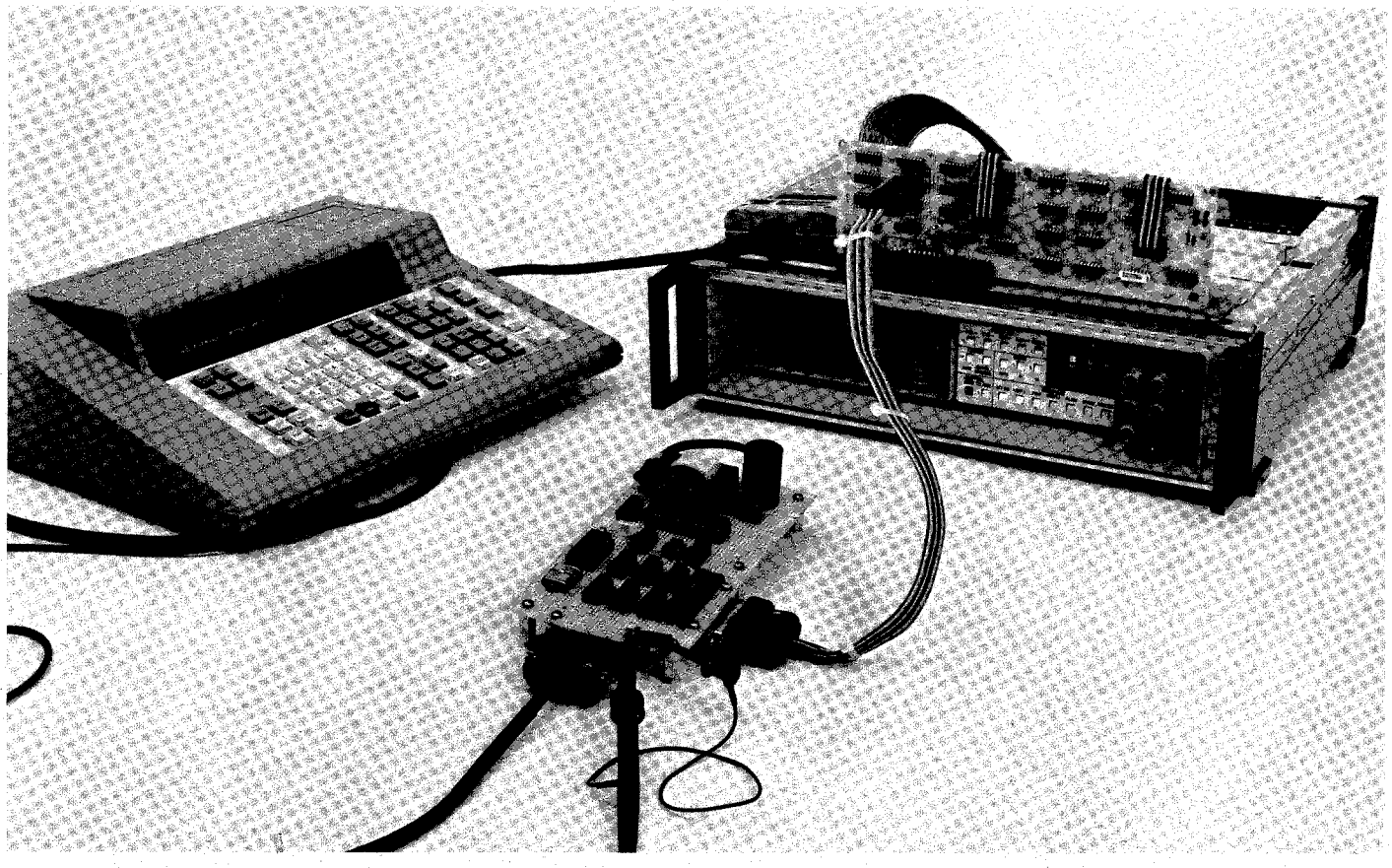
FLUKE
TROUBLESHOOTER
VOL.2 NO.1 FOR MICRO-SYSTEM TROUBLESHOOTER USERS



A scanner for multipoint probing

by Ed Ferguson

I use the 9010A with Guided Fault Isolation programs to troubleshoot products with extensive amounts of I/O circuitry that include A/D converters, DACs, and parallel output ports. Once the microprocessor kernel is tested, I rely on the 9010A probe to test the I/O.

Frequently a large number of pins must be probed to test an IC. It is time-consuming to locate the pin and hold the probe for the duration of the test. Also, mistakes occur when probing the wrong pin or the probe slips off a pin in the middle of a test.

A solution to these problems is to use an IC clip in conjunction with a switching scheme to route one pin at a time, under program control, to the 9010A probe. Now the operator can place an IC clip over a device and gather the readings on the pins. When finished, the clip is moved to the next device. A suitable connector can be substituted for the IC clip to test parallel ports or board edge connectors.

The scanner described here is an electronic switch that will multiplex up to 24 inputs to the 9010A's probe. The scanner acts under command from the 9010A's optional RS-232 interface to select a given input.

(see "Scanner" inside)

Broken probe?

Q: I have just broken the tip off of my 9010A probe. How do I go about replacing it?

A: The only way to replace the metal tip on the probe is by replacing the probe body. The part number to order is PN 611814.

When you receive the new plastic probe body, unscrew the rubber strain relief from the old probe body and lift off the probe canopy (the cover with the colored lenses). At the tip end of the circut board is a large solder pad that connects the probe tip to the circuit board. Unsolder this connection and remove the old probe body from the circuit board. Then resolder the new probe body to the circuit board and reassemble the probe.

Built-in test failure detection

by Jim Clodfelter

The 9010A Micro-System Troubleshooter has several built-in tests that include the Bus test, ROM test, RAM short, RAM long and Auto test. When an error is discovered during one of these tests, the 9010A stops the test and displays an error message indicating the fault type. If these built-in tests reside in a program, the error reporting is still the same: Program execution is halted.

Many customers have asked if there's a way to branch out of the test routine when an error is detected, either to a troubleshooting routine or to indicate the presence of errors without halting program execution. For those customers with the RS-232 interface option, using the set-up commands makes this feat possible.

The following technique was developed by Leo Longo, our Field Systems Engineer in the Tustin office. The set-up command "SET EXERCISE ERRORS? YES" is the command used by the 9010A to halt built-in tests and display the error detected. If the set-up condition is set to "NO", when the error condition is recognized, the test does not stop but the message that would normally be displayed is sent out the RS-232 port on pin 2.

This same action will again be true if the built-in tests are used within a program. Pin 2 is the transmit line for the RS-232 signal and pin 3 is the receive line. What we would like to have is some way for the 9010A to monitor the RS-232 output so that it would know when an error occurred. This can be done by tying the output pin (pin 2) to the input pin (pin 3). Now, whenever an error message is sent out pin 2, a stream of corresponding data would enter pin 3. The activity of data passing from the output to input could be monitored through the status of the RS-232 interface. Figure 1 illustrates the bit assignment for the RS-232 status.

We will monitor the condition of Bit 3 in the status register to check for test errors. If a built-in test fails, the error message will go out pin 2 and re-enter pin 3. This will set Bit 3 to a one (1) since characters were received. If no failure occurred, no input to pin 3, Bit 3 would remain a zero.

To implement this technique, take an RS-232 plug/adapter and tie pins 2 and 3 together, then plug it into the RS-232 port on the back of the 9010A. Set the set-up condition "EXERCISE ERRORS?" to "No". Now design a program that checks the RS-232 status register for a character received condition after each Built-In test. If Bit 3 is high, the test failed because a character was received. The program can then branch to an error-handling routine.

Some special AUX I/F symbols are used in the programming mode to manipulate the RS-232 interface. These can be found in the programming manual or reference card.

Slash (/)—When followed by a hexadecimal digit, places the next byte of data from the RS-232 interface in a register designated by the hexadecimal digit. If data from the RS-232 interface is not present, program execution is suspended until data is present.

Backslash (\)—When followed by a hexadecimal digit, places the status of the RS-232 interface in the lower five bits of the register designated by the hexadecimal digit.

Plus sign (+)—When the last character in an AUX I/F step, prevents a line termination sequence from being sent at the end of the line.

You can use all or parts of this program to branch on individual built-in tests. This routine can be used as a tool by the operator of the 9010A Micro-System Troubleshooter to enhance troubleshooting speed and capabilities, by going directly to a fault isolation program whenever the built-in test finds an error.

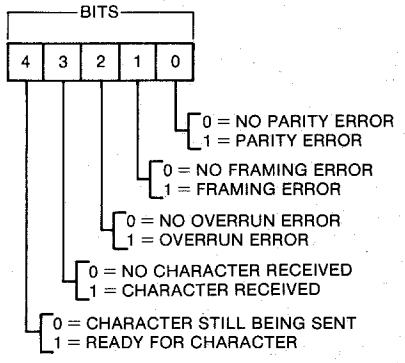


Figure 1. Format of RS-232 Status

```
AUX\1+                      Read the RS-232 status register.
If Reg 1 AND 8 = 0 GOTO 1   We are looking to see if Bit 3 was set high from some
                            previous condition. If it is zero, proceed with your test.
AUX/1+                      Places next byte from RS-232 in register and resets
                            Bit 3.
LABEL 1
Bus Test                    Now perform the desired built-in test.
AUX\1+                      Read the RS-232 status register.
If Reg 1 AND 8 > 0 GOTO A   If Bit 3 is high, we detected an error. Jump to a sub-
                            routine that will perform a Guided Fault Isolation (GFI)
                            routine that tests Bus failures.
RAM Short Test              Perform the next built-in test.
AUX\1+                      Read RS-232 status register.
If Reg 1 AND 8 > 0 GOTO B   If Bit 3 is high, the RAM short test failed. Label B would
                            be a GFI for the Ram area.
ROM Test                    Perform the next built-in test.
AUX\1+                      Read RS-232 status register.
If Reg 1 AND 8 > 0 GOTO C   If Bit 3 is high, the ROM test failed. Label C is a GFI for the
                            ROM area.
I/O Test                    Perform the next built-in test.
AUX\1+                      Read RS-232 status register.
If Reg 1 AND 8 > 0 GOTO D   If Bit 3 is high, the I/O test failed. Label D is a GFI for
                            the I/O area.
GOTO F                      Label F would be the end of your GFI routine.
LABEL F                     End of Program
```

Listing 1. Branch on Error Routine

Scanner...

(continued from cover)

As shown in figure 1, the scanner consists of a serial interface, analog multiplexers, and voltage dividers. RS-232 hex data of 20 thru 37 correspond to channels 1 thru 24. U1, U2, and U3 form the 4800 baud serial link with the 9010A. The lower five bits from the UART U3 are latched into U5 and U6. The NOR gates connected to bits 5 and 6 prevent data below hex 20 (all machine commands) from latching. Latched bits 3 and 4 select 1 of the 3 multiplexers (U9-U11) via the 1 of 10 decoder U7. Latched bits 0 thru 2 select 1 of 8 inputs on the enabled multiplexer. The multiplexers are CMOS analog switches that are essentially 120 ohm resistors when closed. The multiplexers are operating at 12 volts for minimum distortion of the input signals. U8 provides the TTL 5-volt to CMOS 12-volt interface. The high impedance voltage dividers on each channel bias the input to a tristate level when the input is open, and prevent open channels from "following" active pins. The optional LED's provide a binary indication of the selected channel (0-23).

Construction of the scanner is straightforward. Note that a regulated 5-volt and 12-volt power supply are required. Include a .22 μF capacitor from VCC to ground on each IC. The 24 input lines are connected to a DB-25 connector to allow various types of test connectors to be attached. The test cable is an 18-inch long pair of ribbon cables with alternate lines tied to ground. The UUT end of the cable is connected to an IC clip or a board edge connector as required for your application. Unused channels may be left open.

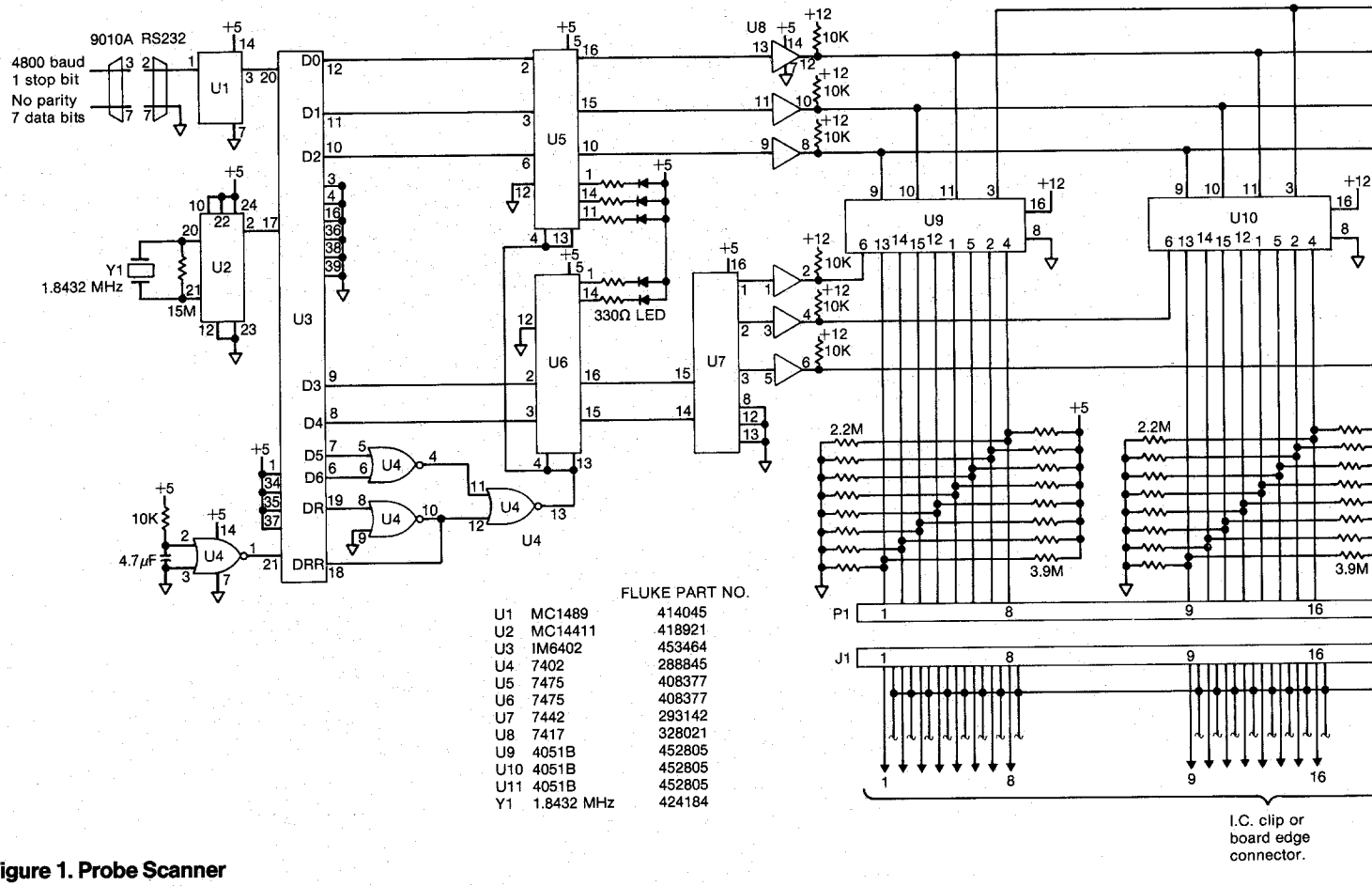
There are two restrictions when using the scanner. First, the inputs are not protected and therefore must not exceed 5 volts. Second the 9010A probe pulser may be routed to a selected channel as the multiplexers are bi-directional. However, the pulser may only be used to drive open inputs since the overdrive capability is lost.

The 9010A program that controls the scanner performs the following steps:

- Instruct the operator to clip the IC.
- Switch a pin to the probe via the scanner.
- Supply a suitable stimulus to exercise the circuit node.
- Measure the probe data.
- Compare the actual probe data to the expected data.
- Repeat for the remaining pins.
- Move the clip to the next device.

A 16-pin IC can be tested in 15 seconds using this approach. Refer to fig. 2 for a sample program.

An alternate approach is to connect both the 9010A and the scanner to a personal computer (PC) equipped with two RS-232 ports. Now the PC can control the 9010A and the scanner, as well as store large amounts of data. This increased capability allows us to "learn" as well as test the IC. This subject will be discussed in a following issue.



		FLUKE PART NO.
U1	MC1489	414045
U2	MC14411	418921
U3	IM6402	453464
U4	7402	288845
U5	7475	408377
U6	7475	408377
U7	7442	293142
U8	7417	328021
U9	4051B	452805
U10	4051B	452805
U11	4051B	452805
Y1	1.8432 MHz	424184

Figure 1. Probe Scanner

9010A application support

The following individuals and organizations have indicated their capability of, and interest in, providing independent 9010A support. The services offered are shown with each name.

Mr. Ali Mosleh
Computer Sciences Corp.
8300 Merrifield Ave.
Fairfax, VA 22031
(703) 560-5051 (office)
(703) 560-1316 (home)
Contract programming and troubleshooting.

Mr. Gary Aiken (Eng. Manager)
Diversified Data Corp.
6551 Loisdale Court
Springfield, VA 22150
(703) 922-9444
Contract programming, engineering support, integrated logistics support, documentation, and training.

Mr. Allan Cody
Electronics Corp. of America
1 Memorial Dr.
Cambridge, MA 02142
(617) 787-5980
Contract programming, testing, and troubleshooting.

Mr. Thomas Bielecki
EMF Inc.
60 Foundry St.
Keene, NH 03431
(603) 352-8400
Contract programming and testing.

Mr. Dick Thomas
General Electric Co.
Instrument & Computer Equipment Repair Service
5096 Peachtree Rd.
Chamblee, GA 30341
(404) 452-4905
Contract programming, testing, and troubleshooting.

Mr. Quint Pierson
General Electric Co.
Schenectady Instrument Service
Bldg. 28, Rm. 503
Schenectady, NY 12345
(518) 385-5107
Contract programming, testing, and troubleshooting.

Mr. Julio Cordova
High-Technology Services
1301 W. Copans Rd., Bldg. F
Pompano Beach, FL 33064
(305) 973-4949
Contract programming, testing, and troubleshooting.

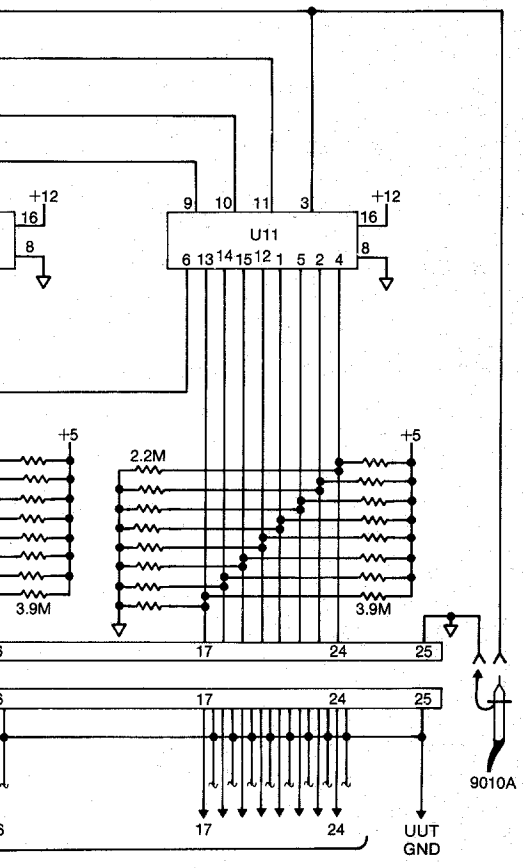
Mr. Mike Pearson
Mike Pearson & Associates
2013 Tiehick Lane
Garland, TX 75234
(214) 495-4510
Contract programming, testing, and troubleshooting.

Mr. John Schira
Quinton Instruments
2121 Terry Ave.
Seattle, WA 98121
(206) 223-7373
Contract programming.

Mr. A. Gallagher (System Eng.)
Wismer & Becker Engineers
7820 Folsom Blvd.
Sacramento, CA 95806
(916) 381-3930, Ext. 475
Contract programming, testing, and troubleshooting.

Mr. Dennis D. Norwood
O'Conner Distributing Co. Inc.
9030 Directors Row
Dallas, TX 75247
800-527-2432 Outside of Texas
800-442-6586 Texas
Contract programming, testing, and troubleshooting video games.

If you would like your name added to the list, please let us know.




```
PROGRAM 1                       !test program - IC 12
DPY-CLIP IC 12 - PRESS CONT#
STOP                            !wait for "CONT"
SYNC ADDRESS                    !sync probe to address
REG8 = C                        !IC 12
REG9 = 1                        !pin 1
REGA = BB34                     !expected signature, pin 1
EXECUTE PROGRAM 10              !scanner program
REG9 = 2                        !pin 2
REGA = 96EC                     !expected signature, pin 2
EXECUTE PROGRAM 10              !scanner program
REG9 = 3                        !pin 3
REGA = 725B                     !expected signature, pin 3
EXECUTE PROGRAM 10              !scanner program
*
*
*                               !continue program for remaining pins

PROGRAM 10                      !scanner program
REG1 = REG 9 OR 20 DEC          !convert IC pin # to scanner channel
AUX-% 1                         !send channel # out RS-232
READ PROBE                      !clear probe
RAMP @ FFFF                     !stimulus for IC
READ PROBE                      !read probe data
REG2 = REG0 SHR SHR SHR
       SHR SHR SHR SHR SHR
       and FFFF                 !probe signature
DPY-IC@8-@9 SIG $A=$2           !IC #, expected & actual signature
IF REGA = REG2 GOTO 1           !signature compare; pass
DPY-+ FAIL#                     !signatures do not compare; fail
STOP                            !wait for "CONT"
LABEL 1                         !return to test program
```

Figure 2. Sample Scanner Program

Solving noise and timing problems

Have you ever plugged your 9000-Series Troubleshooter into an unfamiliar Unit Under Test (UUT) and had difficulty in getting it to run? Maybe BUS TEST passes, but RAM SHORT fails. When you try a WRITE followed by a READ, you get erratic results. You've used the Troubleshooter enough to have confidence in it. It's bailed you out of some tough troubleshooting situations before, but now you're stuck. What's wrong?

When problems getting a UUT to work well with the Troubleshooter occur, they're usually caused by system noise and/or marginal timing.

System Noise

System noise is always present. It's not new; it's always been there. However, it becomes more and more of a problem as system speeds increase, because the sharp-edged fast data pulses generate plenty of noise. This noise is dealt with by system designers enough to get the UUT to operate under expected operating conditions. But when the Troubleshooter Interface Pod is plugged into a UUT, two things happen.

First, the UUT is opened up to the outside world. The integrity of the UUT's own shielding is violated if the UUT cannot be closed up with the Interface Pod connected to it. This may expose the UUT to noise from outside equipment or from other portions of its own circuitry. This situation doesn't usually cause problems, but if it does, it must be dealt with creatively with each different type of UUT.

The second thing that happens is several inches of cable are added to the system bus. Even if you are able to keep this added cable inside of the UUT's shielded environment, it may cause internal system noise to be picked up on the μP lines. (The new strict FCC requirements deal primarily with noise that is emitted into the environment outside of the UUT. For obvious reasons, the FCC requirements do not restrict noise that is contained within the UUT.) The best solution to this problem is to be sure that your Interface Pod uses a shielded UUT cable (the cable that plugs into the μP socket).

Marginal Timing

If system noise doesn't seem to be causing your problem, then you may be troubled by marginal timing. Marginal timing occurs when the system is designed close to the timing requirements of the various devices in the UUT (usually in order to maximize the performance of the UUT). For example, a μP or Interface Pod sends out address information during a READ and expects the system ROM, RAM, or the addressed peripheral device to return stable data to the bus within a predefined time period. The timing margin is the number of extra nanoseconds (nsec) that are available during this process. Marginal timing problems occur when the timing margin isn't long enough for data to be stable when the μP of Interface Pod expects it to be.

Designing a UUT with marginal timing characteristics is generally recognized as poor design practice, but it does happen, intentionally or unintentionally. (To avoid this situation, see the "What's the difference?" article in the August 1983 issue of the TROUBLESHOOTER for an example of timing margin testing.)

A good way to detect a marginal timing problem is to measure the timing of the system using the Troubleshooter and an oscilloscope. To do this, synchronize the oscilloscope to DATA using the Troubleshooter TRIGGER OUTPUT BNC Connector (on the rear panel). Do a looping READ operation and look for marginal timing problems while comparing your waveforms to the timing waveforms in the manual for your particular Interface Pod.

A Timing Margin Estimator

Evaluating the timing is not an easy task but a tool can be built to make this much easier: the Timing Margin Estimator. What is needed is a plug-socket device that will connect between the Interface Pod cable and the UUT μP socket that inserts a delay circuit (see Figure 1) into one of the data or address lines, e.g., DO). The timing margin can be experimentally determined by adding a variable delay in any portion of the system's address or data path and adjusting the delay until a system error occurs.

NOTE: The circuit of Figure 1 will also work on μPs with multiplexed data/address lines. The delay will affect both an address signal and a data signal, but it will still detect the minimum delay that will cause a problem.

If a 10-turn potentiometer with a calibrated knob is used, a synchronized oscilloscope can calibrate the device according to knob settings. Figure 2 shows a sample calibration curve. Due to different drive currents and voltage levels of different devices, the Timing Margin Estimator should be re-calibrated when it is used on a different UUT (or even a different part of the same UUT).

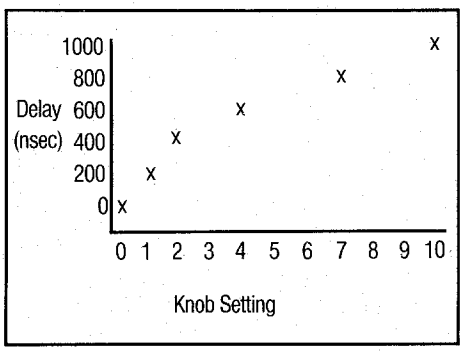


Figure 2. Sample Delay Curve

By adjusting the potentiometer, approximately 10 to 1000 nsec of delay can be introduced into the signal path. Increasing the delay during a looping RAM SHORT is an easy way to determine the timing margin of the RAM. For ROM, the delay must be adjusted while performing a looping ROM test. To speed up the test time, limit the ROM test to about 10 bytes.

The timing margin estimator can be a good tool for evaluating new UUTs and for preparing them for 9000-Series testing. You can also use the Timing Margin Estimator to measure the timing margins on a known-good UUT, and then use those measurements to determine if the timing margins on a suspect UUT are within the proper range.

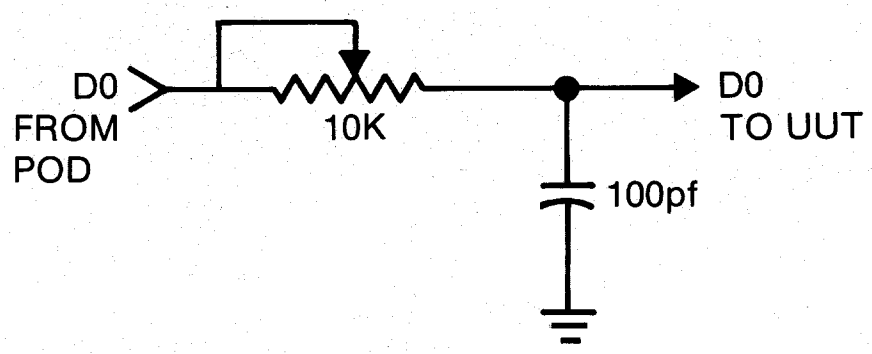


Figure 1. Timing Margin Estimator

Get a free Fluke 77 multimeter!

The fourth issue of the TROUBLE-SHOOTER included an article offering a free Fluke 77 Multimeter. How can you get one? Write an article for the TROUBLE-SHOOTER and have it published. At this time, new customer-written articles are being reviewed. Look for the first in our next issue.

What do we need? Articles you feel will be of interest to other users. Subjects of interest might include:

- Solutions to particularly troublesome testing problems.
- Unique Troubleshooter applications.

Articles should be submitted to:

Troubleshooter Editor
John Fluke Mfg. Co., Inc.
M/S 267-D
P.O. Box C-9090
Everett, WA 98206

Published articles will carry your name and your company's name in the by-line. In addition, when the article is published, you will receive a free Fluke 77 Handheld Multimeter.

Advanced training schedule

The current worldwide Advanced Training schedule is listed below. Contact your local Fluke Sales Office or Representative for prices and registration details.

Remember, we recommend that anyone attending the Advanced Training seminar first attend a free Introductory Training class, available through your local Fluke Sales Office or Representative.

Stockholm, Sweden	Feb 6-10 '84
Paramus, NJ	Feb 14-15 '84
Copenhagen, Denmark	Feb 20-24 '84
Madrid, Spain	Feb 27-2 '84
Burlington, MA	Feb 28-29 '84
Helsinki, Finland	Mar 5-9 '84
Montreal, Canada	Mar 13-14 '84
Munich, Germany	Mar 26-30 '84
Detroit, MI	Apr 3-4 '84
Benelux, Tilburg/Brussels	Apr 9-13 '84
Rolling Meadows, IL	Apr 17-18 '84
London, UK	Apr 24-4 '84
Minneapolis, MN	May 1-2 '84
Oslo, Norway	May 21-25 '84
Tel Aviv, Israel	May 21-25 '84
Benelux, Tilburg/Brussels	Jan 4-8 '84
Paris, France	June 18-29 '84
London, UK	Sep 3-14 '84
Benelux, Tilburg/Brussels	Sep 24-28 '84

Corrections and clarifications

Revised Part Numbers

In the fourth issue of the TROUBLE-SHOOTER a Probeable Socket was mentioned in the article on "Find that bus short". Unfortunately, the part number we used in that article was incorrect. The Probeable Socket can be ordered through our Parts Department by using the following noun and part number: 9000A-7201 (40-pin Basic Accessory Kit) John Fluke Part Number 648907.

In the third issue of the TROUBLE-SHOOTER the 8085 UUT Adapter was introduced. Again, unfortunately, the part number was incorrect. The 8085 UUT Adapter may be ordered through our Parts Department by using the following noun and part number: 9000A-8085-7201 (Accessory Kit) John Fluke Part Number 715862.

Cable Tester Parts List

In the fourth issue of the TROUBLE-SHOOTER we forgot to print the parts list for the Cable Tester (figure 4, Testing non-μP digital assemblies). We have also provided the parts list on a removable sticker that can be placed on issue number 4 in the upper left hand corner of figure 4.

U1, U3-U10, U19	74LS541
U2, U20	74LS138
U11-U18	74LS374

We apologize for any inconvience that we may have caused by these errors or omissions.

FLUKE®

John Fluke Mfg. Co., Inc.
P.O. Box C9090, Everett, WA 98206
(800) 426-0361 (Toll Free) in most of U.S.A.
206-356-5500 from other countries

Fluke (Holland B.V.)
P.O. Box 5053, 5004 EF, Tilburg, The Netherlands
Tel. (013) 673973, TELEX 52237

Printed in U.S.A. P0001E-01U8402/SE EN