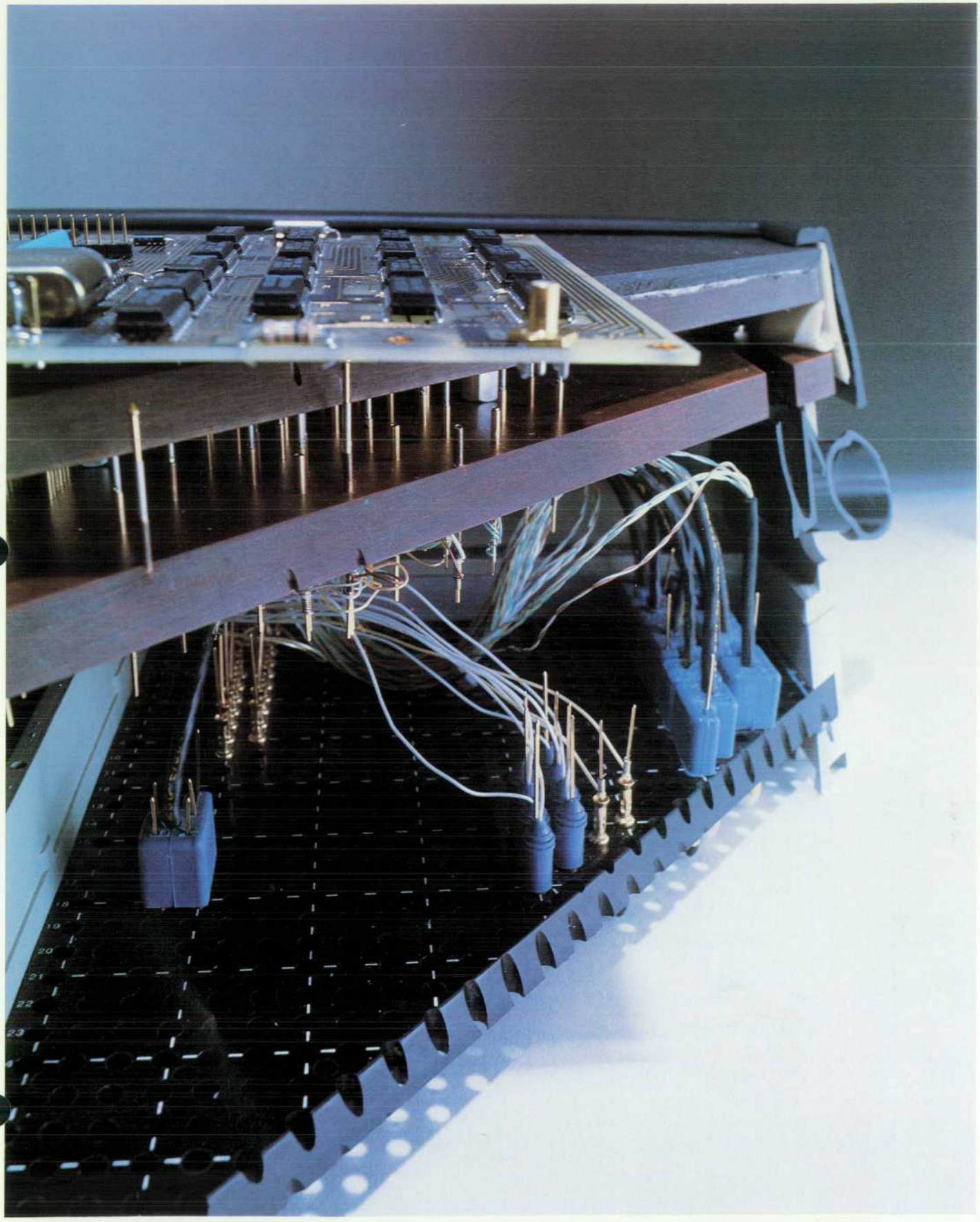
HEWLETT-PACKARD
JOURNAL

OCTOBER 1984



HEWLETT-PACKARD JOURNAL

October 1984 Volume 35 • Number 10

Contents

In this Issue

If our designs and our manufacturing processes were perfect, there would be no need for testing, since every product coming off the assembly line would work perfectly. In reality, the more complex the product, the more testing it needs, and the more difficult the task of developing thorough, exhaustive tests that can be completed in a reasonable amount of time. In electronic printed circuit board production, automatic test equipment can be considered a contribution to the state of the art if it tests larger, more complex boards, tests them faster, tests them more thoroughly, tests them more safely, and/or makes it easier for the test programmer to generate tests. In this issue, the designers of the HP 3065 Board Test System tell us how their system makes contributions in all of these ways. Designed for high-volume production testing of large, complex printed circuit boards, the HP 3065 does both functional testing (Does the board work?) and in-circuit testing (Is any part wrong, misloaded, faulty, or out of tolerance? Are there solder splashes causing short circuits? Is there assembly damage?). By carefully minimizing the biggest contributor to test time—the long overhead time between individual tests while the computer transfers data—the HP 3065's designers have made the system fast, particularly in testing integrated circuits on boards. It can test more than 30 ICs per second of just about any complexity. In 20 seconds, it can test for short circuits and test all of the analog and digital parts on a 14-by-18-inch board with 206 ICs and 100 analog devices.

The HP 3065 does in-circuit testing by forcing the inputs of a device to known levels. To do this, it sometimes has to overpower other devices on the board that are connected to the device under test. This technique, which is called overdriving, has been known to cause failures at times and has been suspected of shortening the lifetimes of circuit components. So that this won't be a problem in the HP 3065, the system's software automatically analyzes each test for damage potential and limits currents, voltages, and test durations to safe values. The article on page 20 describes how these safe levels are determined and maintained.

Like many other board test systems, the HP 3065 uses a bed-of-nails test fixture. Our cover photograph shows it cut in half so you can see what's inside. The board under test is placed on top and then a vacuum is applied (through the hoses you can see in the picture) to pull the platform holding the board down towards the platform underneath it. This forces the pins of the test fixture against the nodes of the printed circuit board. To write a test program for a particular board, all a programmer has to do is tell the HP 3065 what kinds of components are connected to the various nodes. If the board has been designed using a CAD (computer-aided design) system, the HP 3065 can simply be given a data file from the CAD system. The HP 3065 then looks in its large library for individual component tests, writes an overall test program, optimizes it, and generates a wiring diagram to tell the test personnel how to connect the bed of nails to the system's pins (they're in the holes in the bottom platform in the cover photo). If there are components on the board that aren't in the system's library, the programmer has to tell the system how to test them. Usually, 90% or more of the components will be in the library. Three articles in this issue deal with various facets of the HP 3065's automatic test generation software.

High-volume testers for complex boards aren't inexpensive. However, the HP 3065's high throughput and automatic test generation tend to minimize the cost per test, and the information it gathers about production defects can lead to improvement of the design and the manufacturing process. All things considered, it can be the most cost-effective solution to a formidable problem.

-R. P. Dolan

What's Ahead

The November issue will describe the design and features of three different kinds of HP products. An easy-to-use network analyzer, the HP 3577A, and a companion S-parameter test set, the HP 35677A/B, are discussed in three articles. Two more articles describe the HP 293X family of high-quality dot-matrix impact printers and the custom IC developed to coordinate their printing functions. The last article describes a rugged terminal, the HP 3081A, designed for use in severe industrial environments.

The HP 3065 Board Test Family: A System Overview

This board test system features menu-driven automatic test generation, high digital IC throughput, overdrive protection, multiple test stations, and networking capability.

by Thomas R. Fay and John E. McDermid

THE MANUFACTURE OF TODAY's sophisticated electronic equipment usually requires the assembly of one or more printed circuit boards, each containing one or more integrated circuits and a number of discrete components. As these boards grow more complex, increasing in the number of components and interconnections, the need for testing these boards quickly and easily becomes more important. The sheer numbers of parts and subassemblies conspire to make maintaining quality and production yields difficult. What is worse, the farther along in assembly a product gets, the more difficult it becomes to excite, detect, and diagnose manufacturing defects. The result is that it becomes very attractive to test subassemblies at a level of integration high enough so that the number of remaining assembly steps is manageable, but before the view into the causes of problems becomes clouded by complexity. Printed circuit boards fit this bill nicely—not too complex to test, diagnose, and repair; not so low-level that the number of boards to be assembled presents a severe quality problem.

The HP 3065 Board Test System (Fig. 1) is Hewlett-Packard's latest model for evaluating complex analog/digital printed circuit boards. Building on the strengths of HP's earlier board test systems, it has many features that simplify the development of complex board tests and enhance their usefulness in the production environment. A "computer behind the pins" design incorporates an improved in-circuit test program generator, called IPG-II, that is capable of generating correct tests for an average of 90% of the circuit components and digital devices found on a typical circuit board. Up to 22 analog or digital cards may be used in the system. Each digital card can address up to 48 circuit nodes and each analog card can address up to 64 nodes. The wiring connections between the cards and the board are assigned by IPG-II, because the digital cards multiplex one driver/receiver channel for every four nodes (12 chan-

Fig. 1. *The HP 3065 Board Test Family provides printed circuit board manufacturers with a combination of high throughput, excellent test quality, and low programming costs for evaluating complex digital and analog circuit boards. The basic configuration shown consists of a measurement section (right), an instrument section (center), and a programming station (left).*

nels per card). The multiplexing scheme lowers hardware cost and the wiring assignment by IPG-II reduces test setup time.

Because the HP 3065 can perform both analog and digital tests at both the in-circuit device and the board function level, complex ICs and circuits such as ROMs, microprocessors, analog-to-digital converters, memory boards, and instrumentation amplifiers can be evaluated.

In-circuit testing of digital devices often requires overdriving adjacent device inputs and outputs to obtain the necessary test conditions. To minimize any stress on these overdriven devices that might degrade their functionality or lifetime, the HP 3065 has carefully designed safeguards built into its test software.

Careful attention to the HP 3065's hardware and software design has resulted in improved test throughput of greater than 30 ICs per second. A friendly forms entry format allows nontechnical personnel to enter board descriptions quickly to lower the cost of test development. Alternatively, the board description can be supplied directly in ASCII format by the CAD/CAM system used to design the board.

Since the data gathered during production testing is often of value to other manufacturing activities such as quality control, inventory control, and yield improvement, the HP 3065 supports several links for networking to other systems, including earlier HP 306X Test Systems, HP 1000 and HP 3000 Computers, and many HP desktop computers. The box on page 6 describes the HP Q-STAR features recently added to the HP 3065.

Board Testing Considerations

There are several important reasons for using a board test system. Most frequently it is used to filter the manufacturing process so that defects can be detected and eliminated at the earliest possible stage of the process. Early detection reduces the expense associated with correcting these defects.

A second and very important reason is to measure the manufacturing process so that corrective action can be taken to prevent future defects. An HP 3065 System allows parameters to be measured on every component on the printed circuit board as well as the board itself. With this kind of measurement capability, large amounts of data can be taken and examined, control charts developed, and the effect of process changes determined.

A third reason is to increase the capacity of an existing production process. Often, the ability to increase the volume of boards manufactured in a given time is limited by the availability of skilled technicians. If much of their time is spent tracking difficult problems that in the end can be traced to short-circuited, misloaded, wrong, or grossly defective components, then throughput can be greatly improved with the use of a board test system.

The major application area for the HP 3065 is in the production environment. While this system can be used in field or depot repair applications, the necessary quantity of test fixtures is typically larger and the quantity of boards of a single type is lower. Hence, for some field repair applications, the system may not be cost-effective. Also many of the library tests are customized to the type of fault spectrum that is found in production. As an example, on-board short circuits are seldom the cause of a field failure, but can be quite common in the production environment.

Production testing of printed circuit boards has some rather strong constraints. First, the development of tests for boards naturally must follow the development of those boards and hence occurs near the end of the development cycle. On the other hand, testing is a vital part of starting up production of the boards, especially in the beginning when the production process needs to be fine-tuned. The result is a squeeze play on the test development time and the test programmer, especially since products often contain multiple boards that reach production almost simultaneously. In these boom situations, the number of skilled test designers frequently falls short of the requirements, and is often too small in any case.

Another severe constraint is the way that testing fits into the manufacturing process. First, it must not take much time out of the total manufacturing time for a board, or it becomes a bottleneck and a work-in-process inventory builder—neither very desirable. Second, it must test the boards thoroughly so that what emerges from the test process are reliable boards. And third, it must provide diagnostic information so that defective boards can be quickly repaired. Finally, testing must provide useful real-time feedback on the manufacturing process so that production yields can be steadily improved and excess scrap can be avoided. Being able to communicate test results to other manufacturing computer data bases and systems is also important.

The HP 3065 Board Test System deals with all of these problems. Its internal test program generator (IPG-II) automatically generates test sequences from a description of the board so that tests that cover 90% of the manufacturing defects can be created very rapidly. These tests can be used to provide early feedback on board quality and can be further enhanced manually as time permits to increase coverage and further improve yields. The automated nature of the test generation and the simplified user interface combine to allow novice technicians to handle the testing. The use of a multiprogrammed computer as a controller allows the HP 3065 to support multiple programming stations to meet widely varying test development work loads. The same computer has sufficient power to let the HP 3065 support up to three test heads, each providing excellent test throughput.

Within the frequency limitation of the bed of nails (matrix of spring-loaded pins for making connections to the board under test, see cover photograph) and the accuracy limits of the HP 3065, the system is also capable of performing functional testing on the board under test. The full range of HP's programmable HP-IB (IEEE 488) instrumentation can be used to facilitate these measurements. This makes it possible to test individual components and functionally check, adjust, and qualify a printed circuit board with one operation on the same piece of test equipment.

System Hardware

The HP 3065 System is the culmination of two significant development efforts—one in hardware, the other in software. The hardware effort produced about 29 different printed circuit boards (some of which are quite large) and three full bays

of equipment. This hardware has three major components: the controller, the test heads and optional test-head equipment, and the programming stations and peripheral devices.

Controller. The HP 3065C Controller is an HP 1000 E-Series Computer with up to 2M bytes of memory, a 140M-byte disc drive, and a quick-disconnect panel for test heads, programming stations, and peripherals. The controller is capable of concurrently operating up to three test heads and three programming stations. The test heads and the programming stations can be located up to 90 meters away from the controller by adding two bus extenders to the controller for each remote test station. Up to six extenders can be mounted in the controller. Network interfaces are provided for both high-level data link control (HDLC) and X.25 protocols. HP-IB interfaces are a standard part of the system. Additionally, a modem interface is provided for a remote terminal that can be used as either a programming station or for remote diagnostic capability.

Test Heads. A test head consists of two full bays: the measurement bay and the instrument bay. Within the measurement bay are floating voltage and current sources. In addition to dc, ac, and function generator sources, there are ac and dc voltmeters, frequency counters, and a pulse width detector. These sources and detectors can be used with special circuitry to isolate the component under test and measure other parameters such as resistance, capacitance, inductance, diode voltage drop, and transistor ac β. Also within the measurement bay are digital drivers and receivers, memory for test vectors, and a high-speed computer to sequence and validate test vectors. All multiplexing necessary to interface to the board under test is contained in this bay. The instrument bay contains power supplies for the device under test and any optional HP-IB instrumentation that the user may wish to configure into the system. Additional equipment frequently requested for particular applications includes an HP 1980A/B Oscilloscope, an HP 3325A Frequency Synthesizer, an HP 5335A Universal Counter, and an HP 3456A System Voltmeter.

Programming Stations. Programming stations consist of terminals, additional computer memory, and whatever peripherals are desired. These programming stations can be used while the test head is in operation without significantly affecting test-head performance.

HP Q-STAR

The HP 3065 is more than just a board test system, it is a cornerstone of the HP Q-STAR (Quality Systems for Test, Analysis, and Repair) network. This integrated package of software and hardware is designed to simplify the solution of interrelated problems of test, analysis, and repair of printed circuit boards. Some of the features provided by HP Q-STAR include:

- HP CAD-VANTAGE. This software allows users to develop test programs faster by automatically capturing circuit data from a wide variety of commercially available or proprietary CAD systems.
- HP Q-STATS. This software can be used to identify trends in component parameters, manufacturing board yields, repair productivity, and quality by providing test managers and programmers with summarized data from the test and repair processes.
- HP CHEKPOINT. This software automatically verifies that the spring-loaded pins in the bed-of-nails board test fixture are actually making contact to the board under test.
- HP REMOTE SUPPORT/3065. This feature provides on-line communications between HP 3065 computer systems and HP support centers to help reduce downtime when servicing is required.
- High-level network access. This software allows users to share data among various computer systems via RS-232-C/V.24, HP-IB (IEEE 488), and HP-DS links.
- Expanded digital test library. New in-circuit tests for the most popular new LSI devices to help reduce test development time for state-of-the-art board designs.
- IPG-II enhancements. New analysis algorithms reduce the amount of manual adjustments required to complete development of a test program.
- Bar-code data entry. The HP 3065 software now allows the use of a bar-code reader for high-level entry of repair and board identification data for tracking boards through the test and repair loop.
- HP PAPERLESS REPAIR/REPORTING. This software eliminates the need for paper failure reports, permits tracking of the test and repair history for improved repair operation control, and gathers the repair data for analysis by HP Q-STATS.

System Software

The software for the HP 3065 is the result of a large design effort—writing, integrating, and testing about 450,000 lines of source code. This software is a fully integrated system of eleven major packages (Fig. 2):

- Screen editor
- Board Test BASIC
- Manual board topology entry
- Board topology from computer-aided design and manufacturing systems
- Automatic in/circuit test program generator (IPG-II)
- Wire list
- Test plan
- Vector control language (VCL)
- Digital libraries and test safeguards
- Networking
- Confirmation and diagnostics.

Screen Editor. A very important design objective was to integrate these software entities so that the user perceives only one. The key to this integration is the editor. It provides users with a window on the software that is always the same. At the top of the programming station's display screen is a status line where the current state of the machine and any error messages are displayed. The next line down is the command line. A statement in the command line is parsed and executed immediately when the execute softkey is pressed. Execution of statements on this line is independent of the editor's mode. When the command/edit softkey is pressed, the cursor toggles between the command line and the edit area. Files are placed in the edit area with a load command. Whenever editing is done, each line is checked for correct syntax before it is accepted. If an error is detected, it is reported in the status line so that the user can correct it immediately. The definition of what is syntactically correct changes with each mode.

Board Test BASIC. Board Test BASIC is an adaptation of BASIC to the board test environment. It has been written

so that language extensions can be provided for specific board test functions such as testing a resistor or a digital device. The interpretive nature of BASIC makes it easy to test and debug. Specialized statements such as test make invocation of high-speed compiled languages (like that used to test digital devices) very natural and remove the speed limitation often associated with an interpretive language. Board Test BASIC is the control structure for the user. Whether the task is sequencing a series of tests, copying a file from one location to another, or controlling a group of editor commands, it is accomplished through BASIC. See the article on page 25 for more details.

Manual Entry. The manual board topology entry is a special software product optimized for use by clerically skilled personnel. Its "fill-in-the-blank" forms entry is easy to learn and minimizes the amount of typing that needs to be done to enter data manually. Data for entry into this package is taken directly from the schematic and material list for the board. (Examples of displays for entering data for a resistor and a transistor are shown in Fig. 3 and Fig. 4.) This package can be entered either from the command line or scheduled from a BASIC program. Scheduling it from a program allows a user to predefine all files and output devices so that clerically skilled personnel will not need any programming knowledge.

CAD/CAM Entry. The board description entry software is designed to accept ASCII files (from any source) as an alternative way of obtaining information about the board. If material lists and board topology are available from another computer in ASCII form, this method of entry will greatly enhance the productivity of the system. When an ASCII file is available in the predefined format, it can be compiled into a file that can be viewed and edited on the HP 3065 using the board description package. This allows corrections to be made or information added in a straightforward way.

IPG-II. After the printed circuit board information is loaded, the next step is to develop a test program. An automatic in-circuit test program generator, called IPG-II, is provided that produces analog and/or digital tests for each individual component. This analysis takes into account test instrumentation effects on the measurement process and reduces the effects of other devices connected to the component under test by using a technique called guarding.[1] Automatic program generation uses the topology information from the board, material list information, and library information on digital parts and then combines them to produce a BASIC program called Test Plan. Experience has demonstrated that more than 90% of these tests will not require any manual adjustments by the user (see articles on page 11 and page 31).

Wire List. A second output of the program generator is a complete set of fixture wiring instructions and a wire list. Each type of board to be tested requires a specially designed text fixture (bed of nails) which uses spring-loaded pins to make connections to each critical circuit node on the board (see Fig. 5). The wire list is a file that specifies which test fixture connection points must be connected by the user to which spring-loaded pin. For convenience of production change orders, an old wire list can be used as an input to the program generator and a minimum set of changes will be output.

Test Plan. The Test Plan program output by the in-circuit program generator IPG-II serves to sequence all the tests for the board. It can be edited or augmented by the user in the same way as any BASIC program. Test Plan controls four major types of tests: the Shorts test, analog in-circuit tests, digital in-circuit tests, and functional tests. The Shorts test (see article on page 28) is designed to find connectivity errors on the board. It tests the connectivity from one node to all other nodes on the board and expects to find no undesired connections on a good board. Analog tests check passive and many active components for important parameters such as resistance, capacitance, and inductance. Digital tests apply test patterns to prove that the correct device is installed and that a connection exists to all pins.

VCL and Digital Libraries. Digital tests are written in vector control language (VCL). This compiled language is de-

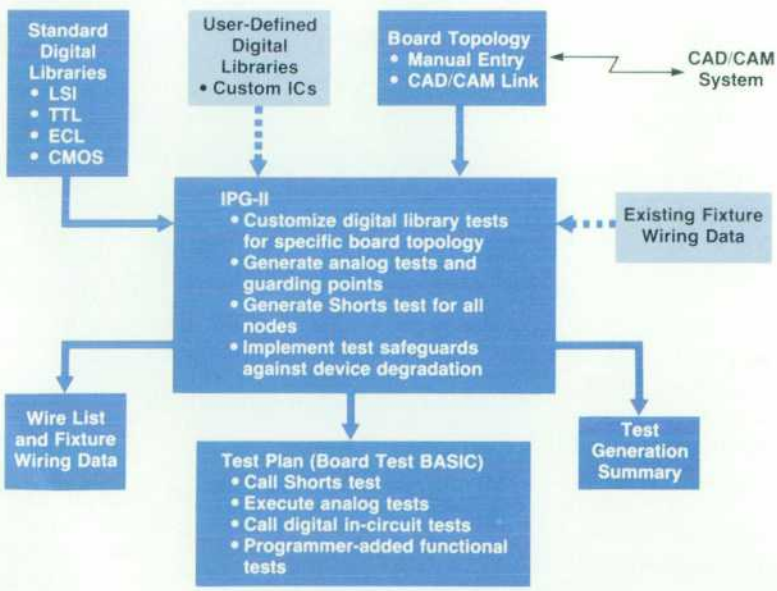


Fig. 2. *The software system supplied with the HP 3065 Board Test System. IPG-II is HP's second-generation in-circuit test program generator. Requiring only board topology as its input, IPG-II uses digital test libraries and the circuit's description to generate the analog and digital in-circuit test program.*

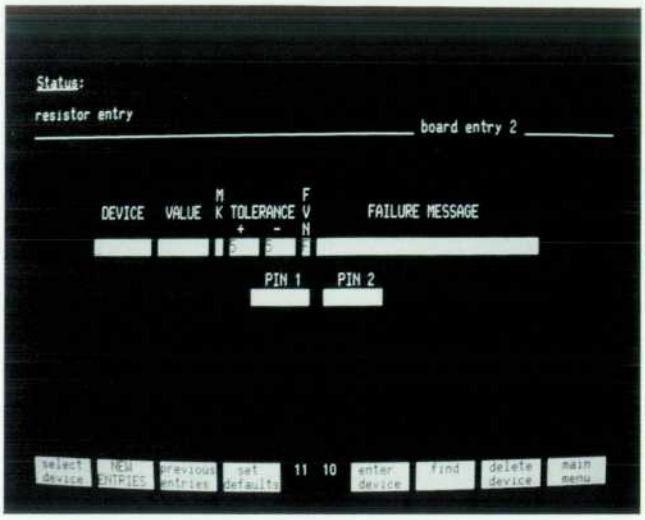


Fig. 3. *Display screen for entering resistor test information.*

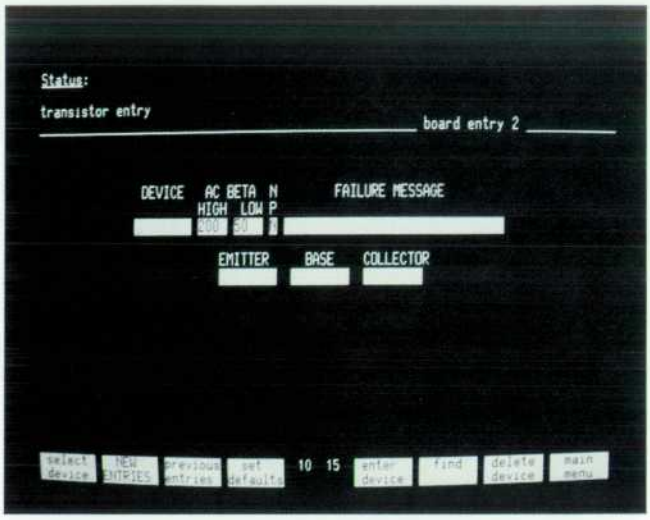


Fig. 4. *Display screen for entering transistor test information.*

signed explicitly for the bit-slice processor within the test head. The high-level language is compiled into object form and stored in a special disc file called the turbofile. When a digital test is executed, the object code for that particular device is retrieved from the disc (via the cache memory) and sent to the test head where it becomes the executable code for the bit-slice processor. Included in the digital device tests are safeguard features to protect devices under test from potential stress caused by in-circuit testing. These protective features and the primary failure mechanisms influencing their design in the HP 3065 are discussed in the article on page 20.

The architecture of the vector control language for a digital test closely parallels the architecture of the hardware. Structurally it is divided into three sections: the device declaration section, the vector definition section, and the vector execution section. Information in the device declaration section describes the device pins and their function, such as input, output, or bidirectional, and provides information on how to connect the device to drivers and receivers. The vector definition section describes the test vectors (device input/output conditions). It is this data that is stored in the pin RAMs. The vector execution section sequences the vectors, subroutines, homing loops, and counters, and forms the executable code for the bit-slice processor.

Networking. A very important feature of the system is the ability to interconnect several HP 3065 Systems. The network used is a topology-independent architecture supporting such diverse structures as star, string, and ring connections. Each network node has a store-and-forward capability and a network-wide nodal addressing system. A network system can be configured with redundant paths and a dynamic message rerouting system automatically bypasses a "down" node or communication link. The integrity of the message is assured by cyclic redundancy checking on all frames that are sent. The interface retransmits all frames received in error so that even large files can be received error free. The result is that a single copy statement in BASIC is all that is required to communicate with a remote device or file. The store-and-forward operation occurs transparently on intervening machines and does not affect testing.

Confirmation and Diagnostics. Finally, no system would be worthwhile if failures couldn't be detected and isolated

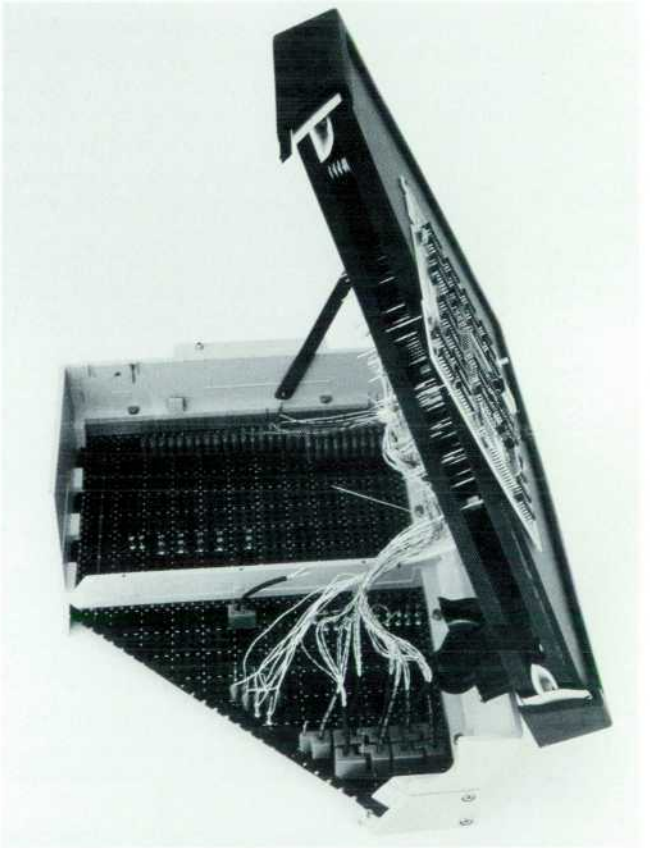


Fig. 5. *Cutaway view of test fixture. Contact to the board under test is made by using vacuum to pull the board down onto a configuration of spring-loaded pins (bed of nails). The HP 3065 tells the user how to wire these pins to the test fixture.*

Confirmation-Diagnostics

Incorporated into an electronic production line, the HP 3065 Board Test System becomes a key factor in determining overall throughput. Therefore, system failures can have a tremendous impact on productivity. Because of this, the HP 3065 is provided with a hardware/software package that can quickly and accurately diagnose failing hardware in the system.

Confirmation-Diagnostics, as this package has become known, had several design objectives. The first was to provide a fast verification of system integrity. This is called confirmation. This test, which runs in less than 20 minutes, verifies that all system hardware is operating properly. This knowledge is very important to a production line supervisor who must be able to trust the board test results produced by the HP 3065.

The second objective of Confirmation-Diagnostics was to provide a set of tests that would be able to locate the exact hardware failure, should one be detected by the confirmation test. This software package will isolate a problem in the system by running more comprehensive tests. These tests are able to isolate a failure down to any single relay or to the electronics on any single printed circuit board. Thus, a failure can be isolated quickly down to a module that can be replaced easily in the field. This saves the customer costly down time.

Implementing Confirmation-Diagnostics was part of the overall product definition from the very beginning. Every printed circuit board designed for the system includes features that allow it to be tested functionally under remote software control. One printed circuit board in the scanner bay was designated specifically as a testing card. The electronics on this card interfaces to the other scanner cards through a special fixture. This test card-fixture combination allows diagnostic access to every scanner point in the system. The system DVM and other test hardware built into the HP 3065 System can use this access to verify all the electronics in the system.

The major investment in implementing Confirmation-Diagnostics was in software development. This portion of the project involved over five engineer-years of effort. The software package includes a menu-driven, easy-to-operate user interface that makes extensive use of softkeys. It also includes almost one thousand tests, each one designed to test particular system functions. These tests are executed and managed by an executive which also keeps a log of all tests performed. This executive also analyzes the test results to determine which module or modules are faulty.

Diagnostics can be run automatically on a complete HP 3065 System, or can be run selectively on any subsystem, printed circuit board, or even individual relays. In addition, any individual test can be run in a loop or single-stepped through its various operations. These features are useful for in-depth troubleshooting by field service personnel.

Randy W. Holmberg
Systems Engineer
Southern Sales Region

quickly. The confirmation and diagnostic packages are designed to do just that (see box). The confirmation part of the software is designed to be run by any board test operator. All that is required of the operator is to establish a valid log-on, place the diagnostic fixture on the system, and press **START**. Then, in ten to twenty minutes, depending on the hardware configuration, a test head can be certified functional. The pass/fail condition is returned to the operator. If the test head is inoperative, service-trained personnel should be consulted. By invoking the diagnostic package for the test head, service personnel can view failing confirmation tests.

An automatic diagnostic feature can be used to determine the most probable module to replace. Another important part of the diagnostic package is the ability to use it remotely. This allows HP service personnel to access a user's system from their office and come much better prepared to solve a problem in the field.

Acknowledgments

The design of the HP 3065 was a large and exciting effort. Many people made contributions to the development of this system. The electrical hardware was the result of the efforts made by G. Siva Bushanam, Al Gookin, Steve Greer, Bob Illick, John Ketchum, Steve List, Dave Potson, Ken Saller, Larry Smeins, Mike Teska, and Brian Wood.

The software and system design was done by Bob Balliew, Doug Baskins, Roy Broeren, Rod Browen, Martha Conant, Bud Cribar, Dave Crook, Dave Glasgow, Bill Groves, Vance Harwood, Mike Hendricks, Randy Holmberg, Doug Manley, Mark Matthieu, Ron Mathieson, Dave McGownd, and Ken Posse.

The mechanical design benefitted from the contributions of Jim Berry, Rod Harris, Jim Hayes, Jim Jones, Muriel Keller, Dave Marttinen, and Bill Smith.

We would also like to thank Cullen Darnell and John Scruggs for their continued support throughout the duration of the project.

Reference

1. D.T. Crook, "Analog In-Circuit Component Measurements: Problems and Solutions," *Hewlett-Packard Journal*, Vol. 30, no. 3, March 1979.

Authors

October 1984

4 HP 3065 Board Test System

Thomas R. Fay

Tom Fay studied mathematics at Kalamazoo College (BA 1971) and the University of Michigan (MS 1972) and computer science at San Jose State University (MS 1980). After working as a programmer for a major retail chain and then as a software development manager for a major semiconductor manufacturer, he joined HP in 1979 as a software engineer. Tom worked on the MODCAL compiler, printed circuit design, and the software for the HP 3065's digital subsystem before assuming his current responsibility as project manager for HP 3065 test generation software. He is the author of two articles on software libraries and documentation and a member of the IEEE Computer Society and the ACM. Born in Decatur, Indiana, he now makes Fort Collins, Colorado his home. Tom is married and enjoys bicycling, swimming, tennis, and fishing.

John E. McDermid

Author's biography appears elsewhere in this section.

11 Automatic Digital Test Generation

Robert E. Balliew

Bob Balliew has contributed to the design of the ECL driver/comparator card for the HP DTS-70 Digital Test System, the design of the HP 3052A Data Acquisition System, and the production of the HP 3060A Circuit Test System since he began work at HP in 1977. A graduate of Colorado State University (BSEE 1977), he was born in Denver, Colorado. Bob lives in Loveland, Colorado, is married (his wife is an HP marketing engineer), and has one child. Outside of work, he enjoys working on his house and yard when not hiking or skiing.

14 Digital Board Test Subsystem

Michael A. Teska

A project manager for enhancements to the HP 3065 Board Test System, Mike Teska also designed the pin electronics for the HP 3065. With HP since 1979, his work has resulted in a patent application related to reducing potential damage caused by in-circuit testing. Mike was born in Ann Arbor, Michigan and attended the University of Michigan, receiving a BSEE degree in 1979. He and his wife are advisors for a church youth group in Loveland, Colorado, where they and their new son make their home. When not sailing his Hobie Cat on nearby Boyd Lake, Mike enjoys backpacking, woodworking, and playing guitar.

Matthew L. Snook

Involved with both R&D and production for the HP 3065 Board Test System, Matt Snook has been with HP since 1979. His contributions have resulted in two patent applications and one conference paper related to circuit board testing. Born in Mexico, Missouri, Matt attended the University of Missouri at Rolla, earning a BSEE degree in 1979. He lives in Loveland, Colorado and has many interests—modern music and dancing, skiing, scuba diving, camping, boardsailing, motorcycling, and travel.

20 In-Circuit Test Safeguards

Vance R. Harwood

Interested in data security and software design methodologies, Vance Harwood is an R&D manager at HP's Manufacturing Test Division in Loveland, Colorado. He holds a BSEE degree (1975) from the University of Colorado and has done graduate work in computer science and electrical engineering at Arizona State University. Before coming to HP in 1979, Vance worked on high-speed digital communications design and microprocessor-based controllers and instrumentation for two major defense contractors. At HP he has done software and hardware design for the HP 3065 Board Test System and its predecessor, the HP 3060. Vance is a member of the IEEE, was born in Denver, Colorado, and is active in church functions. He lives in Loveland, Colorado, is married and the father of two children, and is interested in skiing, tennis, and science fiction.

23 Digital Test Library

Randy W. Holmberg

Born in Los Alamos, New Mexico, Randy Holmberg studied electrical engineering at Colorado State University (BSEE 1978). After a year of graduate work in computer engineering at Purdue University, he joined HP in 1979 as a circuit test marketing engineer. He then transferred to the R&D lab to work on software for the HP 3065 Board Test System and now is a systems engineer for HP's Atlanta Sales Office. He has taught several courses on HP's circuit test products and is active in the Metropolitan Community Church of Atlanta. Randy enjoys swimming, hiking, and working on his new home in Marietta, Georgia.

25 Board Test BASIC

Mark A. Mathieu

A native of the upper peninsula of Michigan, Mark Mathieu studied computer science (BSCS 1979) at Michigan Technological University. After working for nearly two years as a systems analyst on a Univac 1100/80, he joined HP in 1981. Mark developed many of the compilers, Board Test BASIC statements and functions, and utility routines used in the software system for the HP 3065 Board Test System. A member of the ACM, he lives in Loveland, Colorado, where he recently bought his first home. Interested in almost anything athletic, Mark also enjoys occasionally reading a science fiction or fantasy novel.

28 Short-Circuit Testing

T. Michael Hendricks

A graduate of the Massachusetts Institute of Technology (SB 1969, SM 1969, and EE 1972), Mike Hendricks worked on radar signal processing and medical data base systems before joining HP in 1980 as an R&D software engineer. His contributions to the Shorts test for the HP 3065 Board Test System have resulted in one patent application. Mike was born in Baltimore, Maryland and is a member of the IEEE. Living in Loveland, Colorado, he enjoys running, swimming, ice skating, and acoustic guitar.

31 Improved Analog Test Generator

John E. McDermid

A native of Twin Falls, Idaho, John McDermid joined HP in 1969 after receiving an MSEE degree from the University of Alberta at Calgary. He also holds a BSEE degree awarded in 1967 by the University of Idaho. The program manager for the HP 3065 Board Test System, John also contributed to the HP 3490A Multimeter and the HP 3437A Digital Voltmeter and is the author of several papers on active filters and board test products. He is married, has a son and a daughter, and lives in Loveland, Colorado. Outside of work, John looks forward to fall when he can go bowhunting in the Rocky Mountains.

Automatic Test Program Generation for Digital Board Testing

by Robert E. Balliew

REDUCING THE AMOUNT of time and the level of programming skill required to develop a valid test program is a highly desired feature of a printed circuit board test system. Using a "fill-in-the-blank" form of component and board data entry, the in-circuit program generator (IPG-II) of the HP 3065 Board Test System automatically creates a custom analog and digital test plan for a board, cutting test programming time for complex boards from weeks to a few days.

This article discusses the digital test generation methods used by IPG-II. The analog test generation methods are discussed separately in the articles on pages 28 and 31. The major tasks done by the digital portion of the program generator are scanner card pin assignment for digital nodes, modifying digital library tests affected by the board topology, and disabling other devices on the same bus with the device under test. Typically, IPG-II can write more than 90% of the tests for a circuit board without any user intervention other than data entry.

The program generator writes tests for each digital IC on the board. Each digital IC's test is kept in a separate file. The digital tests are compiled and a statement in the HP 3065's Board Test BASIC language causes a compiled digital test to be loaded into the test head and executed. The HP 3065 System contains a library of digital tests which the program generator uses to create the digital test for each IC on the board. (This library is discussed in the article on page 23.) To simplify the system for the user, the same language is used to write new tests for the library and/or for direct execution.

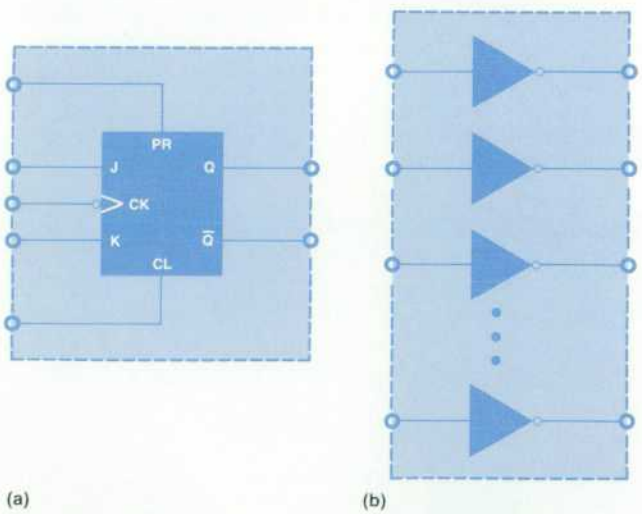


Fig. 1. *(a) Example of a board topology that cannot be efficiently assigned for in-circuit testing. Here, seven channels and a multiplexing ratio of 1:1 are required. (b) Example of a board topology that can be efficiently assigned. Here, only two channels are required and the multiplexing ratio is 1:N, where N is the number of inverters on the board.*

Fig. 2. *Configurations of a* NAND *gate used as an inverter.*

To reduce hardware costs, the digital stimulus/response channels in the HP 3065H Test Station are each multiplexed to a number of nodes on the circuit board under test (see article on page 14 and box on page 13). Obviously, the more nodes that can be multiplexed to one channel, the fewer the channels that are required for a given set of nodes and the lower the cost. However, there is a practical limit since each node of a single device under test must be assigned to a different channel. In addition, no node in the group of nodes assigned to a channel can require any of the other nodes in the group for any subsequent part of the board test sequence. In the HP 3065H, the multiplexing ratio is 1:4 and the task of assigning device nodes to channel groups is handled by the program generator so that there are no conflicts.

Assignment Algorithm

Before committing to a multiplexed digital hardware design, the feasibility of performing the desired test pin assignments had to be determined. Three things determine the efficiency of the pin assignment—the topology of the board (degree to which its ICs are interconnected), the assignment algorithm itself, and the multiplexing ratio. It is possible to postulate a board that cannot be efficiently assigned, namely a board with a single IC (see Fig. 1a). It is also possible to postulate a board that can be efficiently assigned, namely a board with N separate circuits (where N is greater than or equal to the degree of multiplexing, see Fig. 1b). All real boards must lie somewhere between these two extremes.

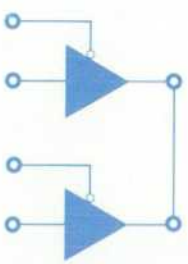
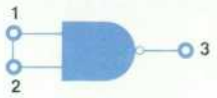


Fig. 3. *Example of a bused part topology.*

(a)

Vector	Pins		
	1	2	3
1	0	0	1
2	0	1	1
3	1	1	0
4	1	0	1



(b)

Vector	Pins		
	1	2	3
1	0	0	1
3	1	1	0

(c)

Fig. 4. *(a) NAND gate and library test table. (b) Use of NAND gate in (a) on circuit board under test. (c) Modified library test for topology shown in (b).*

An optimal assignment algorithm would be far too complex computationally, considering that a board could have in excess of one thousand nodes. The performance of an assignment algorithm is a function of how many different ways exist to assign the pins to the nodes without a conflict. To answer these design questions, two assignment routine variations were coded and run on over a dozen different real board topologies. Multiplexing ratios of 2:1, 4:1, 8:1, 16:1, and 32:1 were tried. The following general trends were observed in the data: all boards were assigned with good to very good efficiency (90% to 99%), the larger the board the more efficient the pin assignment, and the larger the multiplexing ratio the less efficient the assignment. Both variations of the assignment routine performed well. However, one had a slight advantage. One variation always attempted to assign pins so that existing channel groups were used fully. The other variation attempted to assign pins to channel groups in as balanced a fashion as possible. This second approach tended to leave openings in more groups, allowing more choices later in the process, which intuitively accounts for its slight efficiency advantage.

The assignment algorithm used by IPG-II consists of the following. First, the board topology is read in and then tests for each digital device are read from the HP 3065's device library. The program generator determines which test uses which nodes. The nodes are sorted in a manner that tends to make the pin assignment faster and more efficient. The program generator then loops through the list of nodes and looks for a pin location in the scanner to assign the node. If there are no tests in common between the tests required by the new node and the nodes already assigned to a multiplexed group, then the node is assigned to that group. Otherwise, the next group is checked. If no partially filled group is found in which a node can be assigned, then a new group is allocated. If upon completion more groups (channels) are allocated than the existing test-head hardware configuration can support, the user is informed of how many additional scanner pin cards are required. (The HP 3065 can contain up to 22 digital scanner cards, each providing 12 channels.)

Real World Problems

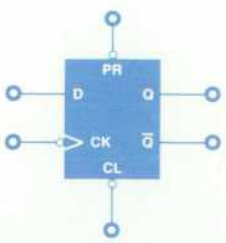
The basic concept of a digital in-circuit test system is to test each IC on the board as if it were a stand-alone part. However, since the ICs are actually installed on a printed circuit board, there are problems with simply testing the part as if it were a stand-alone IC. First, it is a common practice in digital design to change the function of a part by tying nodes together, tying a node high (logic one), or tying a node low (logic zero). For example, a NAND gate may be converted into an inverter by tying its inputs together or tying one input high (see Fig. 2). Another problem induced by having the IC installed on a board is that some of the ICs may be bused together (see Fig. 3). To observe the response of one device on a bus, all other devices on that bus must be disabled.

IPG-II has the intelligence to handle testing problems that might be encountered because of the above constraints. This reduces the effort required to write a new test for the library because the library test programmer need not be concerned with anticipating and handling all the possible various alternate topologies for a device. Instead, the programmer concentrates on writing a test that does a good job of testing the part and the program generator takes care of adapting the test for alternate topologies. For devices that can be disabled, all that is required in the device test library is the information on how to disable the device.

Digital test generation involves analyzing the board topology around each digital part on the board. Based on this analysis, the standard library program for each device is automatically modified by IPG-II to resolve conflicts between the topology and the library program and to disable other devices bused to the device under test. The topologies for which the program generator must modify the library test are nodes tied high, low, or unknown, unconnected nodes, and nodes tied together.

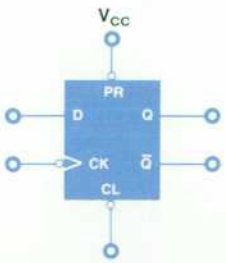
Resolving Topology Conflicts

All nodes that are tied high, tied low, tied unknown, and unconnected in the topology have their pin numbers replaced by a * in all assign statements in the digital library



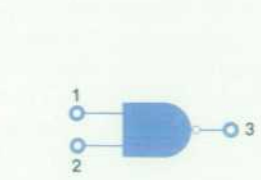
(a)

Vector Group	Pins D	CK	PR	CL	Q	Q̄
Unit 1	—	—	0	1	1	0
	—	—	1	1	1	0
Unit 2	—	—	1	0	0	1
	—	—	1	1	0	1
Unit 3	1	1	1	0	0	1
	1	1	1	1	0	1
	1	0	1	1	1	0
	1	1	1	1	1	0
	•		•			
	•		•			



(b)

Fig. 5. *(a) D-type flip-flop and library test sequence. (b) Use of flip-flop on board under test. For this topology, the library test is modified by removing test vector unit one.*



(a)

Disable 3 with 1 to zero
Disable 3 with 2 to zero

Vector	Pin		
	1	2	3
1	0	0	1
2	0	1	1
3	1	1	0
4	1	0	1

(b)

Fig. 6. *(a) Open-collector output* NAND *gate. (b) Library test and disable vectors for (a).*

program. The * pins are ignored when the test is compiled. This technique allows the program generator to handle nodes that are tied high, low, or unknown, or that are unconnected without requiring it to modify and reformat all the vector definition blocks in a test program.

The program generator looks at the test, checking each test vector separately for conflicts with the topology. If any pin setting is in conflict with the topology, then the vector containing it is in conflict, and the program generator will not allow the vector to be executed. If the test is a combinatorial digital program, then conflicting execute, preset counter, count, or call statements are deleted. If the library test is a sequential digital program, then the entire digital unit in the program (unit through end unit statements) containing a conflicting vector is removed.

To show how the program generator handles a typical test situation, suppose that the library contains a test for a NAND gate as shown in Fig. 4a. The board under test uses this part as shown in Fig. 4b. Hence, IPG-II modifies the library test by removing the execute commands for vectors two and four, resulting in the test shown in Fig. 4c.

Obviously the program generator cannot simply remove only the conflicting vectors from the test for a sequential device and still expect the test to work properly. To handle this situation, sequential tests are organized into units that are independent of each other. Each unit contains one or more vectors as shown in Fig. 5a, which lists a portion of the library test for a D-type flip-flop. As an example, consider the case where this device is connected as shown in Fig. 5b. Since this device test is sequential and the first vector in unit one is in conflict, IPG-II modifies the library test for this topology by removing unit one.

Disabling Buses

To test a device, all outputs of other devices bused to the device under test must be disabled so that the response of the device under test can be observed. The program generator disables devices by using one vector executed one or more times, depending on the devices involved

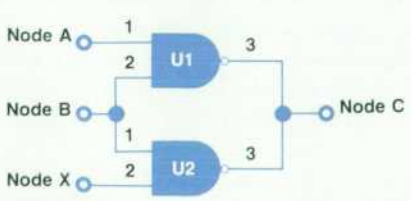
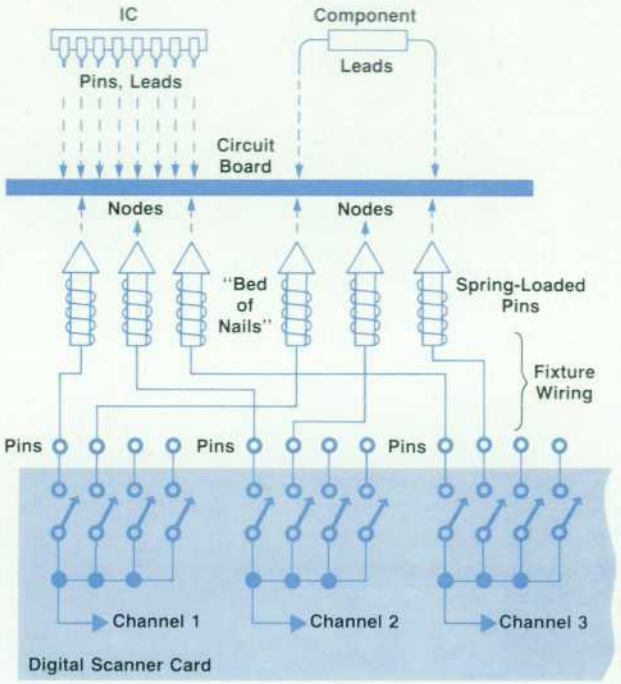


Fig. 7. *Bused arrangement of two of the parts shown in Fig. 6.*

Board Test Connection Terminology

The reader not intimately familiar with board testing may find the nomenclature used to describe circuit and test system electrical connections to be occasionally confusing. For example, the term "pin" can be used to describe a lead of an IC, the spring-loaded pin in the "bed-of-nails" test fixture, or an input/output connection on a digital scanner card. Hence, the figure below may be a useful reference.



(some may have to be clocked), and then left set for the duration of the device test. If the program generator cannot find a way to disable all the buses simultaneously without affecting any of the nodes of the device under test, it checks all the vectors remaining in the test for conflicts with the enabled device. If there is a conflict, the program generator asks the user to add the code to disable the buses manually. IPG-II will disable as much of each bus as it can.

For example, suppose that the device shown in Fig. 6a has a test in the library as shown in Fig. 6b. Fig. 7 shows two of these devices connected in a bus arrangement. When testing U1, the program generator must determine how to disable U2 without affecting the test for U1 and then modify the test for U1 so that it performs the disable vector. Fig. 8 shows the modified test. Note that the program generator can not use node B in Fig. 7 to disable U2, because that node is required by the test for U1.

IPG-II disables buses on a device by adding the nodes required to disable the buses to the device's test program. These nodes are specified in assign statements added to the declaration section of the device's test program. The value

Vector	Node			
	A	B	C	X
Disable	—	—	—	0
1	0	0	1	0
2	0	1	1	0
3	1	1	0	0
4	1	0	1	0

Fig. 8. *Executable test for U1 in Fig. 7 showing inclusion of the disable vector for U2.*

required for each node is specified in each node's assign statement. All nodes required to disable the buses are assigned to a set of pins named Disablegroup. The program generator also adds an input statement that specifies Disablegroup to the device's test program.

After all digital units and vectors that conflict with the topology around the device have been removed, the disable vector is placed above the first digital unit. The execute vector statements (or a repeat loop and an execute vector statement) are placed inside of a new unit. The execute statements refer to a vector called Disablevector which has no defining vector block. Disablevector is created by the compiler to set all the pins to their default settings. Disablevector may be executed several times before the test starts. The number of times it is executed is determined by the disable statements from which Disablevector was constructed.

Modifying Library Tests

IPG-II forms the executable digital test from the library source. This minimizes the number of machine-generated statements the source contains. Since it is possible for some of the tests to require some manual modification after the program generator has run, the more readable the test, the easier the modifications. The program generator puts comments in the test programs it generates to help the user. Most of the comments start with ! IPG: to help flag them as comments placed into the source by the program generator. One comment documents which revision of the software created the test and what day and time the source file was created. Another comment informs the user which devices and nodes must be disabled before a device can be tested. In most cases, the program generator will not list every device and node that was not disabled but will list the key devices and nodes that, if disabled, will assure that no device bused to the device under test is enabled (this is because of a common enable for many outputs). If IPG-II removes a unit from a device test, it generates a comment informing the user that a unit was removed from the test. Whenever IPG-II finds a conflict between the topology around a device and a vector, it generates a comment informing the user of the conflict. If a test unit was removed because of this conflict, the conflict comment appears right below the removed unit comment; otherwise, it appears at the location of the removed vector's execute, preset, count, or call statements. The comment shows the name of the vector and the pin name or node name that has the conflict. Pin names are shown in cases where the conflict is with a node tied high, low, or unknown, or unconnected. Node names are shown if pins tied together are in conflict. The comment also indicates the reason for the conflict—conflict with pin tied high, conflict with pin tied low, conflict with pin tied to an unknown value, conflict with an unconnected pin, and conflict between pins tied together.

Acknowledgments

Tom Fay provided the library access routines and many helpful suggestions about testing and structuring the HP 3065's in-circuit test program generator. Bill Groves, who helped in the definition of IPG-II, developed the pin assignment algorithm and provided much insight about algorithm design and efficiency.

Rod Browen coded the analog portion of IPG-II, including writing of the test plan and the Shorts test.

Digital Subsystem for a Board Test System

by Matthew L. Snook and Michael A. Teska

IN ANY PRODUCT DEFINITION, it seems that a major issue is how to divide features into software and hardware implementations. However, often certain product features can only be realized in hardware because of their very nature. The digital hardware subsystem for the HP 3065H Test Station of the HP 3065 Board Test System was designed to address universal test needs in the areas of fault coverage, cost, performance, ease of operation, and throughput in addition to a multitude of specific digital in-circuit test needs. These needs include long complex pattern sets, a high pattern rate to meet the minimum clock rate required by dynamic devices, short test duration to minimize the potential for damage to overdriven devices, high throughput to minimize testing costs, and a set of test vectors known as a homing sequence to initialize some sequential devices to a known state before testing.

A traditional architecture approach might have addressed most of these goals by simply using more high-speed RAM behind the test pins to store a greater number of test vectors. However, this leads to higher costs and to lower system throughput because of the large amount of test pattern data to be downloaded, which is by far the controlling factor in determining throughput. The HP 3065H architecture uses one local high-speed control card

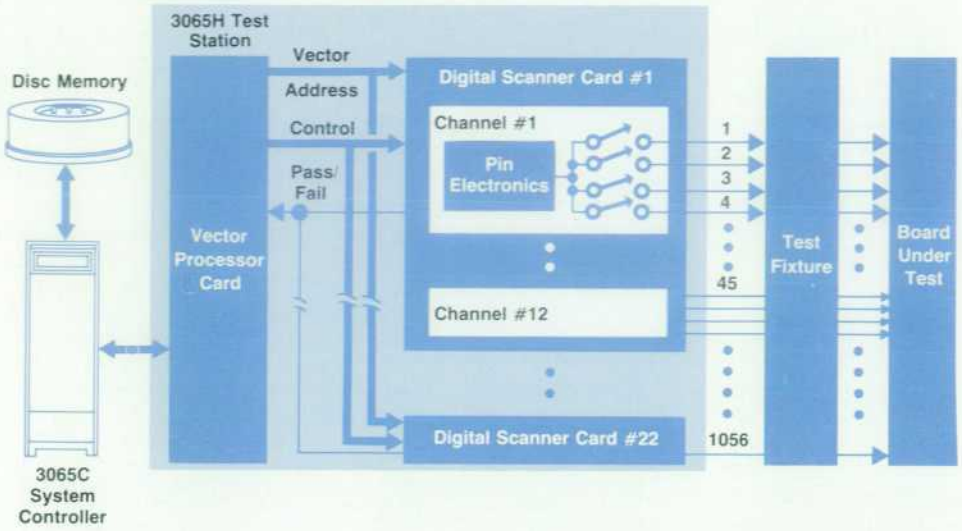


Fig. 1. *Architecture of the HP 3065 Board Test System.*

to sequence the proper execution of test vectors residing on several digital scanner cards, each containing RAM-backed digital drivers and receivers.

Board Test System Overview

The basic architecture for the HP 3065 Board Test System is shown in block diagram form in Fig. 1. The HP 3065C System Controller runs the board test program that sequentially tests each device on the circuit board under test. Within the HP 3065H Test Station is a single high-speed control card, called the vector processor card, and several digital scanner cards. Each driver/receiver channel is multiplexed through relays and wiring to four spring-loaded test pins in the "bed-of-nails" test fixture designed for the board under test (see cover photograph). A vacuum in the test fixture assembly pulls the board under test down to connect its circuit nodes to the spring-loaded pins. There are twelve channels per digital card and up to 22 of these cards can be installed in the HP 3065H, resulting in up to 264 channels multiplexed to 1056 nodes on the board under test.

By placing a great deal of emphasis on the capabilities of the high-speed control card, the HP 3065 Board Test System was designed to perform long complex pattern sets of both an algorithmic and manual nature, do real-time pass/fail checking off-line from the host system controller, provide pattern rates up to 2.5 MHz, and much more. Throughput limitations imposed by the time required for memory accesses are eased by downloading only a small amount of information and allowing the high-speed control card to create the long complex patterns programmatically.

The HP 3065C Controller downloads all of the test pattern data for a specific device test from disc memory to the vector processor card and digital cards. This data includes the vector data for the pin RAM on the digital cards, the sequence program for the program RAM on the vector processor card, and other specific test parameters to be set up on various cards. The vector processor card then takes over control and locally executes the program to test the device, giving pass or fail results to the controller when finished. This allows the controller to perform other tasks during test execution, which increases system throughput.

To execute a particular vector, the vector processor card issues the proper vector address to all pin RAMs, accesses the test vector data consisting of drive data and expected response data, clocks the vector data, waits a programmable receive delay time, samples the response data, and then examines the pass/fail result indicating whether all of the responses were correct or not.

Digital Subsystem Architecture

Fig. 2 shows an expanded view of the digital subsystem architecture with a more detailed block diagram of the vector processor and digital scanner cards. The vector processor consists of a 4K × 32-bit RAM with 32-bit instruction words which are addressed by the sequencer and clocked into the pipeline register for execution. The sequencer controls execution of program flow by doing conditional and unconditional branches. The ALU (arithmetic logic unit) and registers are used for repeat loops, parameters for subroutines, and algorithmic counter generation. The priority encoder and indirect address RAM are used for algorithmic counter generation as explained later. The multiplexer selects one of three sources to be used as the vector address depending on whether the vector is a simple vector, a parameter vector, or an algorithmic counter vector. The timing and control block consists of a two-phase 10-MHz clock circuit and an algorithmic state machine that control the entire vector processor card program flow. The timers measure the programmable time relationships of the clock and control signals sent to the digital scanner cards.

After a vector address is issued to the digital cards, the vector data is accessed from the 1K × 4-bit test pin RAM and clocked into a J-K flip-flop. The two bits clocked into the J-K flip-flop can indicate one of four possibilities: 0, 1, keep (K), or toggle (T). Keep means that the next state is the same as the present state and toggle means that the next state is the opposite of the present state. The state of the J-K flip-flop indicates the expected response or the drive state that the driver will drive. The third bit in the pin RAM is for driver enable to turn the driver output on or off (high impedance). The last bit is the mask bit used

Digital Test Throughput

Testing digital devices on a system such as the HP 3065 Board Test System involves applying a set of digital stimulus signals to the device to be tested and verifying that the digital responses generated by the device are correct. A typical test involves a sequence of these stimulus/response patterns, called vectors, to test for a variety of manufacturing defects that might affect the correct operation of the device on a printed circuit board (Fig. 1).

The trend, as devices become larger and more complex, is for the corresponding tests to grow similarly. This means that tests involve more pins (increased vector width) and more vectors (increased vector depth). In addition, users frequently wish to add functional subtests to the device tests, again resulting in larger digital tests.

Test size can be viewed as the product of the width of the vectors (the number of device inputs and outputs) and the number of vectors in the test. For a typical TTL part, this doesn't look too alarming—16 pins times 200 vectors requires 3200 bits of information to specify the test. However, as devices get larger, the amount of information gets quickly out of control—64 pins times 2000 vectors requires 128,000 bits of information. The problem is compounded by the fact that there are increasing numbers of such devices on a typical circuit board, and each device must have its own test. All of this information must be created, stored, retrieved, and transferred to the test-head electronics as part of the generation and execution of digital tests.

The HP 3065 addresses this potential testing bottleneck with a combination of hardware and software features. One goal of the hardware design for the HP 3065 was to reduce the amount of information needed to generate the vectors making up the test. The design was guided by a number of observations. The first observation is that the vectors applied to the device under test generally involve a fair amount of redundancy. The same vector pattern is frequently used in various portions of the test. The same sequence of vectors, with possibly a few vector differences, is also repeated frequently within the same test. The HP 3065 takes advantage of this by separating the storage of the vectors from the sequence of execution information. Unique vectors are stored in vector RAMs, and a special processor, the vector processor, controls the sequence of application of these vectors to the device. This processor permits repeat loops, subroutines with parameters, and repeated use of the same vector.

A second observation is that the vectors themselves often have a great deal of similarity. The HP 3065 takes advantage of this by permitting vectors to be specified in terms of the changes (toggles) or nonchanges (keeps) from the last vector executed (Fig. 2). As an important special case, vectors are often the same except for a particular subset of the pins that have a count associated with them—address pins are a typical case. For this case, the HP 3065 has the capability of generating these "count" vectors algorithmically, with the result that literally thousands of vectors can be generated using very compact tests. This count capability is particularly powerful when coupled with the ability of the HP 3065 to calculate a cyclic redundancy code (CRC) checksum for the device outputs over all the vectors applied to the device under test. Using this capability, extremely compact tests can be created for ROMs by using the count feature to generate addresses and the CRC capability to specify the ex-

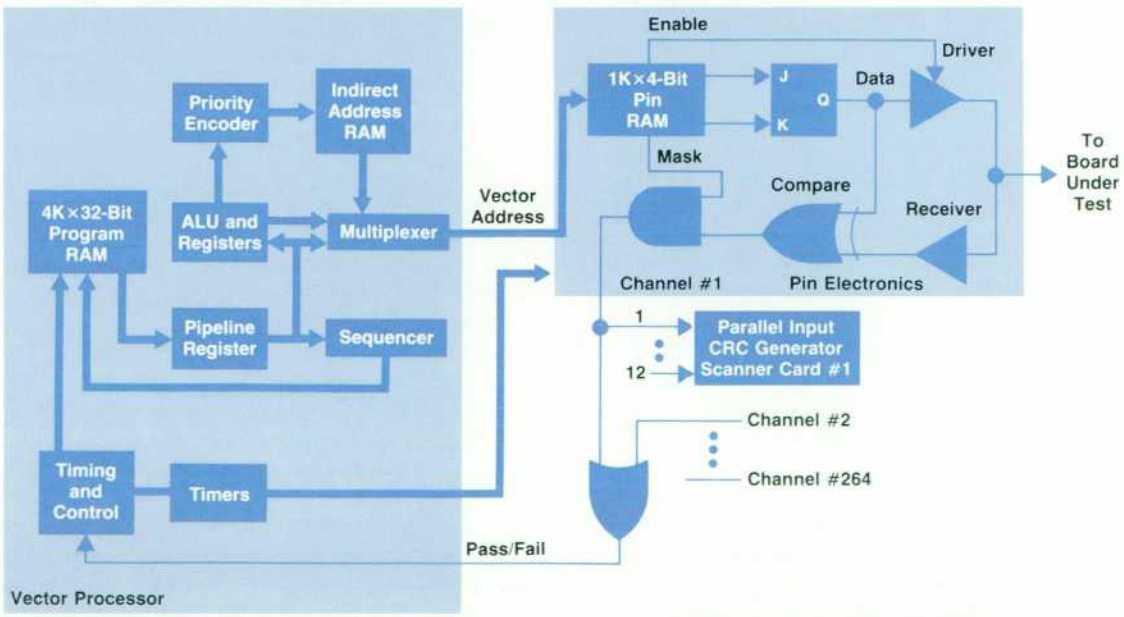


Fig. 2. *Digital subsystem architecture of the HP 3065H Test Station (see Fig. 1) Although only the pin electronics block diagram for one test channel is shown, there are 12 channels on each digital scanner card and the HP 3065H can accommodate up to 22 cards or 264 channels.*

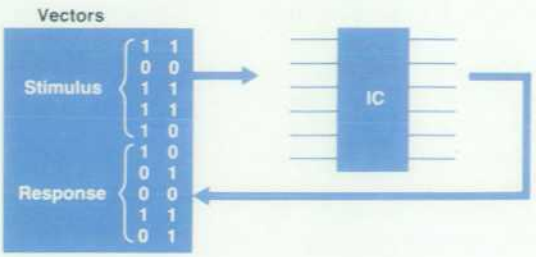


Fig. 1. *Digital in-circuit testing is defined by vectors that describe the input test pattern and the expected response pattern for a device.*

Unique Vectors	+	Test Sequence	=	Test Patterns
1: 0000		1		0000
2: TTKK		2		1100
3: KTKT		3		1001
		2		0101
		3		0000
		1		0000
		3		0101
		2		1001

Fig. 2. *By using keep and toggle descriptions, the number of vectors required to describe a long set of digital test patterns can be greatly reduced, increasing test throughput and reducing data storage size. In the above example, three unique keep/toggle vectors can be used to form a test sequence of eight test patterns.*

pected ROM data contents concisely.

Even with the advanced architecture of the HP 3065 hardware, the amount of data associated with testing the digital parts on a board can be significant. The primary bottleneck associated with digital testing is still getting each test from the system controller's disc memory to the test hardware. For this reason the HP 3065's software is specially designed to enhance speed through this bottleneck by tuning the software and data organization to support the normal way that digital testing is done for a printed circuit board. The key is that the typical board test will test all devices in a specific sequence that does not vary. To take advantage of this, digital tests are stored in a specially organized file called the turbofile. This file is built to occupy contiguous blocks of the disc and contains the tests in the order that the board test will need them. The software to execute the digital tests uses a large buffer in memory for reading in the tests so that as testing proceeds each disc access reads in a number of digital tests. By doing this, disc access overhead can be shared over a number of tests, reducing dramatically the overhead per test caused by disc accessing. The advanced hardware architecture permits compact tests that make this scheme feasible. As an example, a disc access might take 40 milliseconds on the average. Because of the HP 3065's software structure, a block of 20 tests can be read during one access, resulting in 2 milliseconds of overhead associated with each test instead of 40 milliseconds of overhead per test if each test were read from the disc individually. A second part of tuning the software for increased throughput is to make maximum use of the DMA interface between the system controller and the test-head electronics. Tests in the turbofile are set up to require no processing at run time before invoking the DMA transfer from the controller to the test head, and these tests are further designed to perform relay closure operations that require settling delays at the beginning of the DMA operation. This overlaps the downloading of the test's data with the delay required for the relays to settle. Thus, a typical digital test gets loaded into the test head while relays are closing and the download time is typically masked entirely by relay closure time.

Acknowledgments

Vance Harwood and Bill Groves provided the leadership that resulted in the software that makes the HP 3065 go.

Thomas R. Fay
Project Manager
Manufacturing Test Division

when the response is not to be tested. Therefore, there are nine basic states that the vector data can indicate: drive 0, drive 1, drive keep, drive toggle, receive 0, receive 1, receive keep, receive toggle, and X (don't care—receiver masked and driver off). The concept of keep/toggle vectors is a key feature in being able to generate a large number of test patterns from a small amount of downloaded data (see box on page 16).

All channels used for a single device test (up to 264) have their compared responses ORed together to create a single pass/fail line which is monitored in real-time by the vector processor card after each vector. This pass/fail result can indicate a truly failing vector, or can be used as a test condition for a conditional jump statement at the end of a homing sequence loop (e.g., "jump" out of the loop if "passed").

Each digital scanner card also has all twelve channels of compared response fed into a parallel input cyclic redundancy check (CRC) generator on that card. The CRC generators on all cards are simultaneously clocked with input data on selected vectors until the test is complete and a signature is generated from the bit stream. Thus, all outputs of a device can be captured with only one application of the test stimulus which, for example, is an ideal way to verify the contents of a large ROM.

The design of the vector processor card includes features that help execution speed such as one level of pipelining instructions, two-phase 10-MHz clocks to allow two sets of circuitry to communicate with each other at a high rate, the prefetch of instructions and execution when possible, a special algorithmic state machine that executes each instruction as quickly as possible (rather than all at the rate of the slowest one), and a minimal instruction set customized for this application. By using generous portions of conventional high-speed RAM and bit-slice processor components, these features were achieved without the need for custom devices.

The vector processor card supports other vector sequencing hardware features that include subroutines with up to 16 vector addresses passed as parameters, repeat loops up to a count of 65,536 times with up to 16 levels of nesting, and algorithmic counter generation of up to sixteen simul-

taneous 16-bit counter fields.

The fact that the vector data is stored in the test pin RAM and is accessed by sending a vector address from the vector processor card means that the same vector may be accessed repeatedly during a test. This reduces the number of vectors that have to be stored for a test and thereby reduces the amount of data to be downloaded for a test, which increases throughput.

Keep/Toggle Vectors

Keep/toggle vectors allow the next state of a test pin to be determined by the current state. It can be observed that for a set of N-bit-wide patterns in which only one bit changes between successive patterns, there exists a set of only N unique keep/toggle vectors that can be used to describe any such change. For example, an 8-bit Gray code counter in which only one bit changes between successive patterns would then require only eight unique keep/toggle vectors in which only one bit toggles for each new vector and the rest are kept the same. See box on pages 16 and 17 for another example.

This simple but effective concept is used by the hardware to create a counting sequence up to N bits wide algorithmically on any set of test pins by simply accessing the N unique vectors in the proper sequence. The reduction in downloaded data is substantial since the creation of an N-bit counter that can count through 2^N states requires only N unique keep/toggle vectors plus one initialization vector. In the HP 3065H Test Station, N is a maximum of 16. Hence, 65,536 test patterns can be created from only 16 keep/toggle vectors. Other algorithmic counter sequences are also implemented just as simply by using a different set of unique keep/toggle vectors, including a binary up-counter and down-counter.

An example of the dramatic data reduction is the library test for a 2K × 8 PROM that tests all 16,384 bits of the PROM by algorithmically generating 2048 test patterns from only 12 unique downloaded vectors, resulting in a downloaded data reduction of 171 to 1.

The beauty of this architecture is that the same pin electronics hardware and relay multiplexing are used for all of these features. Any test channel can be bidirectional on the fly, can change at the maximum pattern rate, can be an algorithmic counter pin or simple vector pin, etc.

Digital Scanner Card

Digital signal quality was one of the primary design considerations in the design of the digital scanner card. At issue is the quality or fidelity of the signals at the device under test (DUT), and hence, the quality of the test.

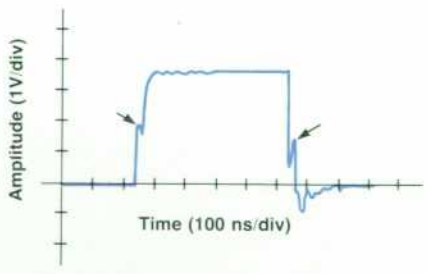


Fig. 3. *Digital test stimulus waveform with glitches (indicated by arrows). If not eliminated or reduced, these glitches can cause erroneous test results.*

Before we get into details on the methods to achieve good signal quality, let's first examine what will happen with poor signal quality. Some of the possible testing problems caused by poor signal quality are:

- Double clocking
- CMOS latch-up
- Oscillations
- Nonrepeatable tests.

These testing problems are caused primarily by glitches, voltage overshoot and undershoot, and slow edge speeds (rise and fall times) in the stimulus applied to the DUT.

Glitches are the most prevalent and most difficult problem to solve (see Fig. 3). They can be responsible for the double clocking of a circuit (a counter, for example), causing the test to fail. Glitches can also lead to unrepeatable tests when they are caused by other events during the board test that may not occur each time the test is run (for example, homing loops may be executed a different number of times depending on how the board powers up). These glitches may appear on control lines and cause a part to reset or perform other unintended operations.

Glitches can be caused by crosstalk in the test fixture or in the test system if proper care is not taken in the test system design or the test fixture wiring. They can also be caused by capacitive terminations at the end of a transmission line (which the test fixture contains, regardless of the wiring scheme used).

Overshoot and undershoot occur when a transmission line (the wires in the test fixture) is terminated with an impedance greater than the characteristic impedance of the line. (This is very common when testing CMOS, whose impedance is typically about 500Ω). The size of the overshoot and undershoot increases with the edge speed (rise and fall times) of the source (in this case, the board test system). Of course, overshoot and undershoot can also be caused by poorly designed pin scanning electronics.

Very slow edges at the DUT can cause the DUT to oscillate. This occurs because the input of the DUT is held in its switching region (for TTL, between 0.8V and 2.4V) for a long enough time to let noise on the input (from the output switching) cause the device to oscillate. If the impedance of the source driving the input is sufficiently high, the noise may be so severe that even when the source reaches its steady-state value, the input may still be moving through the switching region, causing the oscillations to continue. As before, the design of the pin scanning electronics also influences the edge speed of the test stimulus.

The digital scanner cards are designed to reduce these problems. The most significant feature of these cards is the use of microstrip transmission lines. The printed circuit board is done in such a manner that the traces connecting the pin electronics to the test fixture have a characteristic impedance of 90Ω. The test fixtures that interface the HP 3065H Test Station to the DUT (see Fig. 4) use twisted-pair cable which also has a characteristic impedance of 90Ω.

There are several advantages to using twisted-pair cable. Crosstalk is reduced because of the shielding effect of the twisted-pair ground wire and the mutual inductance be-

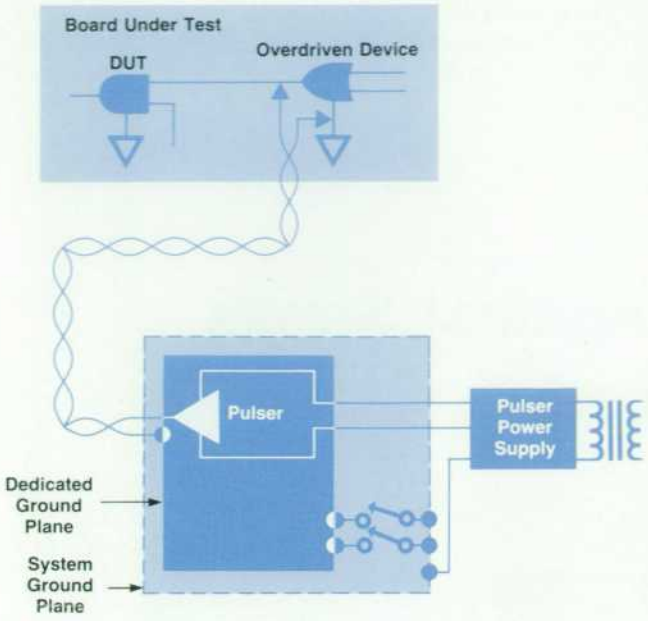


Fig. 4. *Test wiring configuration for the HP 3065H Test Station. Twisted-pair cable is used to connect the HP 3065H to the board under test.*

tween the two wires of the twisted pair. Initial placement of the wires and subsequent moving of the wires when modifying a fixture do not cause the characteristic impedance of the twisted pair to change. Since the DUT waveforms are determined by the interaction of the load and the transmission line (twisted pair), and the transmission line does not change, the waveforms are very repeatable. The 90Ω transmission lines also present a low enough impedance to the input of the DUT so that any noise coupled to it from its output switching is small and does not cause it to oscillate.

The use of twisted-pair cable greatly reduces common mode coupling. This is because, in the ideal case, all the current traveling up to the DUT on the signal wire is returned on the ground wire of that twisted pair. Hence, there are no common paths in the ideal case. To approximate the ideal case as closely as possible, the twisted-pair grounds are all taken directly to a dedicated ground plane on the digital scanner card. This dedicated ground plane assures us that the ground currents of the pin electronics and associated circuitry do not cause common mode coupling onto the signal ground. The dedicated ground plane is connected to system ground through two mercury-wetted relays (see Fig. 4). To reduce the chances of common mode coupling further, the dedicated ground plane for the twisted pairs connected to a digital scanner card is disconnected from system ground when the pin electronics of that card are not in use. Thus no current can return through the grounds of those twisted pairs. To reduce the currents returned to other digital scanner cards with active pin electronics, a dedicated overdriving power supply is associated with each digital scanner card. This supply floats, its only ground connection being to its own digital scanner card. Thus, the overdrive current supplied by a digital scanner card must return to that card. Since the return path through other digital scanner cards is of higher impedance (because of trace lengths, etc.) the current returns primarily to the card that supplied it.

Common mode coupling on the DUT is also reduced by wiring the other end of the ground wire of the twisted pair to the ground pin of the device being overdriven (not necessarily the DUT) and the signal wire to the output of the device being overdriven (not necessarily the DUT). This reduces the possibility of overdrive currents summing on the DUT ground planes and causing common mode coupling, thereby creating unrepeatable tests.

There are a few disadvantages to using twisted-pair cable. The digital scanner cards are designed to minimize these. As mentioned earlier, capacitively terminated transmission lines can cause a glitch at the DUT. This is most commonly found when the DUT is driving the test system (that is, the HP 3065 is receiving a pattern from the DUT). In this situation, the driver is disabled (if this is a bidirectional node) and the receiver has a high-impedance input. This is necessary so that the test system does not load the DUT. However, the driver and the receiver have capacitances associated with them. The digital scanner cards reduce this capacitance by careful design of the test pin electronics. The actual driver and receiver modules use surface-mounted parts. This reduces trace length and moves the circuitry away from the ground plane, thereby reducing the capacitance. To reduce the effect of the remaining capacitance, a small amount of inductance is put in series with the driver and receiver to isolate the capacitance.

Another potential disadvantage is the possibility of overshoot and undershoot when terminating the twisted-pair cable with an impedance greater than 90Ω. Recall that this is the case for CMOS. The HP 3065 reduces this problem by giving the user the choice of two edge speeds at which to run tests. The slow edge speed (40 V/μs into 90Ω) is designed specifically for CMOS testing. Here the edge speed has been reduced enough to reduce overshoot and undershoot to a level that greatly minimizes the possibility of CMOS latch-up.

One final item that should be mentioned is that the length of the twisted-pair cable is directly proportional to the propagation delay of the transmission line. The longer the line (and hence the longer the propagation delay), the slower the edge speed becomes and the longer the DUT must drive the characteristic impedance of the line. Glitches caused by capacitive terminations also increase in size as the line becomes longer. Hence, it is advantageous to keep the line lengths as short as possible. The digital scanner card has been designed so that the test pin electronics are very close to the test fixture connector. The maximum distance from the connector to the pin electronics is about 10 inches. This minimizes the propagation delay of the transmission line and allows higher-speed testing and high-quality signal waveforms.

Acknowledgments

The initial digital subsystem architecture of the HP 3065 was defined with the help of several key people, beginning with John McDermid, Bill Groves, and Bill Nicolay. Their guidance along with that of Al Gookin laid the framework for the hardware implementation phase of the project, in

which many dedicated people, too numerous to list here, were involved. Credit and thanks goes to them for their contributions.

Special thanks to G. Sivia Bushanam for his extensive work on transmission lines and their applications in the HP 3065 and to Al Gookin who defined the grounding scheme.

Safeguarding Devices Against Stress Caused by In-Circuit Testing

by Vance R. Harwood

AN IN-CIRCUIT DIGITAL TEST SYSTEM is a bully. When the tester wants a device input low, it forces the input low. It doesn't matter if another logic device is trying to drive the device input high, the test system always wins. Being a bully has some significant advantages, such as easy test development and good fault diagnostics. However, this heavy-handed approach can stress a logic device in a manner that is usually not explicitly allowed by the device's operating specifications.

Understanding the severity and impact of these stresses was a key issue that faced the HP 3065 Board Test System design team when our investigations into digital in-circuit testing began. A team of engineers was assigned the task of understanding the impact of in-circuit digital testing on logic devices and recommending a course of action to the rest of the design team.

The group amassed a list of possible damage mechanisms and analyzed them one by one. Some damage mechanisms such as electromigration and dielectric breakdown were found to be insignificant in regard to in-circuit testing. Three items on the list—CMOS latch-up, bond-wire heating, and junction heating—seemed to have the potential to damage devices during a test, so these became the focus of attention.

Most digital in-circuit test systems use high-current drivers to force the inputs of the device under test to the desired test condition. These drivers are connected to the board

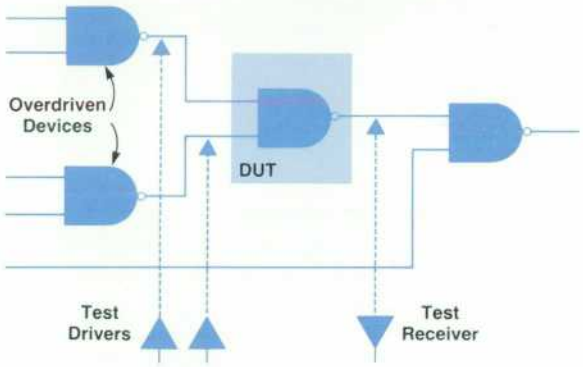


Fig. 1. *In-circuit test connections to a* NAND *gate.*

under test by an interface that is customized for a specific board. The interface is called a test fixture (see cover photograph) and usually consists of two plates, one that has connections to the test system and another that has spring-loaded pin connections (bed of nails) to the board being evaluated. These two plates are connected by wires of some sort that can vary in length from a few inches to several feet.

The inputs to the device under test are usually connected to the outputs of other logic devices, but sometimes a logic device has connections between its own outputs and inputs. The device under test is not the one being overdriven, but rather some outputs that are electrically connected to

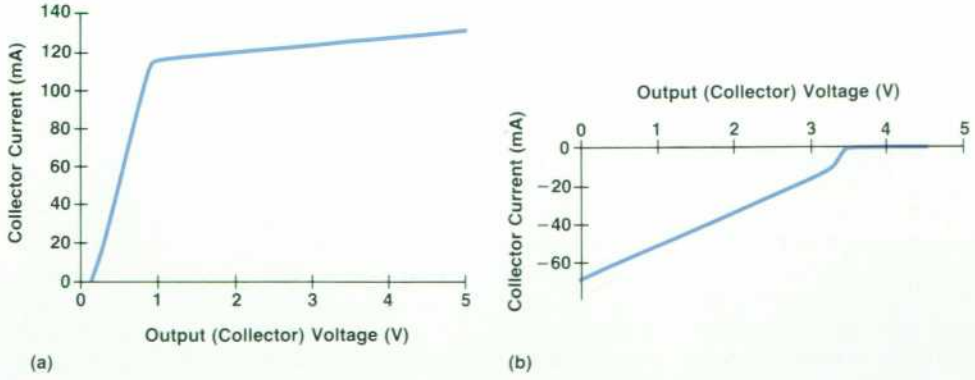


Fig. 2. *Overdrive characteristics of a typical TTL LS digital device, a 74LS04 (see Fig. 3). (a) Forward characteristic for forcing an output high when the output is zero. (b) Reverse characteristic for forcing an output low when the output is a one.*

the logic function of interest. Fig. 1 shows the connections for a simple NAND gate test. A worse-than-average situation might occur when the eight inputs of a logic device are connected to the eight outputs of an octal buffer. The outputs of the buffer could conceivably always "agree" with the test system drivers, but the conservative approach is to assume that the drivers and involved logic outputs are always "disagreeing." The curves in Fig. 2 show how much current is required to overdrive each output of a typical TTL LS (low-power Schottky) part to a one or a zero level. The worst-case drive current is always for the logic one case, and can exceed 500 mA for some of the newest logic families.

The result is that output transistors that might dissipate ten milliwatts in normal operation are called upon to dissipate close to a watt for short periods of time. A typical digital in-circuit test on the HP 3065 Board Test System takes less than two milliseconds. Within that test period, most driver outputs do not stay at the same value, but rather change depending on the test requirements. Again the conservative approach is to assume that a logic output is overdriven to the worst-case level for the duration of the test. Fig. 3 shows that the outputs of our example buffer cannot be treated as being completely independent. The output transistors all share a piece of silicon a tenth of an inch square. Silicon's thermal conductivity is similar to copper, which means that heat flows readily through the chip. The output stages of the buffer all share the same power and ground connection. This means that the bond wires that connect the chip to the package leads must carry the sum of all of the output currents. These bond wires are made of gold or aluminum depending on the package type and are about one thousandth of an inch in diameter.

CMOS Latch-Up

The latch-up problem occurs with CMOS logic families. If any input or output of a powered-up CMOS part has a voltage more than 500 millivolts above or below the power supply voltages, the device may self-destruct. This occurs because the CMOS transistor structure forms a parasitic silicon-controlled rectifier (SCR) as shown in Fig. 4. If this

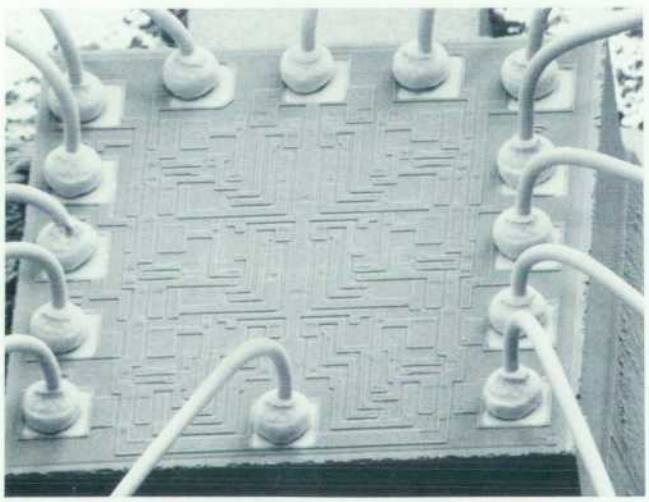


Fig. 3. *Photomicrograph of a 74LS04 digital IC chip and bond wires.*

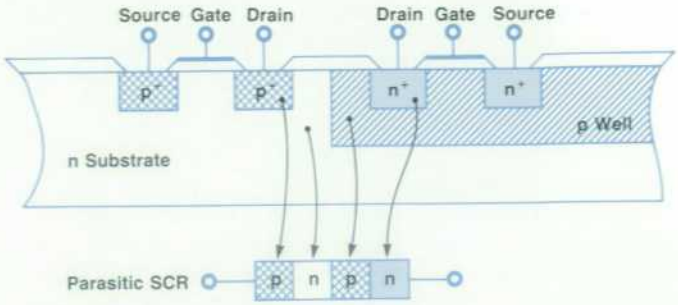


Fig. 4. *Cross section of a bulk CMOS device structure showing the parasitic SCR structure that can cause latch-up.*

SCR is turned on by excessive voltage on the I/O pins, the device may destroy itself by drawing too much current. The obvious solution for CMOS device protection is to limit the voltage swings impressed on the board. However, the combination of high-performance drivers and long wires in the fixture can still lead to excessive voltage swings at the device during in-circuit testing. The solution to this problem is straightforward. The HP 3065's drivers have a selectable slew rate which limits the voltage swings to a safe range when used with twisted-pair test fixture wiring.

Bond-Wire Heating

Unfortunately it was not possible to eliminate the bond-wire heating problem with a hardware solution. When high currents are passed through the bond wires, they heat up from resistive heating and if the current is sustained at a high enough level, the wires will melt. The current through each bond wire is not a controllable parameter, because it is a function of the device being overdriven and the desired voltage level. The only practical way to limit the temperature of a bond wire is to limit the length of time that current flows in the wire.

A practical solution required that the team understand the heating behavior well enough so that bond-wire temperature could be limited to a safe level without unduly limiting the allowed test durations. The problem was initially modeled using an electrical analog approach. Later, the result was double-checked with a finite-element simulation and finally with a closed-form analytical solution.

A technique that measures the average temperature of sample bond wires was used as an experimental verification of the model. Fig. 5 shows the time-versus-current restrictions that limit the peak temperature of an aluminum bond wire to less than 250°C.

The next task was to incorporate this protection into the HP 3065 Test System. A simple hardware timeout was unacceptable because any single timeout value would be either too short or too long, depending on the current required for each test. The solution selected validates each test before it is run on the test system hardware. The maximum current in each of the overdriven packages is predicted with the aid of device information contained in the HP 3065's device test library and the maximum test duration is computed by the system's software. The software is designed so that the test duration is minimized. For example, initialization sequences have timeouts that

fail a part as soon as it is apparent that the part will not initialize. If the combination of test time and current would cause more than the allowed temperature rise, the programmer is informed that the test has been rejected and will not be executed.

Junction Temperature

Limiting the junction temperature of the overdriven output transistors turned out to be a two-part problem. The power dissipated in the junction is obviously important, but there are sometimes other outputs on the same chip that have to be considered too. The junction heating problem turned out to be similar to the bond-wire problem in that the only practical way to control the temperature is to limit the duration of the test.

The first pass at modeling the junction heating behavior used an electrical analog method of modeling. Direct temperature measurement of the chip using internal protection diodes as thermal sensors showed that this approach was too simplistic to describe the thermal behavior adequately. The next pass at the problem used finite-element simulation to predict the peak temperature of the junction. The results of the simulation were surprising; it predicted that the junction would exhibit considerable heating within the first microsecond of the overdrive condition. It also predicted that the temperature dropped off rapidly with distance from the junction. Fig. 6 shows a typical temperature-versus-time curve for single and multiple overdriven outputs. Notice how the number of overdriven outputs on the chip is unimportant until approximately one millisecond has passed. These results were verified by an analytical closed-form solution of the problem and by temperature measurements taken within 15 μm of the output transistor.

The HP 3065's software limits junction heating with almost the identical approach used to protect against bond-wire heating—by prescreening the tests to be run. But for junction temperature, the critical parameter is power density instead of current and the thermal contributions from other outputs must be added. Probably the most important difference between bond-wire and junction heating is the difference in cooling characteristics. Bond wires have very small thermal masses and hence cool rapidly. Junctions

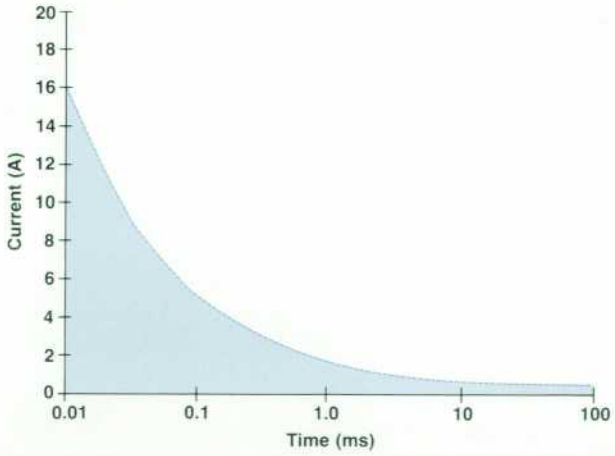


Fig. 5. *Graph of safe operating region (under curve) for overdrive current versus test duration to limit bond-wire heating.*

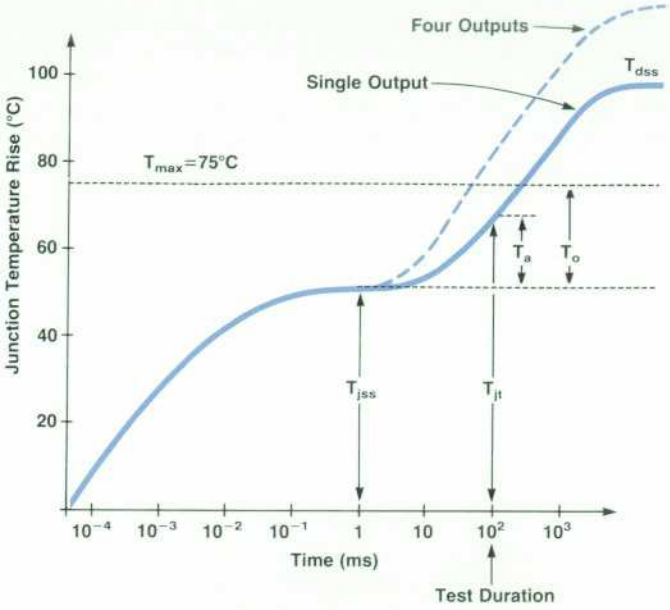


Fig. 6. *Graph of transistor junction temperature versus time for a typical 74LS04 IC. Solid curve is for a single overdriven output. Dashed curve is for four overdriven outputs. (Curves are the same for periods less than one millisecond).*

are effectively heat-sinked by the chip and its package, but if heated, these thermal masses require substantial cooling times. The software predicts how much heating will occur and inserts pauses between tests when needed to allow for cooling.

As a final confidence check of the HP 3065's safeguard measures, hundreds of parts of different processes, part numbers, and manufacturers were overdriven within the constraints of the overdrive guidelines and then checked for problems. Some of the parts were burned in for 1000 hours at 125°C followed by 1500 hours at 150°C to check for any lifetime degradation mechanisms that might have been missed. The results of the tests confirmed that the devices were being kept within their safe operating limits.

Acknowledgments

Although a large number of people contributed to this investigation, the author would like to single out Elton Bingham, G. Siva Bushanam, John Ketchum, and Roger Story for their creative solutions to many of the problems described.

Extensive Library Simplifies Digital Board Test Setup

by Randy W. Holmberg

THE LARGEST COST involved in owning a circuit board tester is the cost of test programming. When testing in-circuit digital devices, this cost can be significantly reduced by using a library of preprogrammed tests. Having device information and test safeguard data readily available in a permanently stored library makes it much easier for a test engineer to set up a test sequence for a printed circuit board.

One of the major design objectives of the HP 3065 Board Test System was to make ownership economical. For this reason, development of a large, comprehensive device test library was crucial to the success of the project.

Design Objectives

The device library provided with the HP 3065 Board Test System covers the TTL, CMOS, and ECL logic families and includes over 2700 SSI, MSI, and LSI devices. Because of the large number of SSI and MSI devices on a typical printed circuit board, the library design team concentrated most heavily on these devices. The LSI test group includes most of the major 8-bit and 16-bit microprocessors, a variety of peripheral devices, and many commonly used ROMs, static RAMs, and dynamic RAMs.

The throughput of digital testing was a major concern. The actual execution time of a test does not affect throughput as much as the amount of time required to download a test to the test head. Because longer tests require longer download time, one of the library design goals was to concentrate on making the SSI and MSI tests as short as possible, consistent with adequate fault coverage. Because of the relatively lower number of LSI components per board, minimizing LSI test length was not considered so important to overall throughput.

A library test's primary purpose is to detect manufacturing faults. These faults include improperly loaded devices (either loading the incorrect device or loading the correct device incorrectly), bent pins, or damaged devices. These types of faults can be found by testing for I/O pin faults. Pin faults are classified as nodes either "stuck-at-one" or "stuck-at-zero" (usually referred to as SA1 and SA0). The library tests are therefore aimed at detecting SA1 and SA0 conditions on the I/O nodes. For many SSI and MSI devices, this amounts to a complete functional test. If required, additional functional testing may be added by the user.

Another important objective of the library was to be self-documenting. The source code for each device test is the only documentation provided with the library. Therefore, the tests had to be written in a readable, consistent format that would be easy to understand.

Design Tools and Verification

The people writing library tests had to have a good intuitive knowledge of digital devices and a solid understanding of the principles of digital testing. These people had to be able to translate timing diagrams and data sheets into thorough fault-detecting sets of test vectors.

In addition to writing device tests, a major task of the

```
        combinatorial

        assign  VCC       to pins  14
        assign  GND       to pins  7

        assign  Inputs_AB to pins  1,2

        assign  Output_Y  to pins  3

        family  TTL

        inputs    Inputs_AB
(a)     outputs   Output_Y

        ! ******************************************

        vector    Inputs_00
              set Inputs_AB to "00"
              set Output_Y to "1"
        end vector

        vector    Inputs_01
              set Inputs_AB to "01"
              set Output_Y to "1"
        end vector

        vector    Inputs_10
              set Inputs_AB to "10"
              set Output_Y to "1"
        end vector

        vector    Inputs_11
              set Inputs_AB to "11"
              set Output_Y to "0"
(b)     end vector


        ! ******************************************

        unit          "NAND gate test"
                      execute  Inputs_00
                      execute  Inputs_01
                      execute  Inputs_11
                      execute  Inputs_10
(c)     end unit
```

Fig. 1. *Library test example for a fictional single-element* NAND *gate (see Fig. 2). (a) Declarations section. (b) Vector definition section. (c) Execution section.*

library development team was to devise methods of verifying the tests. Each library test had to be verified to prove that it behaved the same as the real parts and to prove that the test would indeed fail a part exhibiting any one of the SA0 or SA1 faults. In addition, the verification had to be virtually automatic.

One of two methods used for test verification relied on TESTAID, the simulator software for HP's DTS-70 Digital Test System. This simulator uses its own library of digital devices to determine the effectiveness of test patterns. Its library contains a majority of the devices in the TTL, ECL, and CMOS groups stored in the HP 3065's library.

Automating the simulator verification required writing a software package that translates an HP 3065 VCL (vector control language) device test program (library test) into a test program for the simulator. The simulator analyzes the library test patterns using Boolean equation analysis and the simulator's gate-level model of the device. The software package then generates a report for the library test programmer, telling of any errors in the library test and listing faults that the library test did not detect.

Automating the simulator greatly enhanced the productivity of the library test designers. Because of the large number of parts to be included in the HP 3065's library, test writing began long before actual test-head hardware was available. The simulator was used to verify these tests. It also increased the quality of the library by helping the test designers learn better test techniques as they worked. The simulator is sensitive to timing problems and fault propagation problems that could result in failing good devices or making a test intermittent. The test designer was notified of such problems immediately after writing each test. This immediate feedback was a constant quality assurance check that greatly enhanced the test designer's abilities.

The second method of test verification used the actual HP 3065 test-head hardware. This method was used for all devices that could not be verified using the simulator verification method. This included devices not contained in the TESTAID library. In addition, a certain sample of the tests already verified by the simulator method were verified again with this method. This was done as a cross-check on the reliability of the verification results generated by the simulator.

Using a special test fixture and special test software, each vector control language test could be run on the prototype hardware, testing several real devices. The software checked to see that the test completed successfully. Then it automatically inserted all possible faults (one at a time),

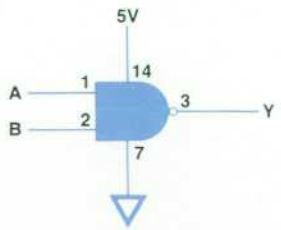


Fig. 2. *Pin layout for fictional single-element* NAND *gate used for library test example in Fig. 1.*

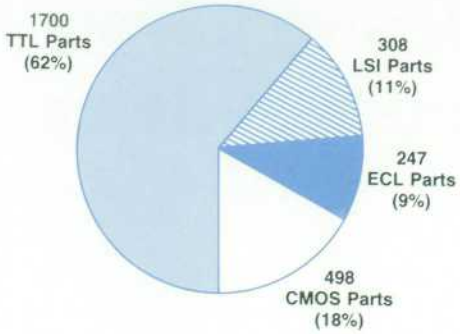


Fig. 3. *Distribution of digital parts covered by the test library of the HP 3065 Board Test System.*

using relays in the test head, and reran the test. A report was then generated to summarize the performance of the test. Any faults not detected by the test were reported. Like the simulator verification, this method provided immediate feedback to the test designer.

Handling Different Topologies

Developing a set of test patterns for most digital devices seems at first glance to be rather straightforward. However, testing in-circuit devices presents some special problems, chiefly because the devices can be connected in various topologies. The HP 3065's in-circuit test program generator (IPG-II) tries to make a library test fit the actual topology on a given printed circuit board. For IPG-II to be successful, the library tests must anticipate the most common topologies. It was not sufficient to write a test that could test a device in a topology where all of its leads are free. Therefore, before extensive writing of device tests began, the library development team defined test philosophies for each category of device type to be tested. These served as general guidelines to the engineers and technicians writing tests. These guidelines considered not only the importance of topologies, but also test throughput trade-offs and protection against potential device damage.

Description of Actual Test

Library tests are written in three distinct sections. (Refer to the example test in Fig. 1 for a fictional single-element NAND gate outlined in Fig. 2). The first section is the declarations section containing various descriptive information, such as family, inputs, and outputs. Notice that names chosen by the programmer are assigned to the various device leads, or pins. This simplifies programming and makes the test easier to read.

The second section defines the vectors that the test will need to use later. The programmer names the vectors descriptively to produce a more readable test.

The last section of the test is the execution section. This is where the actual execution sequence is contained, using the vectors previously defined. This section can use repeat loops, homing loops, and subroutines.

Summary

The initial objective was to have 2700 part numbers in the HP 3065's device test library. The actual number completed was 2753 (see Fig. 3). This represents 1700 TTL

devices, 247 ECL devices, 498 CMOS devices, and 308 LSI devices. Some of the microprocessors included are the 68000, 8086, Z8002, 8080, 8085, and 6800.

Since introduction, the device test library has been used with demonstration boards, alpha-site boards, competitive benchmark boards, and customer boards. Results indicate that the HP 3065's library tests cover 90% of the parts on these boards. In addition, the tests adapted correctly to the circuit topologies better than 90% of the time.

Acknowledgments

Davis Glasgow was responsible for designing and implementing the hardware verification method as well as writing tests. Two technicians, Tom Hindley and Ted Turner, did an excellent job writing device tests and assisting with the verification process.

Vance Harwood managed the library development group. His direction and technical suggestions were invaluable. I would also like to thank Mike Barker, my current manager in HP's Southern Sales Region, for his support.

An Interpreter-Based Board Test Programming Environment

by Mark A. Mathieu

SINCE AN AUTOMATIC in-circuit test program generator (IPG-II) in the HP 3065 builds most of the test programs for each circuit board, most of the board test programmer's time can be spent debugging and optimizing programs instead of developing them. To provide a strong debugging capability, an interpreter-based system is used. As an added advantage of an interpreter-based system, the use of system-dependent compilers and linkers can be avoided. Hence, the operating system of the underlying machine—RTE-6/VM on an HP 1000 E-Series Computer—can be more easily hidden behind a programming environment shell.

Board Test Basic

To add flexibility and power to the programming environment, a general-purpose BASIC-like language, Board Test BASIC (BT-BASIC), was defined. Although similar in most areas to the version of BASIC used by HP 9000 Series 200 Computers, some of the syntactic restrictions of that language are removed. For instance, optional line labels are used instead of line numbers and identifiers with lengths of up to thirty-two characters are permitted. A multiple-statement-per-line syntax is defined, providing a powerful tool for the immediate-execute capability of the interpreter. Some of the other features defined in Board Test BASIC are:

- Block-structured statements that include loop/exit if/end loop (a general-purpose loop construct), if/else/end if, and for/next.
- Subprograms—both subroutines and functions—with passed parameters. Variables in the main program are also accessible to subprograms by way of the global statement.
- on conditions that enable a program to handle breaks and run-time errors via call, goto, and recover statements.
- Intrinsic functions that include scientific calculation functions, bit and string manipulation functions, and string/numeric and numeric/string conversion functions.
- Statements and functions specific to the board test environment. These include the ipg statement which invokes IPG-II, the resistor statement which tests a resistor, and the autofile function which returns the identity of the test fixture.
- Statements that provide a data logging capability—which is necessary for understanding and controlling the board test process.
- Statements and functions that provide HP-IB (IEEE 488) and general-purpose I/O capabilities.
- Statements to support user-defined softkeys.
- Statements to provide a command editing capability. The change, move, and delete commands are examples.

By making editing statements and board-test-related statements a part of the general-purpose test language, learning to program in the board test environment is made easier.

Programming Environment Shell

The programming environment shell consists of a terminal screen display and a set of editing and execution features. The terminal screen (see Fig. 1) is formatted into four distinct areas: the status line, the command line, the workspace, and the softkey labels. The status line is used by the system to provide syntax error messages, run-time error messages, and other information. The command line is used to enter statements for immediate execution. The workspace contains the program lines being edited (both screen and command editing are supported). The softkey labels denote various useful functions for the current application.

Screen editing involves the use of the keyboard to edit the workspace. Keys such as **NEXT PAGE** and **ROLL DOWN** can be used to view the entire workspace. Keys such as

DELete **CHAR**acter and **CLEAR LINE** can be used to modify portions of the workspace. Lines can be entered or modified in the normal typing process.

Command editing involves the immediate execution of statements that perform a function on the workspace such as changing, listing, or finding. Several of the statements are based on the concept of a "marked" region (see Fig. 2). By using the mark softkey, a section of the workspace ranging from a single line to the entire workspace can be delineated. The section can then be edited in much the same way as the entire workspace and can be duplicated, moved, or deleted.

As a line is modified or added to the workspace, an incremental compiler validates the line syntactically. If the line contains a syntax error, an error message is displayed on the status line and the line must be corrected before editing can occur on any other line in the workspace. In this way, correct syntax for all lines in the workspace is ensured. Besides validating syntax, the incremental compiler aids typing by performing three types of conversions on source lines. Two conversions have the benefit of reducing the use of the **SHIFT** key on the keyboard. One conversion automatically converts apostrophes to quotes, allowing the programmer to type string constants with apostrophes. The second conversion replaces the \ character by the | character when the programmer uses the \ to separate statements. The third and most beneficial conversion allows the programmer to abbreviate keywords. This conversion automatically expands the keyword when the abbreviation is not ambiguous. In an environment where keywords such as capacitor, potentiometer, and incircuit are commonly used, the ability to abbreviate them as cap, pot, and inc comes in handy.

The immediate interactive execution features of interpreter-based systems are well known. Board Test BASIC's multiple-statement-per-line capability, a large command stack, and the ability to execute all but a handful of the statements from the command line add to the power of those features. The general lack of performance of interpreter-based systems is also well known. Some of the techniques used to improve test execution performance in the HP 3065 are described later.

Implementation Techniques

Compilation. When a line in a Board Test Basic program is compiled by the incremental compiler, a second form of the line, called intermediate code, is generated and kept with the source line as shown below for a typical program line.

Line: `ipg "/boards/" & val$(Board), Error| beep`

Intermediate code:

String constant	:	/boards/
Numeric r-value identifier	:	Board
Operand counter	:	1
Opcode	:	val$
Opcode	:	&
Numeric l-value identifier	:	Error
Operand counter	:	2
Opcode	:	ipg
Operand counter	:	0
Opcode	:	beep

The intermediate code is the concatenation of the postfix representations of the statements in the line. The postfix

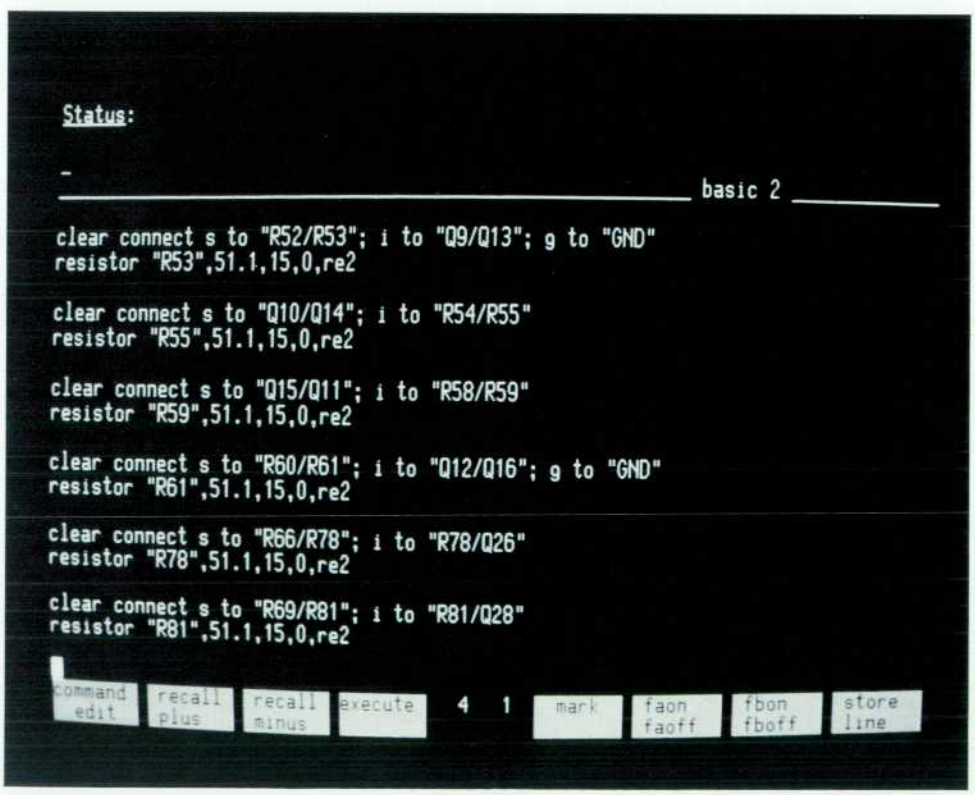


Fig. 1. *Typical HP 3065 programming terminal display.*

representation (reverse Polish notation) for a statement consists of cells corresponding to its operands, an operand counter cell (not generated for the assignment statement), and an opcode cell. Expressions themselves are postfix constructs. However, operand counter cells are not generated for opcodes that denote operators in Board Test BASIC. Special attention is given to operand cells corresponding to identifiers in Board Test BASIC. Information on their use is added to these cells for use during execution. To avoid having to recompile each line as a program is brought into the workspace, get/save/re-save and load/store/re-store operations copy the intermediate code into and out of files along with the source lines.

Execution. Conceptually, the interpreter resembles a stack architecture computer. Parts of the interpreter correspond to a program memory, a program counter, an operand stack, and a processing unit. In simple terms, execution amounts to the following:

1. The cell in memory pointed to by the program counter is fetched.
2. The program counter is incremented.
3. The cell is either placed on the operand stack (if it is an operand) or executed (if it is an opcode).
4. Step 1 is returned to.

The execution of an opcode generally results in operands being removed from the operand stack.

The intermediate code structure generated by the compiler is specialized for editing. For execution, the intermediate code is copied as needed into the memory of the interpreter. The memory structure is more compact than the intermediate code structure and also has better locality (more easily restricted to a portion of system memory). The size and locality improvements combine to help the HP 3065's virtual memory system adapt the working set to the executing program.

Copying the intermediate code into the interpreter's memory does not in itself result in a running program. Many of the functions that cannot be performed by the incremental compiler must be performed on the first execution pass of the program. These include, for instance, building symbol tables, locating destinations of goto statements and subprogram invocations, and matching up block-structured statements. Since these functions are time consuming and are only performed once, a technique called backpatching is used to speed up subsequent execution passes. Essentially, backpatching is the technique of modifying the program in memory during execution to include newly learned information. As a simple example, assume that the statement print A is to be executed for the first time. The operand cell for the variable A specifies that it is a compiler-generated cell. Realizing this, the interpreter automatically—before stacking the operand—tries to locate (inserting if necessary) the variable in the current symbol table. If successful, the interpreter replaces the compiler-generated cell with a corresponding interpreter-generated cell. Later executions of the statement are much quicker because no symbol table processing has to be done for the interpreter-generated cell.

Some of the first-execution tasks such as locating destinations of goto statements and subprogram invocations and matching up block-structured statements only involve backpatching as the final step in the process of their being made ready for execution. Many interpreters have a prerun process that performs these tasks by making a pass through the program before execution of the program begins. Since board test programs are often over one thousand lines long and usually have long segments of code devoid of any block-structured statements, it was decided that the functions of a prerun process would be performed when needed during the execution of the program by a process called LOOK_AHEAD.

LOOK_AHEAD is triggered by three events: a reference to an undiscovered label, a reference to an undiscovered subprogram, and the execution of an opcode of a loop, if, or for statement. As an example of LOOK_AHEAD in action, refer to the program segment below.

```
    goto L2
    .
    .
L1: loop
        exit if A
        .
        .
        if B then
        .
        .
        else
        .
        .
        end if
    end loop
    .
    .
L2: print A, B
```

Since the label L2 has not yet been discovered in the first line, LOOK_AHEAD is invoked. In searching for L2, LOOK_AHEAD encounters the label L1 and inserts it into the current symbol table. Continuing on, LOOK_AHEAD encounters the loop statement and places it (and its location) on a stack. This stack is used to validate nesting of block-structured

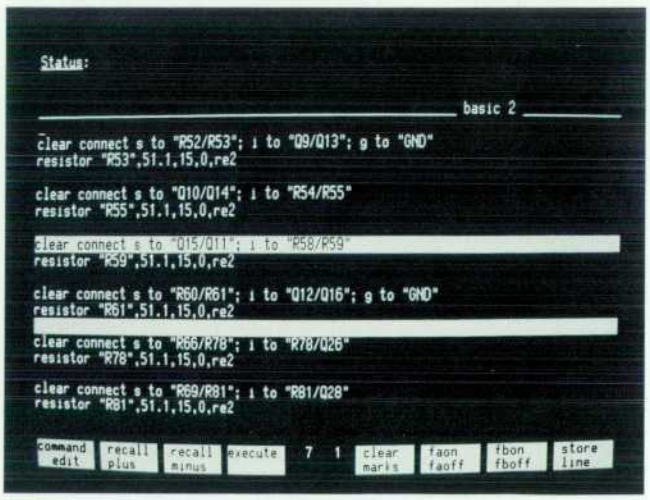


Fig. 2 *Terminal display of Fig. 1 showing a marked region (inverse video in midscreen).*

statements and provide information for backpatching them into execution-ready code. The exit if, if, and else statements are added to the stack shortly thereafter. Upon encountering the end if statement, LOOK_AHEAD validates the if/else/end if structure, backpatches it into executable form, and removes the three items from the stack. The discovery of the end loop statement causes similar processing for the loop/exit if/end loop structure. Then LOOK_AHEAD finds L2, inserts it into the symbol table, and returns control to the interpreter.

To take advantage of the speed improvements caused by backpatching, the executable form of the program is reused on successive executions. Since a board test program may be executed many times in succession (i.e., for an entire production run of boards), the speed improvements made during the first execution are repeatedly used. Empirical results on board test programs show a six-fold speed increase between the first execution and successive executions because of backpatching.

Acknowledgments

There were many members of the project team that designed and developed the software for Board Test BASIC and the test programming environment. Doug Manley designed and developed both the editor and the heart of the interpreter. Martha Conant was responsible for the HP-IB and general-purpose I/O statements and other miscellaneous statements and functions. Dave McGownd (Gowie) developed nearly all of the intrinsic functions and hierarchical file system interface statements and also implemented the break processing. Ken Posse was responsible for most of the statements and functions that interface with the test system hardware. Mike Hendricks developed the software for the Shorts test (with assistance in the form of a driver from Doug Baskins) and HP CHEKPOINT in the HP Q-STAR package (see box on page 6). Tom Fay was responsible for the digital testing software. Ron Mathieson developed the software to provide a network capability. More recently, Mike Wahl improved analog testing throughput, Carroll Morris upgraded the digital debugging facilities, and Laurie Shammel added an RS-232-C/V.24 interface capability.

Many others also deserve acknowledgment for their assistance. Doug Baskins and Rod Browen, our RTE operating system gurus, provided elegant solutions whenever we had performance problems to solve or operating system limitations to circumvent. Bob Balliew was always able to give an answer or an idea that showed a wisdom beyond his years. The inputs and suggestions of marketing people like Tom Newsom were always useful and needed even when we did not really want to hear them.

I would like to acknowledge one final person for his contribution. Bill Groves provided invaluable leadership, guidance, and knowledge during the design and development of the HP 3065.

Testing for Short-Circuit Failures

by T. Michael Hendricks

THE PURPOSE OF THE HP 3065 Board Test System's Shorts test is to locate short circuits on printed circuit boards caused by production problems like solder splashes and bent pins. For a variety of reasons, boards may also contain short circuits that are known to be there. These are not caused by production defects, but are inherent in the design of the board. Inductors, small resistors, and printed circuit board traces are common sources of known short circuits.

Because of the possibility of known short circuits, the HP 3065's Shorts test is done in two stages. The first, or learning, stage is intended to locate the known short circuits. During the learning stage, the Shorts test is run on a known good board. Any short circuits found during the learning stage are assumed to be inherent in the design of the board and not caused by production problems. A list of these known short circuits is stored internally in the HP 3065 file system for use during the second, or testing, stage. The two purposes of this stage are to locate any additional short circuits (which presumably are caused by production defects) and (optionally) to confirm that all of the known short circuits are actually present.

Shorts Algorithm

Speed was the primary consideration in the design of the HP 3065's Shorts test. Because a maximally connected network of N nodes contains on the order of N^2 branches, the problem of finding a short circuit in any branch is inherently an N^2 problem. The maximum number of nodes the HP 3065 is capable of handling is roughly 1400, which corresponds to approximately two million branches to be checked. An algorithm requiring this many individual operations would be unacceptably slow, so a two-phase algorithm was chosen instead.

Both the learning and the testing stages use this two-phase algorithm. The first phase is called the detection phase. It is a sequential scan of all nodes on the board to determine if any short circuits are present. The detection phase is a quick way of finding out if a board is free of short-circuit failures and if so, passing it. It typically tests many branches in parallel, which gives it good speed, but this can introduce ambiguity in the location of any short circuits it detects. Therefore, if short circuits are found, the second phase of the algorithm must be called. This isolation phase is used to pinpoint exactly which

branch(es) are responsible for the detected short circuits.

The isolation phase consumes substantially more system resources than the detection phase. Efficiency therefore demands that it be invoked as little as possible. This is the reason for the separation of the Shorts test into a learning stage and a testing stage. (Speed is less important during the learning stage than during the testing stage, since it is a one-time task.) During testing, the known short circuits are excluded from consideration so that they need not be redetected and reisolated. Unless the board has some unknown short circuits (failures), the isolation phase is never invoked during testing.

The stored list of known short circuits may be confirmed during testing by an optional part of the algorithm called the Opens test. The Opens test attempts to determine if any of the known short circuits are missing.

Phantom Shorts

A phenomenon known as phantom short circuits can have a profound impact on the speed of the Shorts test. Because the detection phase tests many branches in parallel, it is possible for it to find short circuits that are not really present. For example, if the resistance threshold is 50Ω, and the board contains a couple of 90Ω branches, it is possible that at one point in the detection phase, the two 90Ω branches will be switched in parallel between the source and the detector, giving the appearance of a single 45Ω branch. Because 45Ω is less than the resistance threshold, the detection phase concludes that a short circuit is present and it invokes the isolation phase. The isolation phase ultimately tests the branches individually and concludes that no short circuit is present. Unfortunately, it is a time-consuming algorithm to run. This situation is referred to as a phantom short circuit because the detected short circuit disappears upon closer inspection. Unlike known short circuits, which are isolated once (during learning) and can be sidestepped thereafter, phantom short circuits must be reisolated on every board tested. It isn't possible to omit them from consideration without introducing the possibility of overlooking real short circuits during testing.

Isolation Algorithm

It has already been stated that the isolation phase is the most time-consuming part of the Shorts algorithm. The HP 3065 reduces the impact of isolation on test throughput by using a binary chop technique. Theoretically, this is the optimal algorithm for isolating a single short circuit. When the detection phase finds an apparent short circuit, it has a detector connected to a single pin (the FROM pin) and a source connected to one or more pins (the TO pins). The purpose of isolation is to determine which, if any, of the circuit board nodes connected to the TO pins are actually short-circuited to the node connected to the FROM pin. It does this by using a recursive algorithm that successively divides the list of TO pins in half.

For example (see Fig. 1), assume that isolation is invoked with pin 0 as the FROM pin and pins 1 through 4 as the TO pins. The isolation algorithm begins by considering all of the TO pins to be active pins. It divides the set of active pins in half. It leaves the relays corresponding to the first half closed, but opens the relays in the second half after noting that they need to be tested later by pushing them to a run-time stack. It next samples the detector to see if there is a short or open circuit. If there is an open circuit, neither active pin in the first half (pin 1 or pin 2) can be shorted to the FROM pin, so they are exonerated. Neither need be tested again. However, if the result is a short circuit, it is possible that either active pin or both are shorted to the FROM pin. In this case, the list of active pins is once again bisected (this time the resulting halves consist of a single pin each) with the first half remaining closed and the second half opened. This bisection is followed by another measurement. If the result is an open circuit, the (single) active pin is exonerated. If the result is a short circuit, the (single) active pin remains suspect. Because the list of active pins has been reduced to a single entry, it is possible to state with certainty that the current active pin is shorted to the FROM pin. Note that it is not correct to terminate isolation at this point. It is possible that several of the original set of TO pins are shorted to the FROM pin, so it is necessary to continue testing until each entry in the original set has been tested. Whenever a short circuit is found or a set of active pins is exonerated, the isolation algorithm returns to the run-time stack and resumes testing with a set of pins that was deferred in some previous bisection. Only after the run-time stack is exhausted can the isolation algorithm terminate.

This binary-chop isolation algorithm has two advantages. It finds a single short circuit involving X TO pins in a

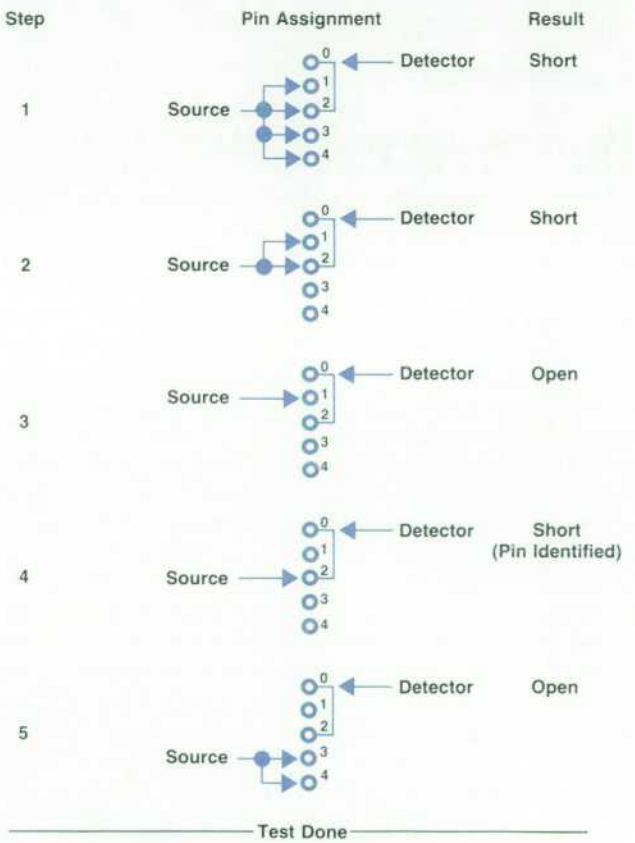


Fig. 1. *Example of use of binary-chop algorithm for isolating a short circuit (between pins 0 and 2 in this example). Here pin 0 is designated as the* FROM *pin.*

maximum of $2 \times \log_2(X)$ measurements. This offers a substantial speed improvement over the linear search that is used in some test systems. It also reduces the impact of phantom short circuits on execution speed. By successively bisecting the set of active pins, it quickly reduces the number of parallel paths to the FROM pin, meaning that phantom short circuits usually disappear after only one or two levels of recursion. This contrasts favorably with a linear search that would require X measurements to recognize a phantom.

Node Ordering

The details of the detection phase are such that the number of parallel paths decreases monotonically from the first iteration to the last. This fact can be used to reduce the incidence of phantom short circuits. As mentioned earlier, phantom short circuits can arise when several different branches are tested in parallel and their combined resistance is less than the threshold, even though all of their individual resistances are greater than the threshold. This means that branches with small resistances are less likely to cause phantoms if they are tested late in the detection phase (because there are fewer in parallel) than if they are tested earlier. To take advantage of this fact, the HP 3065's in-circuit program generator (IPG-II) computes for every node the parallel combination of all resistances connected to that node. It then orders the nodes in the Shorts source pin assignment in decreasing order of this effective resistance. This relieves the user of the burden of experimenting to find the node sequence that yields the fastest test.

Transient Handling

Capacitors and inductors pose special problems for the Shorts test by introducing transients into the response of the detector. The operation of the Shorts test can be modeled as the application of a step voltage to the TO pin(s) and the measurement of the response at the FROM pin at some later time. If a capacitor connects the nodes assigned to the FROM and TO pins, the response to the application of the step is exponential. At the moment of application of the step, the capacitor acts like a short circuit, but after it charges up, it acts like an open circuit. Hence, if the measurement is taken too early, the wrong result is obtained. Inductors behave in a dual but reverse way, first appearing (incorrectly) as an open circuit and later appearing as a short circuit.

The Shorts test deals with these dynamic conditions by repeatedly sampling the detector's response. The sampling scheme is defined by three parameters: T_a, T_b, and T_p. T_a is the amount of time to wait between issuing the commands to establish the relay state in the scanner and taking the first measurement. This time must be sufficient for the relays to actuate and for the detector output to stabilize. After T_a expires, the detector output is sampled periodically with a period T_p. This periodic sampling continues for a maximum time specified by T_b. (The user specifies the value of T_b in the SETTLING DELAY statement.) The T_b interval begins at the end of the T_a interval.

The details of the timing scheme are different between the learning and testing stages. During the learning stage, operation is as follows:

- Take the first measurement at time T_a and start timer for T_b.
- Starting at T_a, take a measurement every T_p seconds until T_b has expired.
- If at any time during T_b the measurement changes state, stop sampling and consider the new state to be the correct one.
- Otherwise, consider the (constant) measurement to be the correct one.

This approach maximizes the probability that the board will be learned correctly. Both capacitors and inductors are given time to reach steady-state values (if T_b is sufficiently long). Resistors have no similar dynamic behavior so the measurement stabilizes instantly for them. The rationale for the early exit when a state change is observed is that a single-pole model for the measurement circuit predicts that the signal being sampled can be no more complex than a rising or falling exponential. Thus, it can cross a threshold at most once. Under this model, once a change is observed there can be no further change. Thus, it is a waste of time to continue sampling. Surprisingly, this means that branches containing inductors and capacitors are actually learned faster than resistor branches.

During the testing stage, there is an expected result for each measurement. During the Shorts test, open circuits are expected, and during the Opens test, short circuits are expected. This testing scheme operates as follows:

- Take the first measurement at time T_a and start timer for T_b.
- If the first measurement conforms to expectations, stop sampling and return the expected value.
- Starting at T_a, take new measurements every T_p seconds until T_b has expired.
- If the expected measurement is obtained, stop sampling and return the expected value.
- If T_b expires before the expected measurement has been obtained, return the opposite of the expected value.

This algorithm has been designed to operate at maximum speed. It never waits longer than necessary for the circuit to stabilize. Since resistors theoretically stabilize instantly, the initial sample conforms to expectations and no further sampling is necessary. For inductors or capacitors, the first answer is likely to be incorrect, but the algorithm waits only until the transient has decayed.

Acknowledgments

I would like to thank John McDermid, program manager for the HP 3065 and my project manager at the start of the Shorts test development; his understanding of board test problems in general and the capabilities of the HP 3065 in particular was invaluable. Thanks also to Bob Balliew for getting me started in understanding the Shorts test problem. Substantial contributions to the software effort were made by Doug Baskins, who implemented many of these ideas in assembly language, Mark Mathieu, who wrote the Shorts test compiler for the HP 3065, and Bill Groves, who supervised the whole effort.

Reducing Errors in Automated Analog In-Circuit Test Program Generation

by John E. McDermid

AUTOMATED ANALOG IN-CIRCUIT test program generation evolved from the need to reduce test program development time. Early program generators functioned primarily as typing aids, relieving the programmer of the tedium of coding but still requiring a large amount of manual debugging. Other important program characteristics, such as test throughput and ongoing program maintenance, were determined by the competence of the programmer. As the programmer began to debug a test program, changes were made through an understanding of how a board test system made in-circuit measurements. This understanding, in effect, constituted a heuristic error analysis of the circuit. The skill of the programmer and the heuristic methods used significantly affected the throughput of a board test system and the amount of program maintenance required.

Today, most modern circuit board testers use similar fundamental measurement principles to isolate the components under test electrically. This article discusses seven major error sources in automated analog test program generation and the application of the results of this type of error analysis for improving board test throughput and simplifying program maintenance. These results guided the design of the analog test portion of the in-circuit program generator (IPG-II) used in the HP 3065 Board Test System.

Sources of Error

The most commonly used analog in-circuit measurement technique is shown in Fig. 1. Z_x represents the component under test. The objectives of this circuit are to force the current I_b to be zero (guard the circuit)[1] and to ensure that all of the source voltage V_s is applied to Z_x. Under these conditions, the voltage $V_o = -R_f/Z_x$ is independent of impedances Z_a and Z_b, making the measurement independent of the surrounding circuitry. Note that a similar statement can be made about the measurement circuit of Fig. 2 when the unknown component is in the feedback circuit of the operational amplifier A.

The models in Fig. 1 and Fig. 2 are highly idealized and only approximately represent the realities of actual test equipment. Either the programmer or the program generator must understand and successfully compensate for the nonideal conditions of source impedance, input impedance of the operational amplifier, guard lead impedance, system residual impedances and their variations, settling delays in the amplifier circuit, basic measurement accuracies of the system, and intrinsic system noise. This compensation must be done within the constraints of the test philosophy (do the measurements require traceable accuracy?) and the desired accuracy goals (are 5% pull-up resistors measured to 5% or 20%?). An understanding of these nonideal characteristics is crucial to developing an automatic in-circuit test program generator that can create test programs with high throughput and low maintenance.

Source Impedance. A nonzero source impedance lowers the actual voltage available from the source. The currents I_a and I_x (see Fig. 3) both flow through the source resistance R_s. This reduces the voltage across Z_x by $R_s(I_x + I_a)$. This error must be compensated by either remotely sensing the voltage or increasing the positive tolerance for resistors and inductors and decreasing the overall tolerance for capacitors.

Input Impedance. Under ideal conditions, the voltage at the inverting terminal of the operational amplifier is zero, causing the input impedance at this terminal to be zero (a virtual ground). The actual impedance in practical circuits is not exactly zero, but is determined by $Z_{in} = R_f/(A(s) + 1)$ where A(s) is the open loop transfer function of the amplifier (see Fig. 4).

If the amplifier has only a single pole, then the equivalent input impedance is an inductor of value $L_{in} = R_f/(2\pi f_c)$ in series with a resistor of value $R_{in} = R_f/(A + 1)$. A is the

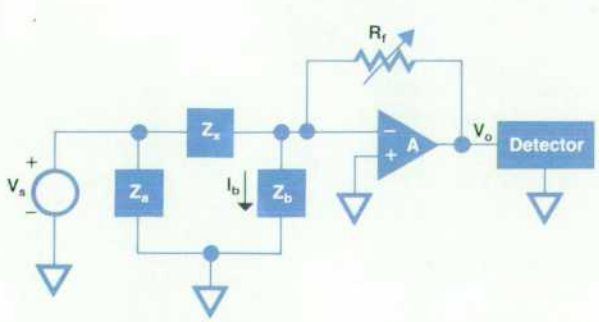


Fig. 1. *Standard analog in-circuit measurement configuration for unknown component Z_x.*

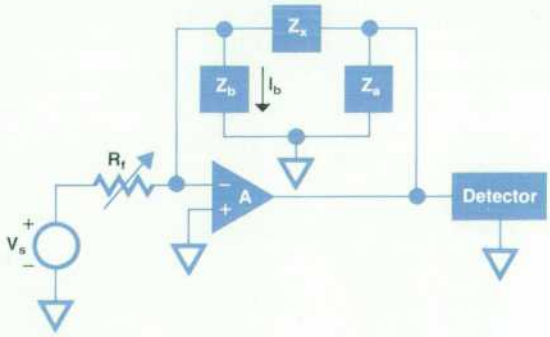


Fig. 2. *Measurement configuration with unknown component Z_x in feedback loop of operational amplifier.*

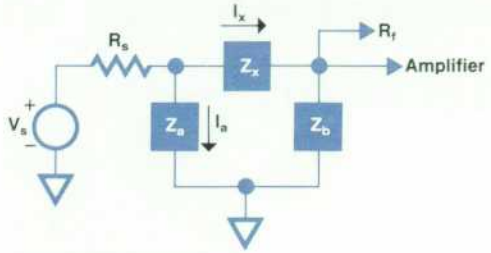


Fig. 3. *One source of measurement error is introduced by source impedance R_s.*

open-loop dc gain of the amplifier and f_c is the unity-gain crossover frequency. The effect of this approximation in the circuit is shown in Fig. 5.

The current I_x now splits, some of it flowing through Z_b and Z_{in} and the rest of it flowing through the feedback resistor. Because this current is smaller than it is in the ideal case, either it must be remotely sensed or the positive tolerances for resistors and inductors and the negative tolerance for capacitors must be increased. Note that when lead impedances are important, these impedances should be added in series with the input impedance.

Guard Lead Impedance. Guard lead impedance is the most significant error contributor of the three lead impedances. In Fig. 6a, the impedances Z_a, Z_b, and R_g form a wye network. This network can be converted to a delta network as shown in Fig. 6b with values Z_a', Z_b', and Z_x'. If R_g is small in comparison with Z_a and Z_b, the values of Z_a' and Z_b' are approximately the same as Z_a and Z_b. However, the value of Z_x' is Z_aZ_b/R_g.

Even small values of R_g can be very significant when guarding situations are difficult. For example, if Z_x is 10 kΩ, Z_a and Z_b are each 10Ω, and R_g is 0.1Ω, then $Z_x' = 1000\Omega$. This makes the measured value lower than the actual value by more than one decade of resistance.

There are two important points to be observed. First, for resistive impedances, this error is in the opposite direction from those of source and input impedance, making error cancellation possible. Second, if Z_a and Z_b are reactive, then Z_x' may not be the same element type as Z_x. The difficulty of analysis may cause a test programmer to abandon an analytical choice of limits. Because of the possibility of different component types and cancelling errors, the programmer may find a very repeatable measurement on a single board, but the board-to-board stability of the test will be poor. The test program will fail good components until the tolerances are adjusted by using some form of statistical inference on a large number of boards. If this process is not completed before production board testing begins, high maintenance costs will result.

Residual Impedances. System residual impedances are impedances that are measured with an open circuit on the test system. These impedances will make large resistances appear smaller and small capacitors appear larger. Although the residuals for most test systems are small, their variation as a percent of nominal value may be large.

Care must be taken that the variation in residuals is compensated either by the test program generator or during the debugging technique. Consider the measurement shown in Fig. 7a. Stray capacitance inside the test system (C_{stray}) can be as much as 900 pF and a residual impedance of approximately 1Ω is not uncommon for the impedance R_g of the guard lead. Fig. 7b models the error contributed by C_{stray} and R_g. The current in the feedback resistor R_f is of the same magnitude as the measurement signal, since the impedance from the negative terminal to ground is one tenth that of Z_x and the voltage at the positive terminal is $0.1V_s$. Not only is the measured value of capacitance high by 1000%, but it is highly sensitive to the value of R_g. A 20% variation in R_g will result in a 20% variation in Z_x. If this test is debugged on a system using a single board, this problem may not be uncovered. It is important that problems such as this either be handled by the program generator or be completely debugged using a technique designed to uncover this type of problem. Failure to recognize this will result in the unnecessary replacement of good parts and an increase in the associated manufacturing and testing expenses.

Settling Delays. Settling delays occur whenever the poles of the operational amplifier and the poles of the network combine to place the closed-loop poles of the system near the imaginary axis. These delays are very strongly affected by the amplifier compensation originally chosen by the amplifier designers. If the operational amplifier has a single pole as shown in Fig. 8a, and the network also has a single pole as shown in Fig. 8b, then the root locus of the closed-loop poles will look like Fig. 8c.

For normal values of amplifier gain, the closed-loop poles will be complex and the settling delay will be inversely proportional to the distance from the pole to the real axis. If the measurement is being made in the feedback path, as shown in Fig. 2, an additional pole (caused by the amplifier's output impedance and Z_a) is introduced and

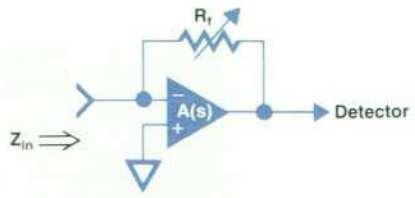


Fig. 4. *Another source of measurement error is the input impedance Z_{in} of the operational amplifier.*

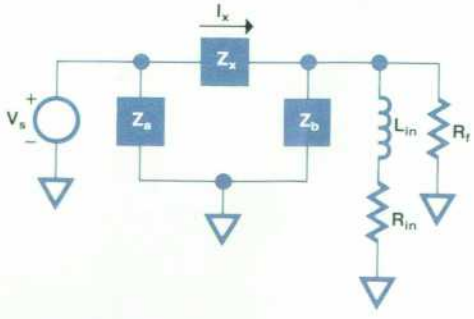


Fig. 5. *If the amplifier in Fig. 4 has only a single pole, then Z_{in} can be represented by an inductor in series with a resistor as shown.*

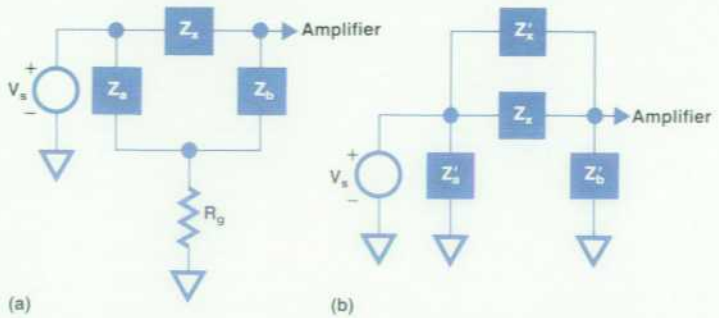


Fig. 6. *The wye network (a) containing the guard lead impedance R_g can be converted to a delta network (b) to determine its effect on the measurement of Z_x.*

causes the locus to bend away from the first pole (Fig. 8d). This causes the settling delay to become even longer, or if the locus moves sufficiently, instability occurs.

The HP 3065 Board Test System and its predecessor, the HP 3060,[2] compensate for this problem by introducing a series of pole-zero pairs along the real axis. The root locus (when a single network pole exists) for these systems is shown in Fig. 9. Under these circumstances the poles follow a locus of approximately constant damping factor. Settling delays are then much shorter than those of a single-pole system.

Certain analog-to-digital detector characteristics can also be used to reduce the effects of long settling delays. If an integrating detector is used and the damped frequency of oscillation of the amplifier output (Fig. 10a) is much higher than the corner frequency of the detector response (Fig. 10b), a significant portion of the settling delay time can be reduced. If the damped oscillation frequency can be predicted from a knowledge of the test system poles and the component under test, then an appropriate combination of settling delays and detectors can be selected by the test program generator.

Measurement Accuracies. All instrumentation has basic accuracy limitations. The degree to which these limitations affect system performance depends on the size of the signal being measured, the source accuracy, the accuracy and dynamic range of the detectors, operational amplifier parameters, and the desired accuracy of the measurement. Often these parameters are imbedded in the test system where their characteristics are invisible to the programmer. If a particular system is performing much better than the general system specification, the programmer may inadvertently develop a test that is too stringent for another system or for even the current system at some future time and temperature.

It is important to realize that these problems can occur in places other than those requiring high-accuracy measurements. Consider the example of Fig. 11. If the operational amplifier has an offset voltage of 10 μV, then the error contribution to the output voltage is 10 mV or 1%. However, if the actual offset voltage can only be controlled to within 500 μV, the error can potentially be as much as 50%. A programmer would have no tool for detecting this problem without running tests on different test systems or at multiple temperatures.

Residual System Noise. Two types of system noise must be considered. The first type, random noise, is generated by broadband noise sources in the input devices. This noise spectrum is modified by the transfer function of the operational amplifier circuit and the response characteristics of the detectors. An example is shown in Fig. 12. The noise voltage in the input devices of the amplifier is represented by the source E_n. In this particular example the noise generated by E_n is approximately the same as if the input device noise had been amplified with an amplifier of gain A_2 and corner frequency f_2. This noise is obviously much higher than would have been generated by a resistor of the same impedance.

The effect of the noise must be evaluated through the detector characteristics. Although these characteristics will vary with the type of system being used, some judgment must be used in selecting the detector. The corner frequency of the detector should be low enough to limit the noise to an acceptable level while not severely compromising system throughput.

The second type of system noise results from discrete

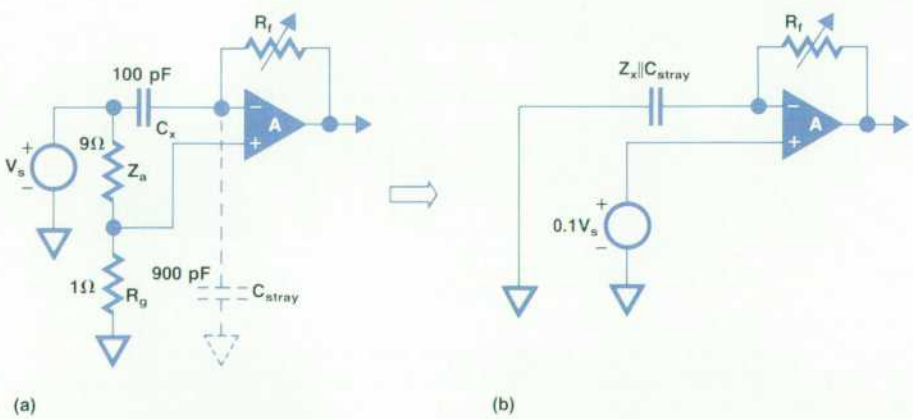


Fig. 7. *The stray capacitance in the test system combined with the guard lead impedance as shown in (a) can be modeled by the circuit shown in (b) to calculate its effect on the measurement of Z_x.*

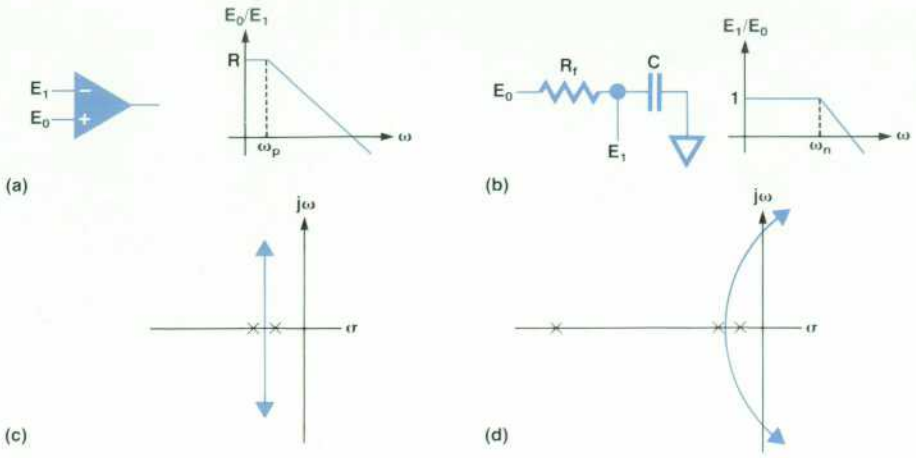


Fig. 8. *A single-pole amplifier (a) and a single-pole input network (b) results in a closed-loop root-locus plot as shown in (c). By moving the network to the feedback path of the amplifier, another pole is introduced as shown in (d), causing longer settling delays or, in some cases, instability.*

spectral components. Often these are related to power line frequencies or the conversion frequencies of dc-to-dc converters used within the system. These frequencies can also be evaluated through the transfer function of the measurement circuit and the total contribution projected. High-Q responses of the circuit that fall near 60-Hz harmonics can add large amounts of noise to the test signals. For these situations it is important to evaluate the sensitivity of the circuit response to discrete system noise components.

IPG-II Results

Using the results of the above error analysis, an improved automatic test program generator (IPG-II) was designed and evaluated by testing a printed circuit board selected from a group of boards populated with difficult-to-test component configurations. The test board was selected by an experienced board test programmer and was in his words "One of the most difficult I have programmed." This board had the following breakdown of automatically testable analog parts:

Resistors	218
Capacitors	69
Inductors	4
Transistors	26
Diodes	29

The original test program generator used by the programmer (not IPG-II) was rerun and the yields of correctly programmed tests determined. These yields were:

Resistors	124	(57%)
Capacitors	30	(43%)
Inductors	0	(0%)
Transistors	13	(50%)
Diodes	0	(0%)

The program was then debugged again by an engineer whose objective was to understand the cause of each incor-

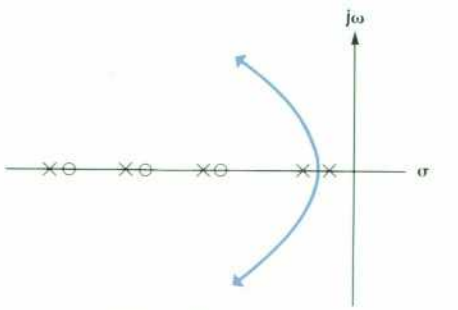


Fig. 9. *The HP 3065 Board Test System compensates for part of the settling delay problem by introducing pole-zero pairs to the plot shown in Fig. 8c. This forms a locus of an approximately constant damping factor.*

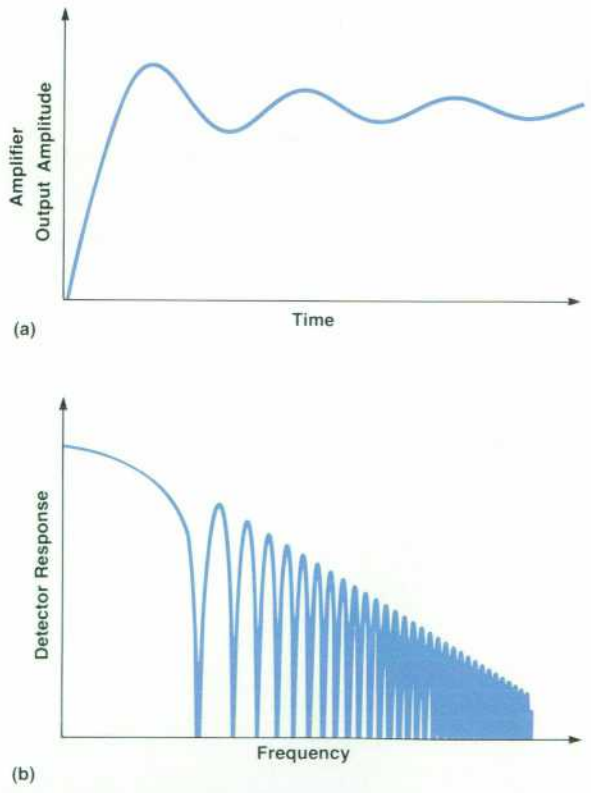


Fig. 10. *Integrating detector measurement system characteristics. (a) Damped oscillation output of amplifier. (b) Response of integrating analog-to-digital detector versus frequency.*

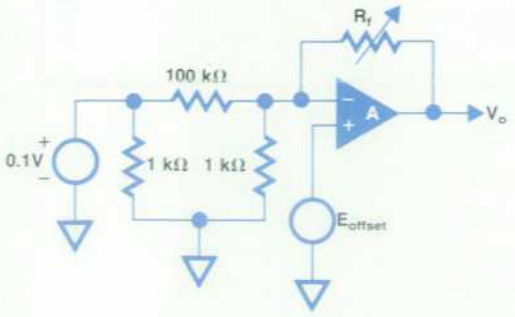


Fig. 11. *The error contribution by the offset voltage of the measurement circuit's operational amplifier can be large.*

rect test. These activities demonstrated that 43 devices were untestable because of circuit topology and measurement limitations of the system. Other tests were modified so that time-consuming measurement options were used only where needed. As a result of this manual debugging activity, the original board test time was reduced from 93 seconds to 54 seconds. The same circuit data base used to create this program was then used to evaluate the new program generator design (IPG-II).

When IPG-II was used to generate a board test for the printed circuit board described above, 321 devices passed initially. In addition, it correctly identified all of the 43 untestable parts as untestable. Each of the tests was measured for accuracy and repeatability using five good boards. Those tests that failed to operate correctly or were marginal were reviewed and the cause identified. The detailed breakdown of the causes of the 25 incorrect tests is listed below:

- Resistors (11 out of 218):
  - 5 required a change in the sense point for the guard bus—a known algorithm defect.
  - 4 had marginal tolerances when tested on multiple boards.
  - 1 required a change in the location of a remote sense point.
  - 1 failed because of an inappropriate potentiometer model.
- Capacitors (6 out of 69):
  - 4 devices required a change in the sense point.
  - 2 devices failed because of significant stray capacitances not entered into the data base.
- Inductors (2 out of 4): Both parts failed because the actual part values were too small to be accepted by the board entry program. Both parts passed when the correct nominal value was entered.
- Transistors (6 out of 26): All failed because of a known algorithm defect.
- Diodes (0 out of 29): No failures.

When the causes of the failing parts had been corrected, the board test time was measured. The time was 43 seconds, representing an improvement of 20% over the optimized result for the earlier program generator. It should be pointed out that these results were not achieved at the expense of long program generation times. The total test programming time required for this board using IPG-II was 31.5 minutes.

IPG-II Features

The new IPG-II program generator is capable of modeling the effect of:

- Source impedance (improved model)
- Input impedance (improved model)
- Guard lead impedance (improved model)
- System residual impedances.

It is also capable of executing a complete sensitivity analysis to establish the variation caused by:

- A 20% change in Z_a and Z_b
- Fixture contact resistance
- Fixture wiring resistance and inductance
- Fixture residual capacitance
- System resistance and inductance
- System residual capacitance.

Settling delays are computed from the positions of the poles and detectors are selected by analyzing the effect of integration on the damped oscillations of the test network response.

A table of the uncertainties caused by accuracy limitations of test system components is included in IPG-II. These inaccuracies are added to the tolerance bounds generated by modeling the component. IPG-II also calculates uncertainties caused by random noise and discrete spectral noise and adds them to the tolerance limits (spectral noise is evaluated out through the 160th harmonic of the ac power line frequency).

Summary

An automatic in-circuit test program generator was modified to compensate for seven major sources of error and evaluated on a difficult analog board. The original turn-on rate (number of successful tests/number of devices to be tested) for the board, without considering all seven major error sources, was 48%. The turn-on rate using the modified program generator (IPG-II, now used by the HP 3065 Board Test System) was 93%. Of the 25 failing tests, 44% of the

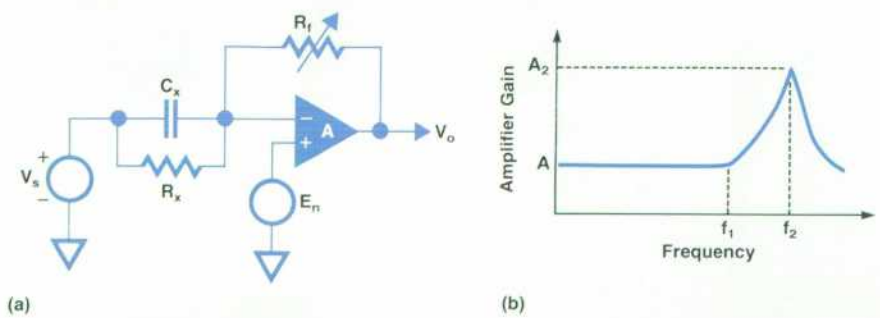


Fig. 12. *System noise can be modeled in the measurement circuit by introducing a noise source E_n as shown in (a). Its effect on the test response is shown in (b), adding a peak at frequency f_2.*

failures were caused by known (and easily correctable) algorithm faults, 20% were caused by incorrect information given to the program generator, and 36% (6 parts) required further investigation. As an added benefit, the board test time was 20% faster than that of a program debugged by an expert test programmer. The cost of this improvement was very low, requiring only 11.5 minutes of additional computer time to generate the revised program.

Acknowledgments

The author would like to acknowledge the contributions of Rod Browen and Dave Crook that made this work possible.

References

1. D.T. Crook, "Analog In-Circuit Component Measurements: Problems and Solutions," *Hewlett-Packard Journal*, Vol. 30, no. 3, March 1979.
2. D.T. Crook, et al, "Hardware Design of an Automatic Circuit Board Tester," ibid.

CORRECTION

In the July issue, the complex result for the complicated expression in the right column of page 26 is incorrect. The correct result given by the HP-71B Computer with a Math Pac is (3.00271078983, 612622756366).

Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, California 94304

HEWLETT-PACKARD JOURNAL

OCTOBER 1984 Volume 35 • Number 10

Technical Information from the Laboratories of Hewlett-Packard Company

Hewlett-Packard Company, 3000 Hanover Street
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
Van Heuven Goedhartlaan 121
1181 KK Amstelveen, The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku Tokyo 168 Japan
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive, Mississauga, Ontario L4V 1M8 Canada

Bulk Rate
U.S. Postage
Paid
Hewlett-Packard
Company

CHANGE OF ADDRESS: To change your address or delete your name from our mailing list please send us your old address label. Send changes to Hewlett-Packard Journal, 3000 Hanover Street, Palo Alto, California 94304 U.S.A. Allow 60 days.

5953-8528