device library. There are, however, compiler memory constraints that must be observed, as described shortly.

The Vector Compaction Compiler

The vector compaction compiler, 'veccpt', is a Turbo Pascal program that accepts files conforming to the device test vector definition format described above. It converts the device and vector information into a single, compact module that the computer and tester can use to test the devices. The source code is in the file 'veccpt.pas'.

Since vector definition modules can take up a lot of file space, particularly if some complex device are included, it would be too limiting to require all of the vector definition modules to exist in a single source file. Indeed, the vector definition files for the over-200 devices included in the standard device library consume over 300K of disk space; that would make for a rather large and unmanageable file! To circumvent this limitation, veccpt references a vector directory file for the names of the vector definition files to be included in the compilation. Each line of the vector directory file specifies the filename of a file containing device test vector definition modules (the test vector "source code"). The files are processed in the order they are named in the vector directory file. The vector directory file used to compile the vector definition files for the standard device library is shown in Fig. 15.

```
ictvec1.tst
ictvec3a.tst
ictvec3b.tst
ictvec4.tst
ictvec5.tst
ictvec6.tst
ictvec7.tst
ictvec8.tst
ictvec9.tst
ictvec2.tst
ictvec15.tst
ictvec41.tst
ictvec42.tst
ictvec43.tst
```

Figure 15 Clone Definition Example

To invoke the compiler from PC/MS-DOS, you simply enter 'veccpt' (without the quotes, of course), and the program will prompt you for the name of the vector directory file and the name of the compaction output file. The vector directory file has a default extension of '.vdr', although this can be overridden by including a different extension. Similarly, the output file has a default extension of '.cpt', but can be overridden.

When veccpt is invoked, it presents a default vector directory filename of 'vector.vdr'. Hitting your RETURN key confirms this default; entering a different name overrides the default. If no extension is included in the new filename, '.VDR' is automatically appended.

After the vector directory filename has been entered, you are prompted for the output filename, and are given a default of the base vector directory filename with a '.cpt' extension, normally 'vector.cpt'. As with the directory filename, you may press the return key to use the default, or enter a new filename if desired ('.cpt' is automatically appended if no extension is specified).

For the "power user," the vector directory filename and the output filename can be specified on the command line, eliminating the prompting by the program (if only one item is specified on the command line, it is assumed to be the input filename and you are still prompted for the output filename). By allowing veccpt to be invoked with the parameters on the command line, you can use batch files to initiate your vector compilations. As with using the ict program, this is particularly helpful if you want to create different device libraries instead of having all devices in the same library -- a different batch file could be used to invoke the compilation for each device library.

The veccpt program uses seven primary arrays to store the compacted vector information. The primary array, VectorTable, holds the actual test vector information, including the device pin function information (which pins are inputs, which are outputs and which are tri-state), the output vector bytes, the input vector bytes, and the "don't care" mask bytes.

Because the ZIF socket has 24 pins, three bytes are used for pin and vector information for every device, regardless of size. Thus the pin function and test vector information is stored as if a 24-pin device is being tested. The "don't care" mask generated by veccpt automatically masks the read-back vector pins that are not associated with the device being tested. Thus, for example, if a 20-pin device is being tested, the bits of the 3-byte read back vector associated with ZIF socket pins 1, 2, 23 and 24 will be masked by the "don't care" mask (the power and ground pins are automatically masked, as well).

Each of the six Device Types supported by the IC tester has its own associated array for storing device names and pointers into the VectorTable array; these arrays are called DeviceType arrays. While the VectorTable array uses variable-length records, with each record being the information to support one device, the DeviceType arrays use fixed-length records, with each record containing a 9-byte field for the device name (8 characters for the name and one byte for the name size) and an integer (two-byte) field for the VectorTable pointer.

The VectorTable array is limited to 32767 bytes of information, and each of the DeviceType arrays is limited to 400 device entries (records).

Figure 16 illustrates the information stored in the various arrays, and how the arrays interact. As shown, device names are stored in the appropriate DeviceType array, and the corresponding device pin function and vector information is stored in the VectorTable array. The DeviceType array also has a pointer pointing to the start of the vector information record in VectorTable.

The VectorTable device record begins with a two-byte field indicating the number of bytes in the record. The next three bytes specify which pins are inputs (set bits) and which are outputs (cleared bits). The next three bytes indicate which pins are tri-state; set bits are tri-state. If a pin is indicated as being tri-state, the 1/0 value in the corresponding bit position of the previous input/output definition bytes is irrelevant. By default, veccpt specifies unused ZIF socket pins as being tri-state.

Following the two record-size bytes and six device pin function definition bytes, the actual test vector information begins. Each complete test vector consists of nine bytes in the record. The first three specify the output vector, the next three specify the expected read-back vector, and the last three specify the "don't care" mask.

As veccpt executes, it stores device name, pin function and vector information into the appropriate arrays. Notice that no Device Type information needs to be stored, since a device's type is determined by which DeviceType array it is placed in.

Device clones are handled somewhat differently. When a device is specified as a clone of another device (the master device), the name of the clone is placed into the next available record of the appropriate DeviceType array. The record number of the original device (in the same array) is then determined and the value 32767 is subtracted from the record number; this value (always negative) is then stored in the pointer field of the clone record. Thus, when a negative integer value is found in the pointer field of a device record, the operating software will know the device is a clone of another device. It will then add 32767 to the pointer value to get the record number of the original device, and will execute the test vectors for that device.

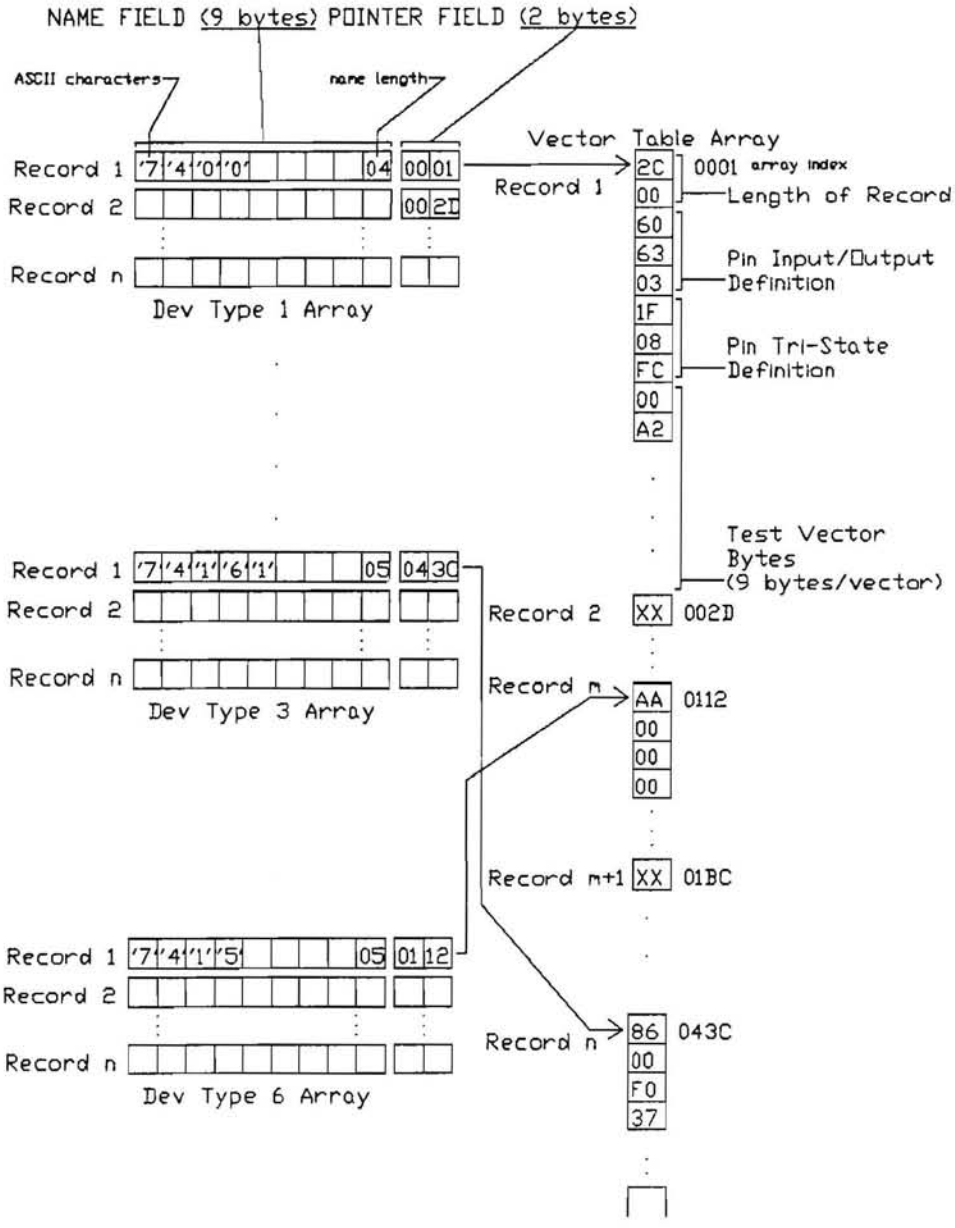


Figure 16 Illustration of Information Storage in Primary 'VECCPT' Arrays, and Array Interaction.

It should be noted that when the veccpt compiler processes a clone, it looks through its arrays to find the named master device. If the specified master device is not found in any of the six DeviceType arrays, an error is generated and the clone device will not be stored in any array (indeed, the compiler would not even know which array should get the clone record). Thus it is essential that clone devices be specified only after (any time after) the corresponding master device has been processed in veccpt's processing sequence.

A typical execution display for veccpt is shown in Fig. 17. As indicated, the program displays the name of the vector directory file, the name of the output (compaction) file, the name of the error file, the name of the current vector file being processed, the number of errors accumulated so far, and the number of devices (successfully) processed so far.

```
*****************************************
***       CIRCUIT CELLAR IC TESTER      ***
***    Test Vector Compaction  Ver 1.0  ***
***     (C) 1987 Circuit Cellar Inc.     ***
*****************************************

Enter directory filename [VECTOR.VDR]:
Enter output filename [VECTOR.CPT


Directory File: VECTOR.VDR
Output File: VECTOR.CPT
Error FILE: VECTOR.ERR
Current Vector File: ictvect3b.ver
Errors:
Devices: 75
```

Figure 17 Typical VECCPT Execution Display

Complete error checking is also included in veccpt, since this is essential for any good compiler. All error information is placed into the 'vector.err' file. Each time a new vector file is opened for processing, the message 'PROCESSING FILE: xxxxx' is also placed into the error file; this allows you to determine in which files the errors occurred. When an error occurs, a copy of the line causing the error is placed into the error file, preceded by an error message indicating the type of error and the line number of the bad line. The list of possible error messages is shown in Table 2; the messages are described in Appendix B.

```
* INVALID START CHAR
* LINE OUT OF PLACE
* SYNTAX
* "END" STATEMENT MISSING -- WARNING
* UNEXPECTED START OF NEW DEVICE
* TOO MANY/FEW PARAMETERS
* COLUMN ALIGNMENT ERROR
* DUPLICATE PIN NUMBERS
* NO VECTORS PROCESSED
* OUT OF ARRAY MEMORY
* DUPLICATE DEVICE NAME
* DEVICE CLONE NOT FOUND
```

Table 2. Veccpt error messages.

When compaction of the test vector files is complete, the compacted information is stored in a binary file (the output file, with the '.cpt' default extension). The format of the data stored in the compaction file is shown in Fig. 18. The program tells you when it is writing the compiled information to the output file, and before finishing execution it displays the number of bytes used in the VectorTable array (of the 32767 bytes available).

It is worthwhile to mention a few important notes about using the veccpt compiler. First, there are no default extensions for the files named in the vector directory file. The extension '.tst' is used for the device test vector definition files in the standard library, but '.vec' also seems like a good choice. In any case, the filename extensions must be explicitly stated when the filenames are placed into the vector directory file. If the vector files do not reside in the directory in which veccpt is executing, the path name to the vector files must also be included.

Second, tabs and other special characters can cause major problems. When you create your vector definition files, the files must be saved as pure ASCII files. If you are using a word processor, such as Wordstar, as your editor, the file must be stored in "non-document" mode, which eliminates special control characters that would otherwise be included in the document. Some editors, like IBM's Personal Editor, store files with tabs in place of some spaces. If you're using an editor that does this, be sure to save the text using the "notabs" modifier. If you do not do this, veccpt will misinterpret many of the lines.

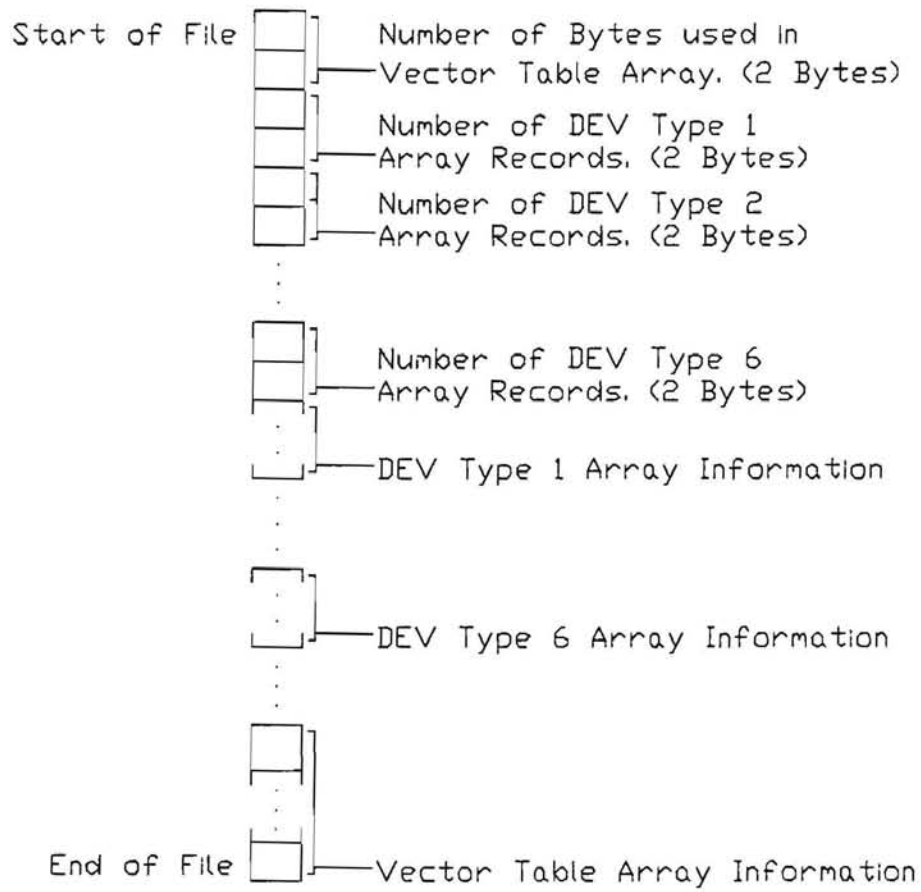


Figure 18 Format of Information Stored in Vector Compaction File (Output from 'VECCPT').

Third, don't be overly concerned about having lots of errors in a compilation. The veccpt compiler processes one line at a time from the source file. If a line has been preceded by the appropriate lines and is syntactically correct, it is processed normally. If the line is out of place or has a syntax error, several flags are cleared (essentially the processing for the current vector definition module is aborted), causing the compiler to begin looking for the start of a new vector definition module (a line beginning with a pound sign (#)). Every non-comment line processed before the start of the next vector definition module is then considered out of place and causes an error. Thus a very simple error early in a vector definition module, such as a column alignment error, will cause every subsequent non-comment line in that module to be flagged as an error.

Fourth, you must pay attention to how the information is placed into the vector definition files. The character in the first column of each line determines what type of line it is, and all processed characters in the file are expected to be in upper case.

Finally, there is one error that is processed only as a warning, and does not abort the processing of the current vector definition module, namely the 'Missing END Line' error. If a vector definition module was being processed properly and the start of a new vector definition module is happened-upon, the fact that an End ('E') line was missing will generate an error, but the device vector definition information will still get stored as if the end line were present. For this to occur, however, two things must be true: at least one complete test vector must have been specified (one 'I' line and a corresponding 'R' line), and the vector definition module must be in an appropriate state to end (that is, not awaiting an 'R' vector specification line).

Diagnostics and Debugging

Once a new device library has been developed using the veccpt compiler, ict can be used to interact with the IC Tester, with the new device library. When test vectors are first developed, however, they must be verified and debugged. To aid in debugging test vectors for new devices, ict includes a 'diagnostic mode' menu option (mentioned briefly earlier), which provides you with additional information about the devices being tested.

When diagnostic mode is set, an extra line, "Vector Failures", is added to the bottom of the display to indicate which test vectors failed when testing a device. When testing a specified device (not when identifying a device) in diagnostic mode, the 'Device:' line indicates the number of pins the device has, as well as the ground pin number and the power pin number. If the device is a clone of another device this is also indicated, along with the device name of the master device. If the device being tested fails, the 'Message:' line indicates how many vectors failed (along with the normal fail message), and the "Vector Failures" line indicates the vector numbers of the first 10 failed vectors (or all failed vector numbers, if fewer than 10 vectors failed). The extra information can prove very helpful when debugging new test vectors. Assuming a known-good device is used for debugging the test vectors, the vector failure information can tell you which test vectors are failing, and this information can be used in conjunction with the pin failure information to show you which test vectors require closer scrutiny.

A Development Example

The tester clearly offers a great deal of flexibility for testing common devices, as well as for developing tests for custom or proprietary devices. Over the last several years, one area of the semiconductor world that has seen phenomenal growth is that of programmable logic. Programmable logic devices (PLDs), such as the ubiquitous PALs (Programmable Array Logic) from Monolithic Memories and other vendors, are generic logic devices containing fuses for determining logic functionality. Often replacing the equivalent of many simpler, standard ICs, PLDs allow complex custom logic functions to be compactly placed into small packages, speeding system designs and reducing circuit board space requirements.

To illustrate the operation of the IC tester, let's take a look at the development of a PAL for a special application, and see how the IC tester can be used to test programmed PALs in production.

Figure 19 shows the logic equations to be placed into the PAL, and the pin assignments chosen for the implementation. The PAL chosen for the implementation is the PAL16L8 -- a combinatorial device supporting up to eight outputs. In order to test the PAL, a series of test vectors must be developed which apply bit patterns to the device inputs and watch for expected output values. The test vectors are based on the logic transfer function (the logic equations) of the device. A set of acceptable test vectors is provided in Fig. 20.

```
PIN 1 = A5              " A5 ADDRESS LINE INPUT
PIN 2 = A4              " A4 ADDRESS LINE INPUT
PIN 3 = A3              " A3 ADDRESS LINE INPUT
PIN 4 = A2              " A2 ADDRESS LINE INPUT
PIN 5 = A1              " A1 ADDRESS LINE INPUT
PIN 6 = A0              " A0 ADDRESS LINE INPUT
PIN 7 = ENABLE\         " ENABLE INPUT (ACTIVE LOW)
PIN 8 = RD\             " READ STROBE INPUT (ACTIVE LOW)
PIN 9 = WR\             " WRITE STROBE INPUT (ACTIVE LOW)
PIN 11 = ERROR\         " ADDRESS ERROR INPUT (ACTIVE LOW)

PIN 12 = CS\            " DEVICE SELECT OUTPUT (LOW)
PIN 13 = CSWR\          " DEVICE WRITE STROBE OUTPUT (LOW)
PIN 14 = CSRD\          " DEVICE READ STROBE OUTPUT (LOW)

CS\ = !(A5 & A4 & A3 & !A2 &!A1 & !A0 & !ENABLE\
      & !ERRROR\);

CSRD\ = !(!CS\ & !RD\);

CSWR\ = !(!CS\ & !WR\);
                              NOTE:  ! = NOT
                                     & = AND
```

Figure 19 Logic Equations for Example PAL16L8 Design.

The next step is to create a vector definition module for the device. Giving the device the name 'PAL1', we create the vector definition module shown in Fig. 21, placing the module into a PC file called 'pals.tst'. In order to compile the vector definition, a vector directory file must be created; the file 'pals.vdr' is therefore created, containing the single line entry, 'pals.tst' (without the quotes).

INPUTS												
A5	A4	A3	A2	A1	A0	EN\	RD\	WR\	ER\	CS\	CR\	CW\
0	0	0	0	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	1
0	0	0	1	1	1	0	1	1	1	1	1	1
0	1	0	1	0	1	0	1	1	1	1	1	1
1	0	1	0	1	0	0	1	1	1	0	1	1
1	1	1	0	0	0	0	0	1	1	0	0	1
1	1	1	0	0	0	0	1	0	1	0	1	0
1	1	1	0	0	0	0	1	1	0	1	1	1
1	1	1	0	0	0	1	0	0	1	1	1	1

Figure 20 Logic Equations for Example PAL16L8 Design.

The vector compilation is now performed, by entering the command: veccpt pals pals

This command invokes the vector compaction compiler, which uses the files specified in 'pals.vdr' (only the file 'pals.tst' in this case) for input, and provides the compacted output in 'pals.cpt'. Error information is placed into 'vector.err'.

To test devices, we run the ict program (with the tester properly connected), insert a PAL to be tested into the ZIF socket (bottom-justified), and specify either the Identify Device option (the easier choice) or the Test Specified Device option, giving the device's name, 'PAL1' (the more-keystrokes option, though pin failures can be identified). As many of the PALs as desired can be tested by inserting them into the tester and selecting the desired test option.

The PC-Host mode diagnostics mode option can also be selected to help in test vector debugging, if necessary.

```
# PAL1                          * PAL 1 DECODER CIRCUIT
S 20 10 20                      * TYPE 5 DEVICE
*    A    A    A    A    A    A    E    R    W    E      C    C    C
*    5    4    3    2    1    0    N    D    R    R      S    R    W
*                                 \    \    \    \      \    \    \
F    I    I    I    I    I    I    I    I    I    I      O    O    O
P    1    2    3    4    5    6    7    8    9   11     12   13   14
**** CHECK FOR ADDRESS SELECT OK ****
I    0    0    0    0    0    0    0    1    1    1      0    0    0
R                                                        1    1    1
I    1    1    1    1    1    1    0    1    1    1      0    0    0
R                                                        1    1    1
I    0    0    0    1    1    1    0    1    1    1      0    0    0
R                                                        1    1    1
I    1    0    1    0    1    0    0    1    1    1      0    0    0
                                                         1    1    1
I    0    1    0    1    0    1    0    1    1    1      0    0    0
R                                                        1    1    1
I    1    1    1    0    0    0    0    1    1    1      1    0    0
R                                                        0    1    1
**** CHECK FOR SELECT/READ OK ****
I    1    1    1    0    0    0    0    0    1    1      1    1    0
R                                                        0    0    1
**** CHECK FOR SELECT/WRITE OK ****
I    1    1    1    0    0    0    0    1    0    1      1    0    1
R                                                        0    1    0
**** CHECK FOR NO SELECT ON ADDRESS ERROR ****
I    1    1    1    0    0    0    0    1    1    0      0    0    0
R                                                        1    1    1
**** CHECK FOR NO SELECT W/O ENABLE ****
I    1    1    1    0    0    0    1    0    0    1      0    0    0
R                                                        1    1    1
E       * END OF PAL1 TEST VECTORS
```

Figure 21 PAL1 Test Vector Definition Module.

CONSTRUCTING THE IC TESTER

When preparing to construct the IC Tester, you must first decide what source of power will be used to power the unit. Refer to the section on "Powering the IC Tester" for the options. If a standard +5v supply is to be used, the 7805 three-port regulator, the 6 C-cell battery holder and the battery clip for the J6 connector are not needed. If batteries are to be used, the regulator, the holder and the clip are required, but the four-pin J2 power connector is not required.

NOTE: Power should NOT be applied to both J2 and J6 at the same time.

The 25-pin DB-25S connector (J1) should be a female part, but is not required if the IC Tester will be used strictly in Stand-Alone Mode. Similarly, the optional LCD is only required for Stand-Alone Mode, and is not required if the Tester is restricted to Terminal Mode and/or PC-Host Mode operation. If the LCD will not be used, the 14-pin male J3 connector is not needed; otherwise J3 should be installed. If mounted on the main board a female mating connector should be attached to the LCD module, if on the front panel board, the LCD module is soldered directly onto J3.

The schematic for the IC Tester is provided in Appendix E, with the component layout (silkscreen) provided in Appendix D, and the (complete) parts list provided in Appendix C. The 390 ohm resistors specified in the parts list are optimum for universally testing nearly all logic families. If the tester is to be restricted to testing special-case devices, such as the 4000-series CMOS devices, refer to Appendix A for a discussion on load resistor considerations.

ASSEMBLING the IC Tester (main board)

1. ( )

Check the parts list to become acquainted with the components and to verify that you have all that are needed. Once you have done this, you may want to gather the following tools:

LOW WATTAGE SOLDERING IRON
ROSIN CORE SOLDER
SMALL WIRE CUTTERS
NEEDLE NOSE PLIERS
LEAD BENDING JIG
I C INSERTION TOOL

Note that the side of the board with the silkscreen outline is the component side and the other side is the circuit side. All components should be mounted on the component side.

2. ( ) Parts designation Rxx

All resistor leads are spaced on .4 inch centers. Insert the resistors into the circuit board, bending each lead over on the circuit side to prevent them from falling out. Clip each lead and solder it on the circuit side as you go along.

3. ( ) Parts designation RNx

RN1 has a dot printed on the pin 1 end. Match the dot with that on the silkscreen. Solder the resistor network on the circuit side of the board.

4. ( ) Parts designation Cxx

Monolithic capacitors have radial leads and can be inserted without forming the leads. Electrolytic capacitors have axial leads and must be formed. Bend the leads so that the printed value and polarity arrow can be seen from the top. The arrow points toward the minus (-) end . This polarity must be followed when installing each cap on the board. The plus (+) end of the electrolytic cap is marked on the circuit board. Insert, bend, solder and clip a few capacitors at a time. Remember to check your work as you go.

5. ( ) Parts designation SKxx

All the IC sockets have a notch in one end to indicate the pin #1 end of the socket. Check the circuit board and match the notches when you insert the sockets into the board. Verify that all the socket leads protrude through the board before soldering.

NOTE: U17 is a 24 PIN ZIF socket. Solder a 24 pin IC socket at this location. The ZIF socket will be plugged into the IC socket and removed if necessary for replacement or relocation to the front panel board if used.

6. ( ) Parts designation Px

Insert the potentiometer into the boards component side and solder the three pins on the circuit side.

7. ( ) Parts designation Y1

Bend the two leads of the crystal away from the printed side and insert it into the board with the printed side up. Solder and clip the leads of the crystal on the circuit side of the board.

8. ( ) Parts designation Dx

Insert the LEDs making sure that the cathode (lead at the same side as the flat spot on the LED's perimeter) matches the stripe on the board. Insert the LEDs all the way to the surface of the board, solder and clip the leads. Recheck the diode for proper orientation.

9. ( ) Parts designation Qx

Insert the transistors into their proper positions. The center lead is bent toward the flat side with the identification markings. Be sure to orientate the transistor correctly before soldering. Solder and clip the leads. Recheck the orientation of the transistors before moving on.

10. ( ) Parts designation PBx

Insert the three push buttons into the component side of the board. Solder the leads on the circuit side of the board.

11. ( ) Parts designation SWx

Insert the large POWER switch at location SW2 with the handle extending off the board. Insert the DIP switch with position 1 at the notched end of the SW1 silkscreen rectangle. (4 positions are needed but the board is laid out for a switch of up to 10 positions). Solder the switches on the circuit side of the board.

12. ( ) Parts designation Jx

Insert the DB-25s (J1) into the component side of the board and solder the leads on the circuit side of the board. Mount the 4 pin Molex power connector (J2) on the board. The locking tab (tall side) should be toward the edge of the board. Solder J2 on the circuit side of the board. Insert the shorter leads of the 1 x 14 pin connector (J3) into the board and solder it on the circuit side of the board. If batteries will be used, solder on a battery clip to J6, red lead to plus (+) and black lead to minus (-).

13. ( ) Parts designation VRx

Mount the 5 volt regulator bending the leads away from the printed side. A screw and nut will secure the regulator to the circuit board which will act as a heat sink. Solder the leads on the circuit side of the board.

14. ( ) Parts designation JPx

Insert the shorter leads of the 2 x 3 pin connector into JP1 and solder the leads on the circuit side of the board. Refer to Figure 4 for the shorting jumper placement. (Check U6 for EPROM size).

15. ( )

Check each component for proper value, placement, polarity (if applicable), and proper solder joints. If possible, clean the soldering flux off the board with a solvent (i. e. TRI-CLOR). Solvents like this one may affect plastics so try and keep them away from the component side of the board (especially the push buttons).

PRELIMINARY TEST

Before inserting the rest of the components, a few tests should be made to see if the unit is operating properly up to this point. Follow the test procedure step by step. If a test does not produce the proper results, go back and check your work. Errors most often occur in soldering, bent pins, improper component placement and/or orientation.

1. ( )

With no IC's inserted in the sockets, apply power to the 'IC Tester' through the power connector J1. The power source should have a minimum capacity of +5V @ 400mA. Six C-cells can be substituted for an external +5 volt supply. The on board regulator will drop 9 volts from the batteries, connected through J6, to +5 volts.

When power is applied, the LED marked 'POWER' should light up. Measure +5 volts on SK1 pin 20 (ground) to pin 40 (+5 volts). If not, turn off the power. Inspect your assembly for excess solder or shorts between traces or IC pins. Most frequently, solder (too much or too little) is the source of errors.

2. ( )

Next, turn off the power (always turn off the power when inserting or removing IC's during this testing phase) and insert all the IC's. Remember all the IC's are marked with a notch or dot indicating pin 1. Match this to the notch in the sockets and the silkscreen. Recheck that all ICs have been inserted. Only SK17 should be empty (Insert the ZIF socket into SK17). Turn on the POWER and press PB3. The LED marked STATUS should blink each time the reset button is pushed.

3a. ( ) Terminal Mode

Power off the 'IC Tester' and connect a terminal or host computer to it via J1. Refer to Figure 3 to set the DIP switches for 'Terminal Mode' at one of the four baud rates listed. Set your terminal for the same baud rate selected. Reapply power to the 'IC Tester'. A menu should appear on the terminal. Try going through some of the menu options.

3b. ( ) Stand-alone Mode

Power off the 'IC Tester' and attach the display (LCD1) to J3 (a mating connector to J3 is included with the display - it is to be soldered on the display only when the display is to be mounted on the main board). 1/2 inch standoffs will secure the display to the main board. Refer to Figure 3 and set the baud rate to 9600 (the 'IC Tester' sends messages out the serial port before going to the LCD, the LCD may remain blank for a few seconds at 300 baud). Reapply power to the 'IC Tester'. A message should appear on the LCD.

ASSEMBLING the IC Tester (Enclosed unit - front panel board)

1. ( ) The main board must be assembled and tested prior to the start of assembling the front panel board.

2. ( ) Parts designation Rxx

All resistor leads are spaced on .4 inch centers. Insert the resistors into the circuit board, bending each lead over on the circuit side to prevent them from falling out. Clip each lead and solder it on the circuit side as you go along.

3. ( ) Parts designation SKxx

All the IC sockets have a notch in one end to indicate the pin #1 end of the socket. Check the circuit board and match the notches when you insert the sockets into the board. Verify that all the socket leads protrude through the board before soldering.

NOTE: A 24 pin IC socket should be soldered into U1. A second 24 pin IC socket is inserted into the first. These 2 stacked-sockets will protrude through the front of the enclosure. The 24 pin ZIF socket is relocated from the main board into the stacked sockets, after the board has been mounted into the enclosure.

4. ( ) Parts designation Dx

Insert the LEDs making sure that the cathode (lead at the same side as the flat spot on the LED's perimeter) matches the stripe on the board. Insert the LEDs all the way to the surface of the board, solder and clip the leads. Recheck the diode for proper orientation.

5. ( ) Parts designation PBx

Insert the two push buttons into the component side of the board. Solder the leads on the circuit side of the board.

6. ( ) Parts designation Jx

Insert the shorter leads of the 1 x 14 pin connector (J3) into the board and solder it on the circuit side of the board. Insert the two right angle 2 x 10 connectors into J5 and J6 on the main board (the bent ends of the connectors go into the component side of the main board with the straight ends pointing off board). Solder the circuit side of the main board. Remove the ZIF socket from socket SK17. Slip the 'front panel board' onto the right angle connectors. The component side of the front panel board faces away from the main board. Push the 'front panel board' so that it is up against the socket SK17 on the 'main board'. When it is perpendicular to the 'main board' and parallel to the edge of the 'main board', solder the right angle connector on the COMPONENT SIDE of the 'front panel board.

7. ( )

Check each component for proper value, placement, polarity (if applicable), and proper solder joints.

8. ( ) Parts designation LCDx

Attach four 1/8 inch standoffs to the rear of the display using four screws inserted through the front of the display. Place the LCD on to J3. It should lay flush with the front panel board. Carefully solder the pins (J3) onto the display side of the LCD board. Clip the tops of the pins flush with the display side of the LCD board. Secure the display to the front panel circuit board with four nuts.

PRELIMINARY TEST

Follow the test procedure step by step. If a test does not produce the proper results, go back and check your work. Errors most often occur in soldering, bent pins, improper component placement and/or orientation.

1. ( )

Insert the ZIF socket into SK1a and SK1b on the front panel board - its orientation is opposite from that in SK17 on the main board. Refer to Figure 3 and set the baud rate to 9600 (the 'IC Tester' sends messages out the serial port before going to the LCD, the LCD may remain blank for a few seconds at 300 baud). Apply power to the 'IC Tester'. A message should appear on the LCD.

2. ( )

Place the rear panel onto the DB-25s. and secure the panel to the DB-25s connector with jack screws. Place the front panel over the front panel board (you will have to remove the ZIF socket from SK1 to do this).

3. ( )

Attach the power supply (if using it) and feed the cord out through the hole in the corner of the read panel. Place the board into the bottom of the enclosure (the front and rear panels should slide into the grooves on the front and rear of the enclosure).

4. ( )

Screw the main board to the bottom of the enclosure. Mount the battery holder (if using) to the bottom of the enclosure. Slide the top of the enclosure onto the front and rear panels (remember the grooves). Screws will hold the top and the bottom of the enclosure together. Replace the ZIF socket and add rubber feet and your done.

*** NOTE ***

If you are unable to make your IC Tester function correctly and lack the test equipment or time to troubleshoot the problem, CCI offers, for a fixed fee, a test and repair service. If you require this service, acquire an RMA. Then send your IC Tester, return shipping address, a phone number where you can be reached during the day and a check or money order (Visa & MasterCard are also accepted) for $50.00 plus $5.00 for return shipping (total $55.00)to:

CCICT REPAIR DEPT. 3
CCI.
Suite 12
4 Park St.
Vernon, CT. 06066

A Return Merchandise Authorization (RMA) number must be obtained before sending anything back to CCI. To obtain a number write to the above address or call 203-875-2751. Your IC Tester will be repaired and returned to you as soon as possible. See the warranty statement at the beginning of the manual regarding other aspects of this repair policy.

APPENDIX A -- Notes On Using 4000-series CMOS Devices

Some devices have characteristics that can make them incompatible with the standard IC Tester configuration; the 4000-series CMOS devices are the most notable examples. These devices have very low output current source/sink capability, causing some of the devices in this family to fail testing when the standard 390 ohm load resistors are used. The only devices in this family currently known to do this are listed in the table.

4017, 4021, 4022, 4023,
4028, 4040, 4042, 4060,
4526
Table listing the 4000-series devices not reliably testable using the standard 390 ohm load resistors.

If the tester is to be used exclusively for testing such exceptional devices, the value of the load resistors can be changed to better match the family being tested. In the case of the 4000-series CMOS devices, a load resistor value of about 1000 ohms seems to be about optimum. This value passes all currently supported 4000-series devices (as well as all 74HC- and 74HCT-series devices) and also seems to allow most 74LS-series TTL devices to pass as well.

Similarly, if device testing is restricted to devices with high input currents, such as 74S-series and 74H-series TTL devices, a lower load resistor value -- perhaps around 240 or 270 ohms --would be more appropriate. This would allow the few devices with exceptionally high input currents (such as the 74S113) to be tested on the IC Tester.

For most devices, the standard 390 ohm load resistor value works well.

APPENDIX B -- VECCPT Error Messages

INVALID START CHAR
The first character of the line is not a valid first character (#, *, S, F, P, I, R, E, or C).

LINE OUT OF PLACE
The line type -- determined by the first character of the line --though valid, is not an expected line type, indicating that the line does not appear in its appropriate place in line sequence.

This error also occurs for all device definition module lines following an errored line, until a new module definition line is found.

SYNTAX
A "general" error message used to indicate improper syntax in the specified line. Common causes are extraneous characters, and spaces in device names.

"END" STATEMENT MISSING -- WARNING
This error indicates that the start of a new device definition module was detected without an "END" (E) line being found for the previous device definition module. The syntax of the previous device definition module was, however, correct, and at least one complete test vector was specified. The previous device definition module was automatically closed, as if an END statement had been included.

UNEXPECTED START OF NEW DEVICE
This error indicates that the start of a new device definition module was detected while the previous device definition module was being processed. The state of the previous module was not complete, at a point where it could be closed as if an END statement were processed.

TOO MANY/FEW PARAMETERS
Too many or two few parameters were included in the line. For example, the Set-up (S) line requires three parameters (number of pins, ground pin, and power pin); if other than three parameters are found, this error is generated.

COLUMN ALIGNMENT ERROR
Column alignment is determined by the pin function (F) line. This error is generated if numbers or identifiers on subsequent lines are not aligned under the columns defined by the pin function line. This error is sometimes elusive because of 'tabs' inadvertently included in the source file.

DUPLICATE PIN NUMBERS

The same pin number was used more than once in the pin number (P) line.

NO VECTORS PROCESSED

The syntax and sequence of the device definition module was correct, but the END line was found with any test vectors having been defined.

OUT OF ARRAY MEMORY

The 'veccpt' vector compiler uses special arrays to store the information from the processed device definition modules. This error indicates that the amount of memory allocated for veccpt's arrays has been exceeded, and no further devices can be processed.

DUPLICATE DEVICE NAME

The name specified in the Device Name line has already been defined in a previously-processed device definition module.

DEVICE CLONE NOT FOUND

The "master" device name specified in the clone (C) line was not found in any of the Device Type arrays. Clone masters must always be defined before related clones can be processed.

APPENDIX C -- IC Tester Parts List (main board)

INTEGRATED CIRCUITS

U1	8031 8-BIT MICROCOMPUTER
U2	MAX232 5V RS232 DUAL RECEIVER/TRANSMITTER
U4	74LS244 - LS TTL OCTAL BUFFER
U5	74LS373 - LS TTL OCTAL TRANSPARENT LATCH
U6	EPROM 27256 (Programmed)
U7	74LS139 - LS TTL DUAL 2 TO 4 DECODER
U8 - U10	74LS374 - LS TTL OCTAL FLIP-FLOP
U11 - U16	74HCT244 - TTL COMPATIBLE HS-CMOS OCTAL BUFFER
U17	24 PIN ZIF SOCKET
U19	74HCT374 - TTL COMPATIBLE HS-CMOS OCTAL FLIP-FLOP

RESISTORS

R1,R18 - R25, R34 - R41,R49 - R55	RESISTOR,4.7K,1/4W,5%
R2	RESISTOR,330,1/4W,5%
R3	RESISTOR,10K,1/4W,5%
R4 - ~~R8~~ R9	RESISTOR,680 OHM,1/4W,5%
~~R9~~	RESISTOR,620 OHM,1/4W,5% → NOT IN KIT THEY REPLACED WITH 680Ω
R10 - R17, R26 - R33, R42 - R48, R56	RESISTOR,390 OHM,1/4W,5%
RN1	SIP RESISTOR,4.7K,10 PIN SIP,
P1	POTENTIOMETER, 10K

CAPACITOR

C1	CAPACITOR,10 UFD,25V,
C2,C3	CAPACITOR,27 PF,
C4,C20 - C23	CAPACITOR,47 UFD,25V,
C5 - C19	CAPACITOR,.1 UFD,
C24	CAPACITOR,.001 UFD,

SEMICONDUCTORS

VR1	LM7805C - VOLTAGE REGULATOR, +5VDC
D1,D2	LIGHT EMITTING DIODE, TIL 220
Q1,Q2	NPN TRANSISTOR, PN2222
Q3 - Q6	PNP TRANSISTOR, PN2907

IC SOCKETS

SK1	40 PIN IC SOCKET
SK2,SK7	16 PIN IC SOCKET
SK4 - SK16, SK19	20 PIN IC SOCKET
SK6	28 PIN IC SOCKET

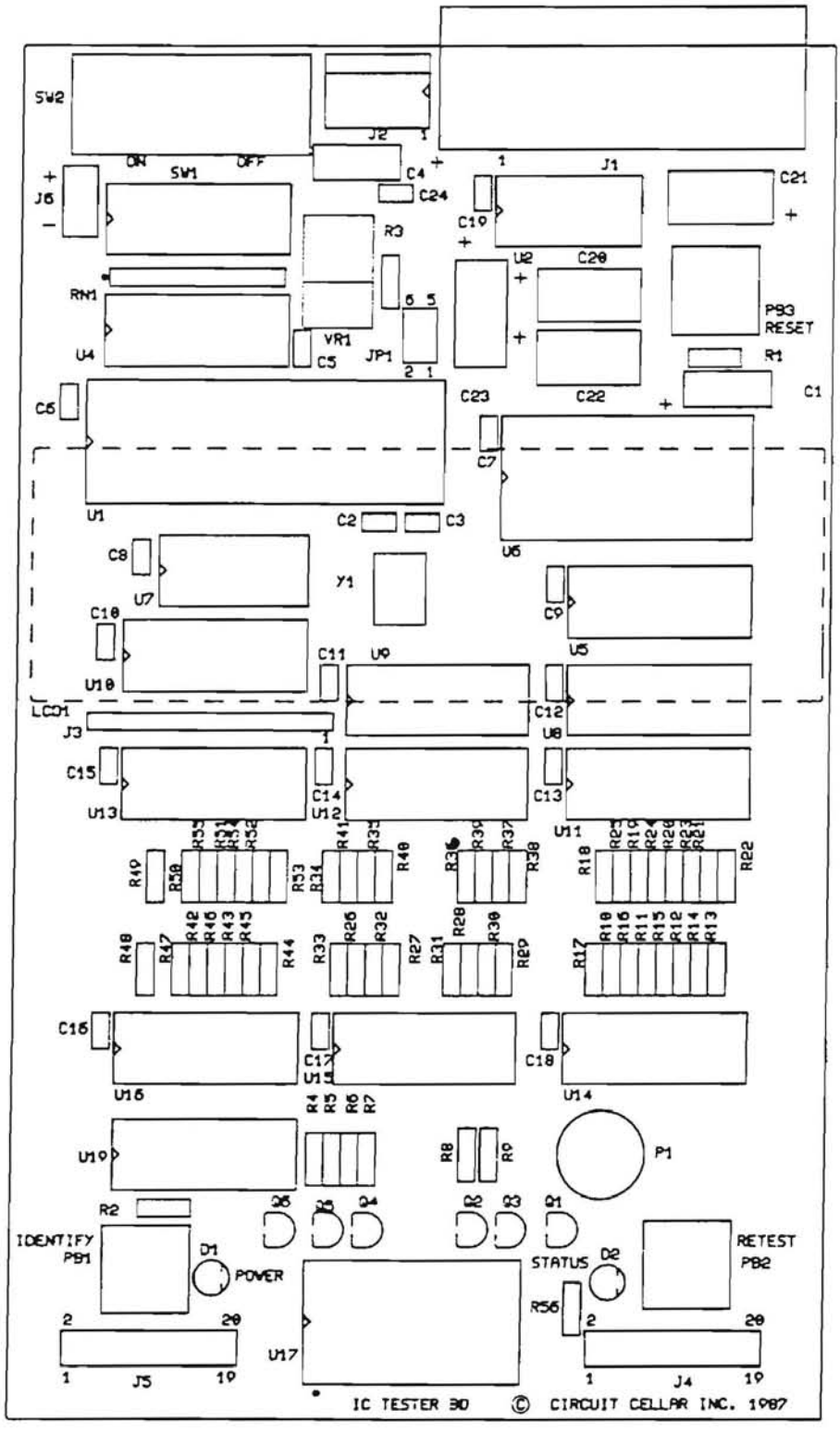
CONNECTORS

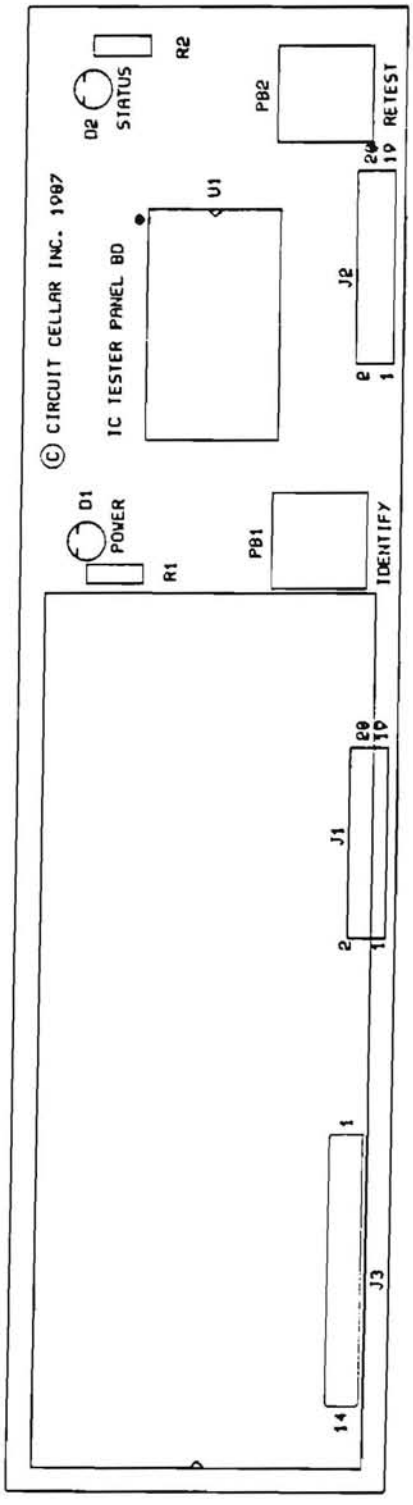
J1	SUBMINATURE DB-25S
J2	4 PIN MOLEX (Supplied with mating connector)
J3	1X14 SQUARE PIN HEADER
J6	1X2 SOLDER PADS (for user supplied batteries)
JP1	2X3 SQAURE PIN HEADER

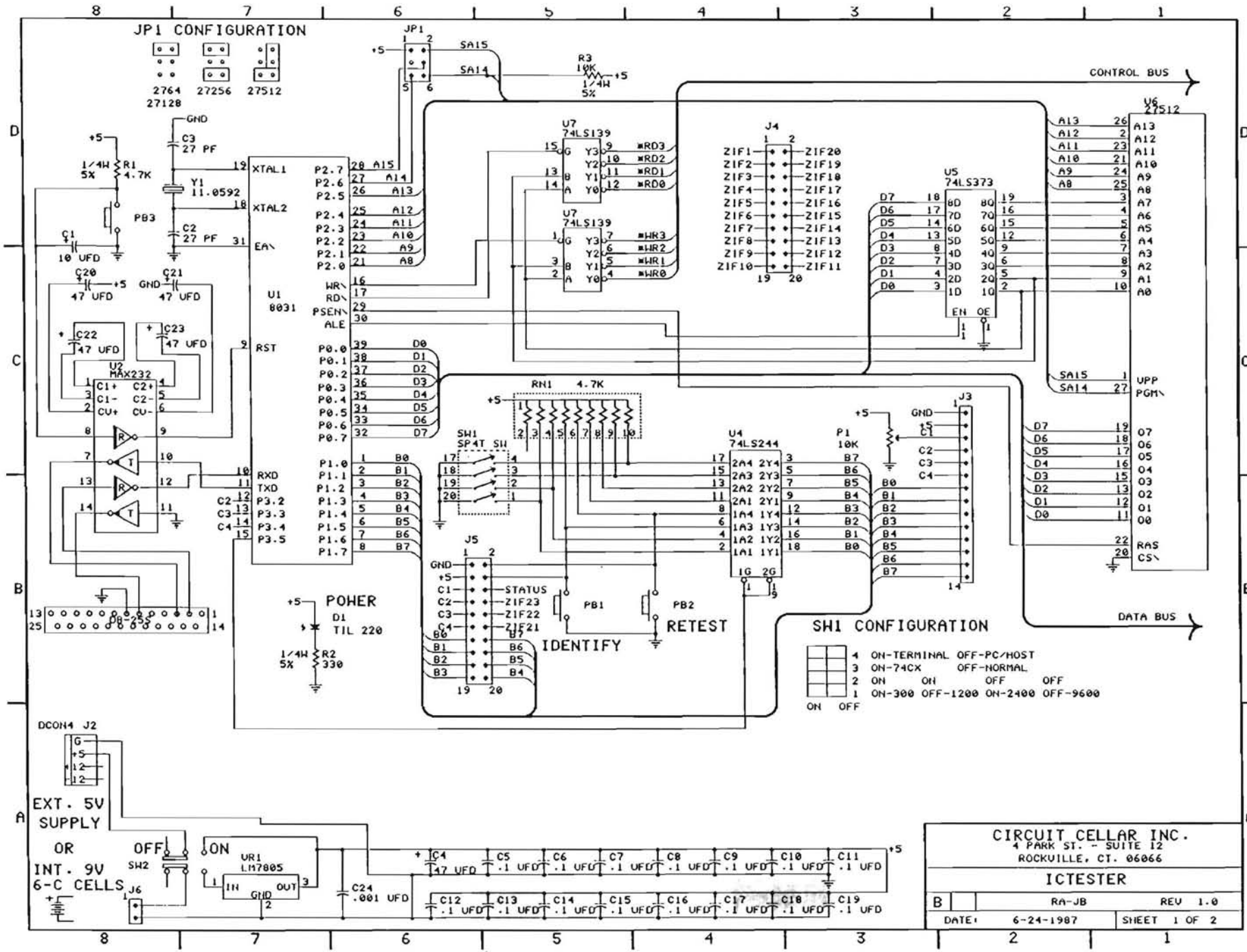
MISCELLANEOUS

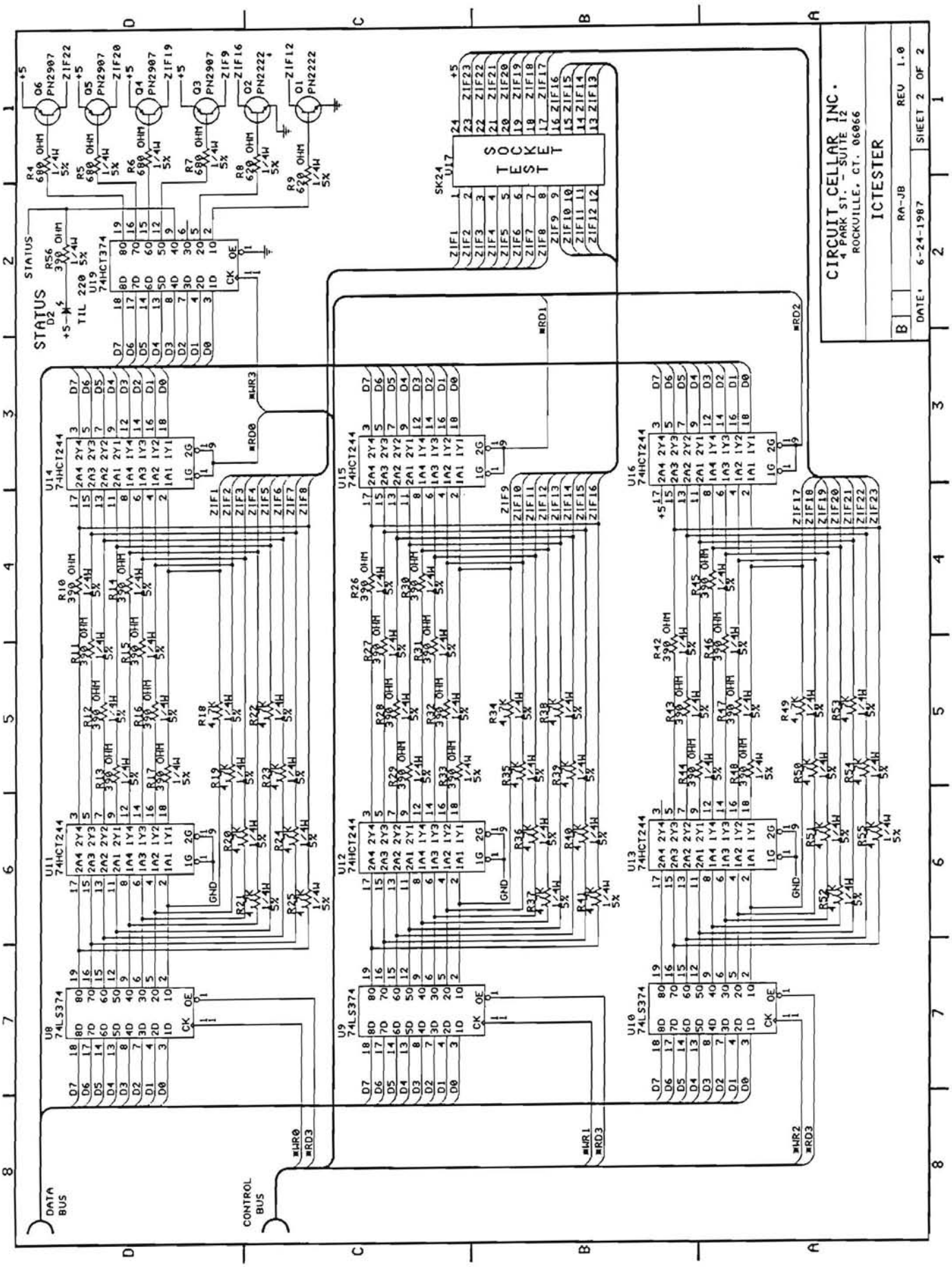
PB1,PB2,PB3	SWITCH,PUSH BUTTON
SW1	DIP SWITCH,SP4T
SW2	SLIDE SWITCH, DPDT
Y1	CRYSTAL, 11.0592 Mhz
PCB1	ICTESTER CIRCUIT BOARD
SJ1, SJ2	SHORTING JUMPERS

APPENDIX D -- IC Tester Component Layout (silkscreens)



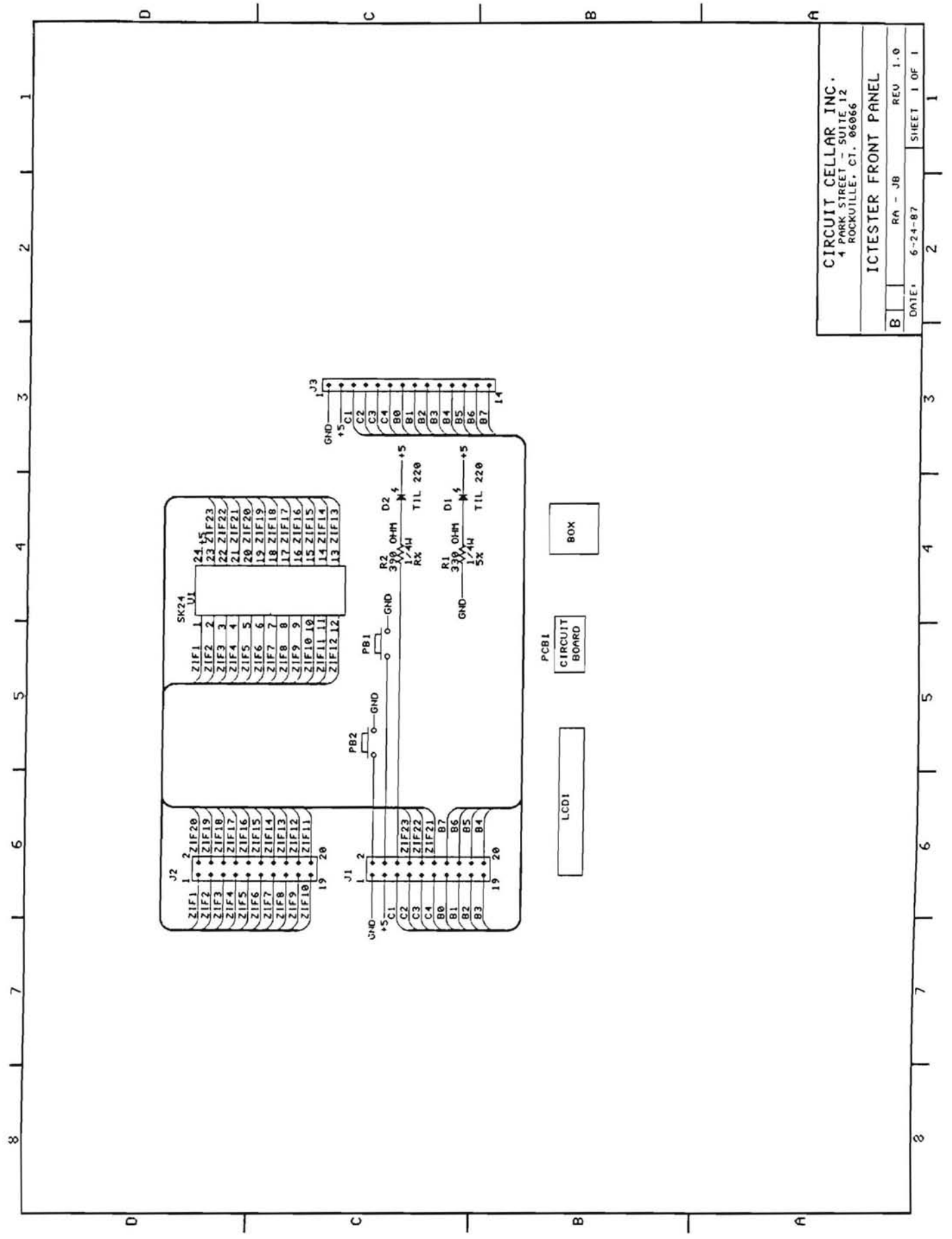



JP1 CONFIGURATION
2764
27128
27256
27512
JP1
SA15
SA14
R3
10K
1/4W
5%
CONTROL BUS
U6
27512
U7
74LS139
U5
74LS373
J4
U1
8031
U2
MAX232
RN1 4.7K
SW1
SP4T SW
U4
74LS244
P1
10K
J3
J5
POWER
D1
TIL 220
R2
330
PB1
IDENTIFY
PB2
RETEST
STATUS
ZIF23
ZIF22
ZIF21
SW1 CONFIGURATION
4 ON-TERMINAL OFF-PC/HOST
3 ON-74CX OFF-NORMAL
2 ON ON OFF OFF
1 ON-300 OFF-1200 ON-2400 OFF-9600
ON OFF
DATA BUS
DCON4 J2
EXT. 5V
SUPPLY
OR
INT. 9V
6-C CELLS
J6
SW2
VR1
LM7805
C24
.001 UFD
CIRCUIT CELLAR INC.
4 PARK ST. - SUITE 12
ROCKVILLE, CT. 06066
ICTESTER
RA-JB
REV 1.0
DATE: 6-24-1987
SHEET 1 OF 2


CIRCUIT CELLAR INC.
4 PARK ST. - SUITE 12
ROCKVILLE, CT. 06066
ICTESTER
RA-JB
REV 1.0
DATE: 6-24-1987
SHEET 2 OF 2
STATUS
STATUS
TIL 220
SOCKET
DATA BUS
CONTROL BUS
74HCT244
74LS374
PN2907
PN2222
390 OHM
4.7K

APPENDIX F -- IC Tester Parts List (front panel)

REF. DES.	DESCRIPTION
RESISTORS	
R1	RESISTOR,330 OHM,1/4W,5%
R2	RESISTOR,390 OHM,1/4W,5%
SEMICONDUCTOR	
D1,D2	LIGHT EMITTING DIODE, TIL 220
CONNECTORS	
J1,J2	2X10 RIGHT ANGLE SQUARE PIN HEADER
J3	1X14 SQUARE PIN HEADER
SCKETS	
SK1A,SK1B	24 PIN IC SOCKET
MISCELLANEOUS	
LCD1	2X20 CHARACTER,LIQUID CRYSTAL DISPLAY,
BOX	ENCLOSURE,PACTEC CF-225 W/HARDWARE
PB1,PB2	SWITCH,PUSH BUTTON
PCB1	IC TESTER FRONT PANEL CIRCUIT BOARD
AP1, AP2	PRE-PUNCHED ALUMINUM PANEL
LP1, LP2	PRINTED LABEL


CIRCUIT CELLAR INC.
4 PARK STREET, SUITE 12
ROCKVILLE, CT. 06066
ICTESTER FRONT PANEL
B
RA - JB
REV 1.0
DATE: 6-24-87
SHEET 1 OF 1
J3
GND
+5
C1
C2
C3
C4
B0
B1
B2
B3
B4
B5
B6
B7
SK24
U1
ZIF1
ZIF2
ZIF3
ZIF4
ZIF5
ZIF6
ZIF7
ZIF8
ZIF9
ZIF10
ZIF11
ZIF12
ZIF13
ZIF14
ZIF15
ZIF16
ZIF17
ZIF18
ZIF19
ZIF20
ZIF21
ZIF22
ZIF23
PB1
PB2
R1
330 OHM
1/4W
5%
R2
330 OHM
1/4W
5%
D1
D2
TIL 220
PCB1
CIRCUIT BOARD
BOX
LCD1
J1
J2

APPENDIX H -- Standard Device Library Version 1.0

4000	4001	4002	4006	4011	4012	4013	4017	4018	4019
4020	4021*	4022*	4023*	4024	4025	4027	4028*	4030	4035
4040*	4041	4042*	4043	4044	4051	4052	4053	4060*	4066
4069	4076	4077	4081	4085	4086	4093	4094	4511	4516
4518	4522	4526*	4532	4585					

7400	7401	7402	7403	7404	7405	7406	7407	7408
7409	7410	7411	7412	7413	7414	7415	7416	7417
7420	7421	7422	7424	7425	7426	7427	7428	7430
7432	7433	7434	7435	7437	7438	7440	7442	7443
7444	7445	7446	7447	7448	7449	74LS51	74LS54	7454
74LS55	7460	7464	7465	7468	7470	7472	7474	7475
7485	7486	74L95	7495	7496				

74104	74105	74107	74109	74112	74113	74114	74125
74126	74128	74131	74132	74133	74136	74137	74138
74139	74145	74147	74148	74150	74151	74153	74154
74155	74156	74157	74158	74160	74161	74162	74163
74164	74165	74166	74168	74169	74170	74173	74174
74175	74176	74177	74178	74180	74183	74184	74185
74190	74191	74192	74193	74194	74195	74196	74197
74198	74199	74230	74231	74240	74241	74242	74243
74244	74245	74246	74247	74248	74249	74251	74253
74257	74258	74259	74260	74261	74265	74266	74273
74276	74279	74280	74283	74286	74290	74293	74295
74298	74299	74323	74348	74352	74353	74354	74355
74356	74365	74366	74367	74368	74373	74374	74375
74377	74378	74379	74386	74390	74393	74395	74465
74466	74468	74490	74533	74534	74540	74541	74573
74574	74590	74595	74597	74623	74640	74641	74642
74643	74644	74645	74670	74688			

* See Appendix A

The above list is the ROM Resident device library of "generic" designations. Testing any IC not on this list will result in a "not identified" display from the test.