

To mux or not to mux?

You may want to rethink your ICT strategy as the tradeoffs associated with the performance, cost, and efficiency of multiplexed in-circuit testers and pure-pin testers evolve.

ALAN ALBEE AND
ANTHONY SUTO,
TERADYNE



FIGURE 1. Shown here is the waveform of a driver programmed to drive 1.5 V under both no-load (blue) and load conditions (yellow). With a 6- Ω load condition, the voltage at the device under test reaches only 710 mV.

In-circuit board-tester pin architectures have traditionally come in two basic configurations. So-called “pure pin” testers dedicate a driver/sensor circuit to every test point on a board. This allows engineers to develop fixtures and test programs in parallel and permits maximum flexibility for last-minute modifications. In “multiplexed” pin testers, a single driver/sensor tests a set of pins—usually eight or 16.

Manufacturers have to weigh the pros and cons of each architecture and choose the best solution for a given situation. Some companies, depending on their product mixes and manufacturing processes, may choose to deploy both types of testers in concert to minimize learning curves, reduce bottlenecks, and keep costs as low as possible.

The pure-pin solution

The earliest in-circuit testers all featured what is now called “pure-pin” design. Every pin that made contact with the board under test had its own driver/sensor combination. Any pin on the tester could be assigned to any node on the board, and (in theory, at least) the tester could drive all pins simultaneously.

Test-program development could proceed in parallel with fixture design and construction. Unless the fixture contained an error, debugging the test program and adding tests did not require any wiring changes. Implementing the inevitable engineering change orders during production involved reassigning pins and rewiring fixtures only to the extent required by the changes themselves. The relatively limited number of pins permitted by this scheme easily met the needs of the time.

The increasing complexity of printed circuit boards, however, began to strain tester resources. New boards often had more pins than testers could accommodate. Tester manufacturers began to provide additional pins, but the need expanded more rapidly than the testers’ ability to address it. In addition, the cost of providing a driver/sensor for every pin and the resulting power consumption by a test system made this approach very expensive.

To reduce costs, tester manufacturers began to compromise on test-pin performance, replacing compo-

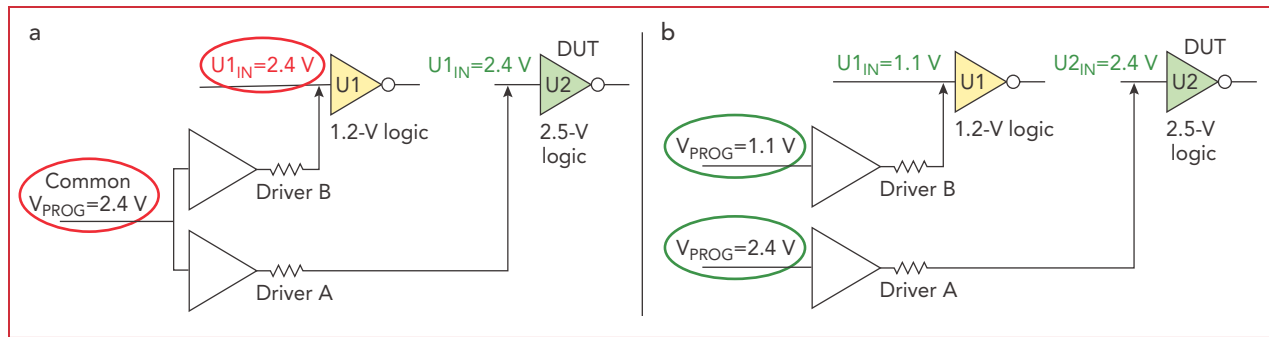


FIGURE 2. a) Assigning logic levels in groups of pins can compromise tests on mixed-logic boards, with a tester forcing damaging 2.4-V levels on 1.2-V devices. b) Testers that have independent per-pin programmable logic levels prevent overvoltage conditions.

nents such as drivers and sensors with less-accurate models. Unfortunately, these compromises also increased the likelihood that board defects would not be detected until later (and more expensive) test stages or—worse—until the product had shipped to customers. And none of these changes addressed the ever-increasing number of pins required to test the newest boards.

Consider some of the compromises and their implications:

- Less accurate drivers.** To ensure accurate voltages at the device under test (DUT) under all load conditions, a tester should employ a closed-loop, low-impedance driver designed specifically for in-circuit testing. To save money, a tester manufacturer may use off-the-shelf, high-output-impedance drivers, but in these drivers, the actual voltage at the DUT depends heavily on how much current the driver requires to achieve the programmed voltage. The voltage applied during a test can vary considerably depending on the nature and state of the net.

The screenshot in **Figure 1** shows the waveform of a driver programmed to drive 1.5 V under both no-load and load conditions. With a 6-Ω load condition, the voltage at the DUT reaches only 710 mV.

Suppose the tester tells a driver to send an input high. The driver can de-

liver an accurate voltage only as long as the output pin connected to that same driver is either shut off or is also high. If the driver tries to send an input high and the output pin is driving low, the actual voltage at the input pin will be much less. Debugging that test—and achieving a reliable repeatable test—becomes much more difficult. In addition, testers will not generally report the actual voltage at the driven pin. Few people will take the trouble to attach an oscilloscope to the pin (and every other questionable pin) to measure the voltage.

between 500 mV and 900 mV. With digital logic at 3.3 V or more, such a discrepancy may prove acceptable.

On the other hand, many devices operate much lower than 3.3 V. These low-voltage technologies may set highs at 1.2 V or even less. At these levels, the “slop” in the sensor’s detection can produce unrepeatably, unreliable tests. In addition, such uncertainty means that running the same test on two different (but allegedly identical) testers may not give the same mix of passed and failed boards. Reliable testing of low-voltage parts,

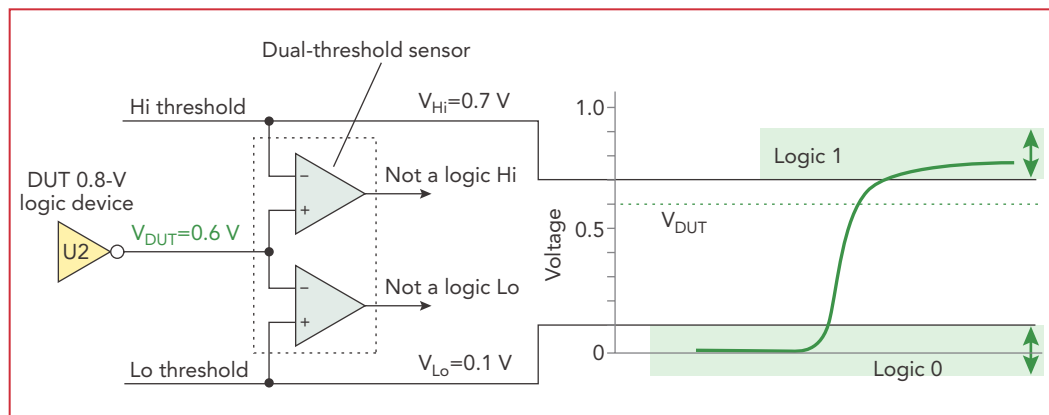


FIGURE 3. A dual-threshold sensor design identifies incorrect or marginal logic levels on a board under test.

Even if the tester can adequately drive pins on a good board, a board fault complicates the situation further. An open enable pin, for example, forces the tester to back-drive outputs that are no longer disabled. The resulting inaccurate voltages can cause false failures.

- Less accurate sensors.** Most in-circuit testers rate their sensor measurements with margins of either ±100 mV or ±200 mV. Therefore, an IC output driving 700 mV can appear to the sensor to be between 600 mV and 800 mV or

therefore, requires an in-circuit sensor with margins as low as ±15 mV.

- Limited logic levels.** Assigning logic levels in groups of pins can compromise tests and even damage parts on mixed-logic boards. Consider the test illustrated in **Figure 2a**, in which the tester forces both U1 and U2 to the same voltage levels. Unfortunately, a driving voltage that accommodates U1 to a dangerous overvoltage condition. Testers that have independent per-pin programmable logic levels, as shown in **Figure 2b**, pre-

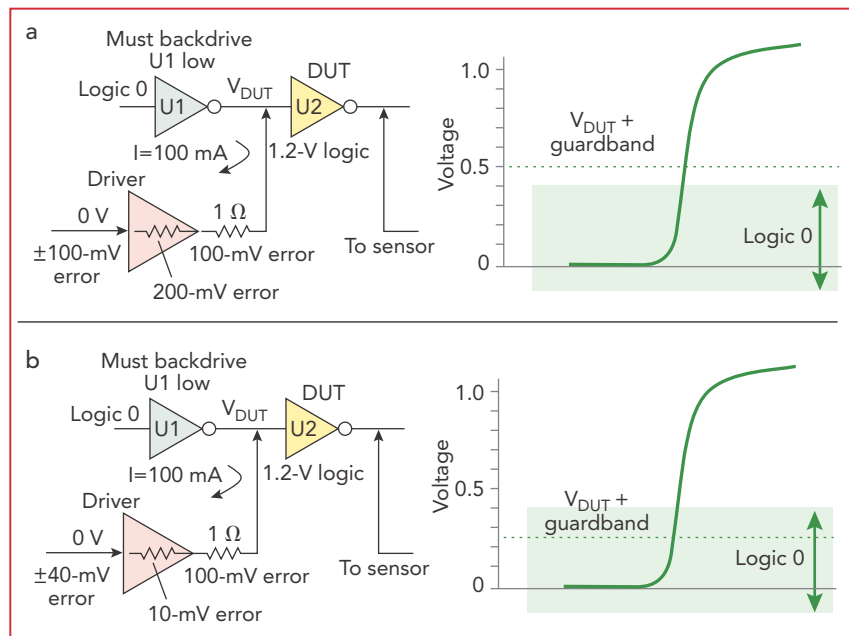


FIGURE 4. a) A unidirectional driver configuration renders U2 untestable because a programmer can't program the output voltage to anything less than 0 V to reduce U2's input voltage. b) A bidirectional driver overcomes this limitation and can reduce noise margins.

vent overvoltage conditions by assigning voltages to each device pin individually depending on its technology, so it will remain within a safe operating range.

Limited logic levels can make debug more difficult as well. One manufacturer, for example, found that he couldn't change the programmed logic level on one particular pin without changing the level for all 32 pins in that group. He had to assign that pin to another group by adding a wire to the test fixture.

- **Single level thresholds.** Defining only a single threshold to denote the transition from 0 to 1 ignores the fact that most devices have VOH_{MIN} and VOL_{MAX} specifications that create a gray area between the two. With a single threshold, a test cannot detect a faulty output transistor that can't drive high enough or low enough for reliable operation of the board in its target system. The board would have to proceed to functional or system test where diagnosing it would be more difficult. Dual-threshold sensors (Figure 3) don't have this ambiguity, so they can verify that the output transistors are functioning according to the manufacturer's specifications.

- **Unidirectional drivers.** Unidirectional drivers always define a low volt-

age at ground. This hard limitation reduces program debug flexibility and prevents users from adjusting a program so the board under test can compensate for noise that can trigger undesirable logic transitions. It also prevents the tester from dealing with negative-voltage technologies such as emitter-coupled logic (ECL).

Figure 4a shows a unidirectional driver. The driver inaccuracy plus the guardband totals 500 mV, and the maximum low input for U2 is 420 mV. In this configuration, U2 is untestable, because the programmer can't program the output voltage to anything less than 0V.

In contrast, the driver inaccuracy and the guardband of the bidirectional driver in Figure 4b total only 250 mV. Again, the maximum low input of U2 is 420 mV. In this case, U2 is testable even without reprogramming the logic low to -0.2 V, but adjusting the bidirectional driver can reduce noise margins.

- **Fixed slew rate.** Different device technologies often require different slew rates. In fact, some devices will not operate consistently unless the slew rate of the driving signal exceeds some minimum. Slew rate also affects the amount of signal ringing and noise. Generally, the faster the edge, the more likely it is

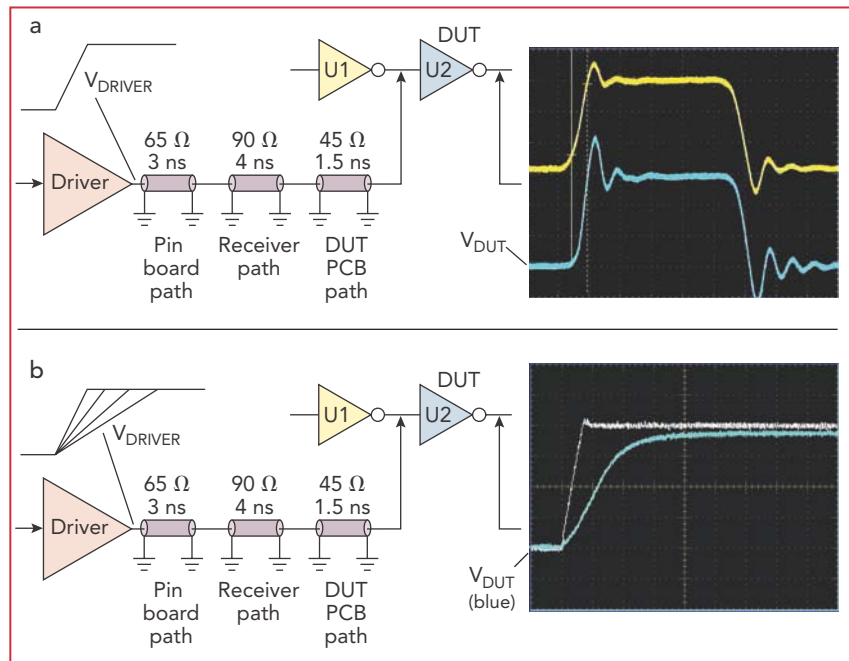


FIGURE 5. a) A fixed slew rate, such as the 300-V/ μ s edge rate, is too fast for some applications, where transmission-line effects will cause overshoot and ringing. b) An in-circuit driver with a fully programmable slew rate can optimize the waveform at a DUT to alleviate these problems.

that the signal will experience overshoot or ringing. Inductance and impedance in the fixture wiring also limit the slew rate.

A fixed slew rate, such as the 300-V/ μ s edge rate in **Figure 5a**, is too fast for some applications. Transmission-line effects will cause overshoot and ringing, and the user cannot lower the edge rates to eliminate potentially dangerous overvoltage conditions. **Figure 5b** shows how an in-circuit driver with a fully programmable slew rate can be used to optimize the waveform at the DUT.

• **Limited system power.** Device packages may include hundreds of signal pins. Required back-drive currents can exceed 250 mA per pin. Because real pin systems allow programmers to drive all pins in parallel, the total of the back-drive currents may exceed the limited power capacity of some systems. In that case, the drivers will fold back and will not reach their programmed voltages, producing false failures. Therefore, the actual number of pins that the tester can drive in parallel is application-dependent. Programmers should be aware that the limited system power ratings of some “pure-pin” systems can

prevent simultaneous driving of tester pins when heavy back-drive conditions are involved.

Multiplexing to the rescue

To address some of the cost and pin-count issues while maintaining the maximum pin quality and performance, some tester manufacturers offer multiplexed test pins. Multiplexing allows test instruments to be shared (or pooled) among many (usually eight or 16) test points, and it expands the number of test points at lower cost. Without the need to support a test instrument behind every pin, multiplexed testers generally offer greater pin capacity and consume less power. Many also offer higher-performance pins.

Unfortunately, multiplexing does have drawbacks. It places additional restrictions on test program and fixture development. Programming becomes more complicated because a measurement cannot randomly assign the drivers/sensors and instruments that will be used during each test. Developers need to use nail-assignment software to analyze the tests and resolve any conflicts before building a fixture. *(continued)*

For tests that require numerous test points, you may not be able to find enough drivers and receivers to perform the measurements unless you assign test points that are spread out. This will result in increased path lengths between the tester nails and the device pins, which may increase noise and signal delays and make initial verification of fixture pin assignments more difficult.

Large devices on the board may have more pins than the tester has real pins available, so you may have to break the test into multiple bursts. To perform basic debug tasks, such as adding guard points and implementing engineering changes, you may have to reassign test pins and rewire the fixture, so the effort and cost of implementing any changes increases.

Which way to go?

Multiplexed and pure-pin (or “nonmultiplexed”) architectures will both work in most situations (as long as they both use high-performance pins). Test-program development times may be slightly longer for muxed systems, but many users are comfortable with them and prefer them because of their generally lower price. High-performance non-muxed systems can be more expensive, but they permit faster and easier program development.

Needing to test a high-pin-count board will tip the balance toward the multiplexed systems. Providing enough real pins to test a board containing more than, say, 4000 nodes is neither cost efficient nor energy efficient.

Lower-pin-count applications requiring fast development or development by less-experienced users would favor the pure-pin solution. If you go this route, you can choose test platforms that let you easily move between pure-pin and multiplexed tests. See “The best of both worlds,” in the online version of this article for more information about one tester that offers this option, www.tmworld.com/2006_11. T&MW

Alan Albee is an in-circuit test product manager at Teradyne where he applies his 23 years of test engineering and applications experience.

Anthony Suto is Teradyne's chief scientist responsible for the Assembly Test Division's ICT and AXI technologies.