Fast and High Quality In-Circuit Test Development Through Expert Debug

Alex Loopik, Dave Cook*, Rod Browen*, David Allport
Intelligent Networked Computing Laboratory
HP Laboratories Bristol
HPL-92-118
September, 1992

expert systems, model-based diagnosis, test debugging, board test, analog circuits

Analog in-circuit program generator (APG) technology has improved to a state where the quality of tests written by APGs can hardly be met by tests written by humans [Crook 90]. One of the best examples of high-quality APGs can be found in Hewlett Packard's HP 3070 board tester.

The primary reason why automatically generated tests are non-optimal is that the input to the APG, the data that describes the board under test, contains inaccuracies. We propose a new debug methodology for tests, in which the root cause for tests that fail during debug is diagnosed and repaired.

In order to aid the test developer and to cut debug time we have developed a debug environment that supports this methodology. An expert system, named ROSALIND, analyses the results of a debug testplan and is able to take additional measurements on the testhead in order to diagnose failing tests.

A research prototype has been implemented and tested. It cuts diagnosis time, facilitates higher test qualities and makes the implementation of revision changes to the board under test much easier.

*Manufacturing Test Division, Loveland, Colorado

Fast and High Quality In-Circuit Test Development through Expert Debug

Alex Loopik[1], Dave Crook[2], Rod Browen[2], David Allport[1]

1) Hewlett Packard Laboratories, Filton Road, Bristol BS12 6QZ, Great Britain

2) Hewlett Packard Manufacturing Test Division P.O. Box 301, Loveland Colorado 80537, U.S.A.

keywords: analog in-circuit test program generation, test development methodologies, test debugging, expert systems, model-based diagnosis

Abstract

Analog in-circuit program generator (APG) technology has improved to a state where the quality of tests written by APGs can hardly be met by tests written by humans [Crook 90]. One of the best examples of high-quality APGs can be found in Hewlett-Packard's HP 3070 board tester.

The primary reason why automatically generated tests are non-optimal is that the input to the APG, the data that describes the board under test, contains inaccuracies. It requires a long debug process to catch these inaccuracies. We propose a new debug methodology for tests, in which the root cause for tests that fail during debug is diagnosed and repaired.

In order to aid the test developer and to cut debug time we have developed a debug environment that supports this methodology. An expert system, named ROSALIND, analyses the results of a debug testplan and is able to take additional measurements on the testhead in order to diagnose failing tests.

A research prototype has been implemented and tested. It cuts diagnosis time, facilitates higher test qualities and makes the implementation of revision changes to the board under test much easier.

1 Introduction

A key area in the test development process of electronic manufacturing tests is the area of debug. This is the first time where all the pieces that have been constructed during test development are fitted together and tested on a real board. Debug is the first opportunity to get real feedback on the quality of the tests and the fixture and to determine if the "known good boards" are indeed fault free. Debug is also the area where the test development engineer has to use all his skill and expertise to find ways to make failing tests work. Moreover debug is the area, being at the end of the test development process, where the test developer is usually working under time pressure to get production testing started.

As a collaboration between Hewlett-Packard Laboratories Bristol research project specialising in diagnostic problems [Allred et al. 91], [Eshghi&Preist 92], [Preist&Eshghi 92] and Hewlett-Packard Manufacturing Test Division, the developer of the HP 3065 and HP 3070 families of board testers we identified an opportunity to help the test developer during analog in-circuit debug, to further improve the quality of the tests and cut time in test development. Together we analysed the problem and formulated the new debug methodology combined with the expert system solution as presented here.

2 Analog In-Circuit Program Generator Technology

Creating highly discriminative tests that take minimal time to run on the board is the essence of developing high-quality electronic manufacturing tests. For instance, setting the appropriate test limits for an analog unpowered test is a difficult task but of crucial importance to the quality of the test.

Analog in-circuit program generator (APG) technology is now so advanced that tests generated by modern APGs are almost impossible to match by test programs written manually [Crook 90],[Crook 79].

To illustrate this, we briefly describe the capabilities of the analog program generator of the HP 3070 board tester.

For generation of the test the HP 3070 APG uses full nodal network simulation of the device under test (DUT) and its relevant environment on the board. The nodal network simulation also includes a detailed simulation of the tester itself. Other features of the HP 3070 APG include:

- Simulation of effects of other devices on the measurement of the DUT;
- Full simulation of advanced test options, such as enhancement (the tester model has 378 model primitives);
- Full simulation of automatic calibration on the tester accommodating various temperature ranges, and automatic calibration temperatures;
- Full simulation of hardware limits and error conditions;
- Statistically based error simulation;
- Monte Carlo analysis of different testers operating at different temperatures to ensure test portability.

All tests are generated to adhere to a user defined test strategy, which defines for each component the accuracy to which the component has to be tested and the time that the test is allowed to take.

We believe that for our research the HP 3070 is the best platform available, because of its state of the art analog program generator. Its high-quality APG is a prerequisite for the new debug methodology.

3 Debugging the Test Suite

Is debug still needed?

One might ask the question, given the high quality of the tests that can be generated by good APGs these days "why is debug still needed?". However, error-free tests can only be achieved in the case that the input data given to the APG gives a correct description of the board under test. With megabytes of data that are needed to describe the board under test and boards going through revisions there are bound to be some errors.

Within the tester all component details are represented and so are the electrical topology and the physical dimensions of the board. Although most of this data is taken from CAD systems nowadays, it does not mean that the data is fault-free. For example, in practice, because of concurrent design, the CAD data that is loaded onto the tester is often of an earlier release than the real board. Also, not all component data, e.g. tolerances, are critical during design and are often not decided on until later in the process, when the CAD system is no longer used.

Most test development methodologies put a lot of emphasis on getting the data right before the test generators are invoked. But, errors will slip through, fixtures will not always be built error-free and the need for debug will continue to exist.

Problems in the current practice

Test development, as mostly practised today, is a linear process: get the data that describes the board under test; run APG; build a fixture; debug the test suite. Debugging a test suite usually involves running the tests on a "known good board" and fixing the tests that fail on the this board. The place of debug in the test development process as used today is pictured in figure 1.

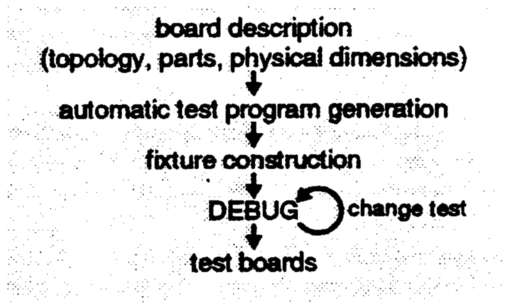
board description
(topology, parts, physical dimensions)
automatic test program generation
fixture construction
DEBUG
change test
test boards

Figure 1. the current debug process

Debug is one of the steps in this linear process. The test developers goal is to fix the failing tests. This either involves finding and repairing the physical fault (i.e a fault in the fixture) or fixing the test.

Debugging analog in-circuit tests is a highly skilful job. The situation can be compared to the process control of a complex plant: in case the automatic machinery (APG technology) breaks down, manual intervention is called for. It is not surprising that debugging analog in-circuit tests is a time consuming task. We found averages of one and a half hour per failing test for experienced test developers. We also found that most test developers are mainly concerned with getting the test to pass on the "known good board". They often work under a lot of time pressure and are therefore not primarily interested in the cause of the failure. Their mode of work usually is to try out changes to the test that have worked in the past, evaluate the effect of these changes and choose the change that is considered best.

If we relate this practice to the fault spectrum that occurs during debug, which is given in table 1, we see that in 80% of the cases, where input data is the failure cause, the root problem of the failing test is not removed.

Table 1. the fault spectrum of debug of analog in-circuit tests[1]

error%	root cause for failing test
80%	board description input data deficiencies
15%	physical problems (≈ fixture problems)
5%	complex topologies or unusual components

Besides the serious problems of debug time and test quality described above, this situation gives rise to another set of problems:

- The faulty data will in many cases also be used to generate other tests, which may not have failed during debug. These tests will be non-optimal.
- The root cause of the problem is not removed and this means that every time APG is invoked again the faulty board description data is reused.

This last issue can cause serious problems. Often the board under tests goes through a number of revisions, for which the changes need to be reflected in the test procedure. This means APG will be invoked for components in a changed topology and this can cause the same debug problems to reappear.

The role of debug

Depending on ones perspective that is taken debug can have different roles:

- The overall role of test development is "to get a fast and high-quality test suite" and debug should play a facilitating role towards that goal.
- For a test developer under time pressure this can be translated into the short term goal of: "a *good*-quality test suite in minimal debug time".
- Taking a long-term perspective another requirement becomes important: "a test suite that can easily be maintained through revision changes of the board under test".

A debug environment should facilitate all these roles.

A new debug-repair process

The aim for the debug environment will be to get high quality tests as well as high quality data and to support the different roles that debug takes in the test development process. As will be clear from the above, to get the highest quality test suite that can be maintained through revision changes of the board it is best to rely on APG technology and make sure that the input data for the APG is correct.

The debug environment will be supporting a feedback loop to correct the board description (see figure 2). This involves a way to locate the faulty data that caused the test to fail, a way to modify the board description and rerun APG on the corrected data. As board description data is the problem the fixture does not have to be changed[2].

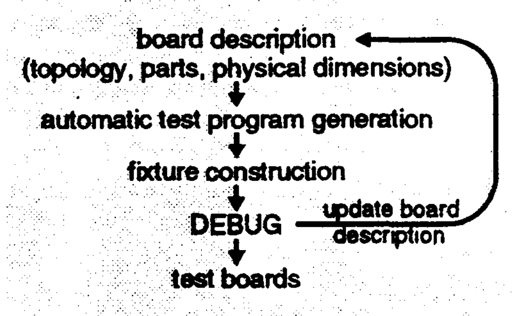

Figure 2. the new debug-repair process for board description errors

Diagnosing the Root Cause

In order to correct the board data, if that is the problem, one needs to know the root failure cause of a failing test. This means that the diagnosis of the root cause of failing tests becomes an important task in the debug process.

However, as time is critical for the test developer, this should not give him extra work. In fact one of the key issues that directed our attention to analog debug is fact that it is such a time consuming process.

Another serious problem seems to be that in order to diagnose the root cause of a failing test one needs to have a thorough understanding of the analog test theory. For test generation this expertise is given to the test developers in the APG technology, but for debug this was not (yet) the case.

1. The fault spectrum is the estimate of experts in HP's board tester manufacturing division. This fault spectrum is dependant on the test development process that is used.

2. In fact this is a constraint that is given to APG for these incremental runs.

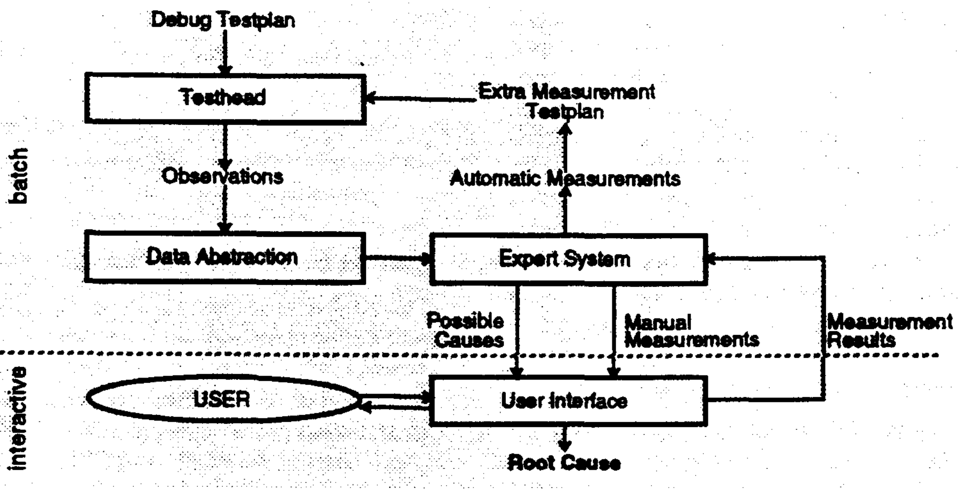

Figure 3. ROSALIND's dataflow

4 Automatic Diagnosis: the ROSALIND expert system

To overcome the two problems of lack of time and the lack of expertise we have, as part of the debug environment, developed a diagnosis system that encapsulates the expertise needed to diagnose failing tests. This expert system, called ROSALIND [Allport 92] [Loopik et al. 92], [Loopik 91] collects data on the performance of the tests, analyses that data, and outputs the most likely root cause(s) for the tests that fail.

We would like to stress that high quality APG technology is a crucial component in this process. The APG should not only be capable of generating high-quality tests, but it should also be capable of high speed test generation that can be invoked in a flexible way. The HP 3070 board tester provides, with its advanced "4th generation APG" and "HP IPG Test Consultant", is the only environment we know of which fulfils these requirements. Our project did make use of this platform and without it our research would not have been possible.

Use Model

The expert system diagnosis is automatic as far as possible, i.e without any interaction from the test developer. His involvement starts when automatic diagnosis process is finished and the system presents the results to him. His task is to accept the diagnosis suggested as the most likely (or a different one, he has the final say). He also has the option to carry out manual measurements to further aid the diagnosis process. The system presents him a list of useful measurements and allows him to feed back the results. Next the debug environment will allow him to initiate the automatic repair, i.e the change of the board description data and the rerun of APG.

For the test developer under extreme time pressure the debug environment will have another option. Rather than repairing the root cause immediately, a temporary fix to the test can be made. The system will compose such a fix, which is designed as a safe option to make the test as effective as possible in the shortest available time. Later, when the testing of boards is already taking place, the test developer can return to these tests and finish the diagnosis and repair of the root failure cause.

Data Collection and Abstraction

Before the expert system is activated ample data is collected about the behaviour of the board under test. A special debug testplan is created, which runs all the analog in-circuit tests a fair number of times on the "known good board". The fixture-board contact is cycled a few times during this process. The results of these tests are logged on the tester. These results are called the raw initial observations which are abstracted into logical statements by an data abstraction module. For example, intermittent failures, noisy test results and probe contact problems are identified by this module.

Hypothesis Generation

The abstracted initial observations are input to the expert system and based on this input it generates for each failing test a set of root cause hypotheses. To do this the expert system uses a causal model that predicts observations from root causes. Based on the observations the expert system uses abduction [Cox&Pietrzykowski86], [Eshghi&Kowalski89] to generate the root cause hypotheses. This means that a root cause hypothesis is, according to the causal mode, a possible explanation for the failing test result. This is in contrast to most "traditional" expert systems, where the expertise is represented as heuristic rules that link hypothesis (as conclusions) to symptoms (as conditions). Our approach is model-based [deKleer&Williams 87], where the rules encode a causal model of the consequences (symptoms) of the possible faults (hypotheses). The advantage of this approach are that the causal model is a more natural representation of the domain, is easier maintainable and allows additional assumptions in the reasoning process (like the single fault assumption per failing test) to be made independently.

Although all possible hypotheses that explain the failing test are generated only the hypotheses that are consistent with all observations will last. All hypotheses that are inconsistent with one or more of the other initial observations will be refuted immediately. So, not just the observations obtained from the failing test are taken into consideration but all observations from all over the board. If, for instance, a bad probe contact is a possible root cause for a certain test this will remain as a valid root cause hypothesis if it is consistent with the results of all tests that use this probe.

Part of the hypothesis generation process involves a search through the board description data searching for components and topological features that could explain the failing of the test. The main effort here is to examine parallel paths in the test setup of the component under test for possible failure causes.

Automatic Extra Measurements

Next the expert system generates a list of additional measurements that can run on the board without any manual intervention and that can possibly distinguish between the different root cause hypotheses. All automatic additional measurements are collected in an extra measurement testplan and run in one batch.

The observations of the automatic extra measurements are abstracted into logical statements in a similar way as the initial observations and fed into the expert system. There they are used either to refute hypotheses or, if that is not possible, as positive or negative evidence for the hypotheses. All the above process is done without any necessary interaction with the user.

This capability to perform additional measurements on the testhead completely automatically is a powerful feature of the ROSALIND system. It opens possibilities for advanced measurements on the topologies of failed tests, which are not easily performed by a test developer both because they are difficult to construct and because the interpretation of their results is only possible by experts in the field of analog in-circuit test theory. A few examples of such measurements currently implemented are:

- a measurements to test the voltage sensitivity of the test;
- the possibility to include different guard nodes;
- the possibility to carry out the test at different frequencies.

Manual extra measurements

If no further automatic deduction toward the root cause can be made and still more than one root cause is possible for a failing test the user can start an interactive session to finish the diagnosis process. The expert system provides the user with a list of useful manual measurements, like inspecting a components value or connectivity to the board. The user can feed the results of these measurements back into the system where they will be used to progress the diagnosis process further. Figure 3 captures ROSALIND's dataflow in a diagram.

Example

Assume the simple case of a two-wire resistor measurement. The nominal value of the resistor r69 is 4.7k Ohm ± 5%. APG has generated a two-wire test with a high limit 4.99 kOhm and a low limit 4.42 kOhm. The average measured value during debug is 4.3 kOhm with a standard deviation of 0.3. Based on this data and the board description the following root cause hypotheses are generated:

- low_value(r69)
- wrong_tolerance(r69)
- intrinsic_resistance(u204, vcc, node29)

The last hypothesis is the result of the network analysis that is carried out on the board description data. It concluded that there could be a leakage path at the test voltage in u204 (between node vcc and node29) which could be the cause of this error and can be modelled as a intrinsic resistance. As the test involves a two-wire measurement only, more complex failure causes like current splitting or source loading do not have to be considered in this case.

Based on these hypothesised root causes observation results can be predicted. One of the prediction rules for instance looks like:

```
predict( standard_value(T), [true, possible]) <-[1]
   root_cause( low_value(R),
   test_device( T, R, resistor),
   no_component_directy_parallel(R).
```

This rule predicts that a test will measure a value out

1. The rules are in Prolog syntax, which means that variables start with a capital and the same variable name within a rule signifies the same variable.

of the standard set of resistor values if the hypothesised root cause is a low resistor value and there is no component directly in parallel with that resistor. As this is the case this prediction will be made for r69:

- predict(standard_value(t_r69), [true, possible]).

t_r69 is the test that tests r69. The observation standard_value returns one of the values true, possible or false. [true, possible] means that standard_value is either true or possible.

The prediction, including the hypothesis that the prediction is based on, is transferred in the consistency maintenance module, which checks the prediction with known observations. As 4.3 kOhm is not a value out of the standard set of resistors and is not close to such a value (i.e. possible) the observation for standard_value(t_r69) is false. This causes an inconsistency with the prediction and results in the refutation of the hypothesis low_value(r69).

Another prediction rule states that if intrinsic resistance is the root cause and the family type of the library device suggests a diode like behaviour of the intrinsic resistance then the test will be voltage sensitive, i.e. if the test is repeated with a higher voltage the error in the measured value will be noticeably different.

Based on this rule, the following prediction is generated and moved to the consistency maintenance module:

- predict(votage_sensitive(t_r69), true).

This prediction does not yet have an observation associated with it. The voltage sensitive measurement can be taken automatically and is therefore requested as part of the batch process to eliminate as many as possible root cause hypotheses. Assume the test is indeed voltage sensitive, then the intrinsic_resistance(u204, vcc, node29) root cause hypothesis cannot be refuted. Although the root cause hypothesis wrong_tolerance(r69) cannot be refuted by any observation it does not explain the voltage sensitivity of the test and therefore is regarded as much less probable. In fact the only root cause hypothesis that explain the voltage sensitivity is the intrinsic_resistance(u204, vcc, node29). The system will suggest this hypothesis as the most probable one.

If the user confirms this suggestion the automatic repair can take place. The intrinsic resistance is added to the library of the u204 IC and APG is ran based on this new board description.

5 Status

The ROSALIND expert system has been developed over the past two years as a collaboration between HP's board test division MTD and HP laboratories Bristol, going through several prototype revisions. Part of the development work was the testing of the system against a extensive database of artificially constructed test cases. In this testing process we made use of an existing testhead simulator that we adapted to simulate faults that we could inject at will. This way the regression testing could be performed in a flexible and controlled manner without having to rely on a board and fixture in which we had in inject faults.

Both ROSALIND and the new debug methodology have been tested in MTD's application centre, a department that does contract test development work for internal and external users. These test runs have been successful and showed the viability of our approach.

We estimate that the full diagnosis and repair process for failing tests can be carried out within 15 minutes per failing test. This could give a speed-up of analog diagnosis of a factor six.

Although we have concentrated on the problems in the debug of analog in-circuit tests, we believe that our approach is also applicable to other areas, like digital in-circuit test and functional test. We believe, the framework of our approach is general enough to capture the diagnosis expertise available in these areas.

6 Conclusions

We have developed a new debug approach for analog in-circuit tests. We see the following benefits of our new debug methodology and expert system approach.

First there are the direct benefits to the test developers and their managers:

- Significant time savings in fault diagnosis and repair of analog tests, both for the test developer and the testhead;
- Overall higher quality tests, both for tests that fail and pass during debug;
- The test developer is more in control of the debug task;
- Availability of statistics on the overall quality of the test suite;
- The test developer does not have to understand the details of the underlying analog in-circuit test theory; and therefore there is a
- Significant decreased learning time for analog debug.

Second there are benefits that positively influence the long-term use of the test suite:

- the overall quality of the model of the board under test as represented in the tester will improve;
- the implementation of revision changes in the board (engineering change orders) will be easier.

Acknowledgments

The work on ROSALIND presented here is the result of a transatlantic collaboration between the RADHA project team in HP Labs, Bristol, UK and R&D of the Manufacturing Test Division of HP in Loveland, Colorado, USA. We thank Bruno Bertolino, John McDermid, Kave Eshghi, Kamran Firooz, Ajay Gupta, Chuck Morehouse, Alan Nelson, Chris Preist and Mike Teska for their contributions and support.

References

[Allred et al. 91] Daryl Allred, Yossi Lichtenstein, Chris Preist, Mike Bennet & Ajay Gupta: *AGATHA: an Integrated Expert System to Test and Diagnose Complex Personal Computer Boards*. Conference on Innovative Applications of Artificial Intelligence, Anaheim, CA, 1991.

[Allport 92] David Allport: *A Tractable Approach to Analogue Circuit Diagnosis.* Submitted to the 3rd Workshop on the Principles of Diagnosis, Seattle Washington, 1992

[Cox&Pietrzykowski 86] P.T. Cox & T. Pietrzykowski: *Causes for events: their computation and application.* In Proceedings of 8th conference on Computer Aided Design and Engineering, 1986.

[Crook 90] David T. Crook: *A Fourth Generation Analog Incircuit Program Generator* Proceedings of the International Test Conference, Washington DC, 1990.

[Crook 79] David T. Crook: *Analog In-Circuit Component Measurements: Problems and Solutions.* Hewlett-Packard Journal, March 1979.

[deKleer&Williams 87] J. deKleer & B. Williams: *Diagnosing Multiple Faults.* Artificial Intelligence 32:97-130, 1987.

[Eshghi&Kowalski 89] K. Eshghi & R.A. Kowalski: *Abduction Compared to Negation As Failure,* Proceedings of the Sixth Logic Programming Conference, MIT Press 1989.

[Eshghi&Preist 92] Kave Eshghi & Chris Preist: *The Cachbus experiment: Model-Bases Diagnosis applied to a Real Problem,* in Industrial Applications of Knowledge-Based Diagnosis, ed. Guida and Stefanini, Elsevier, 1992.

[Loopik et al. 92] Alex Loopik, David Allport, Chris Preist, Bruno Bertolino & Ajay Gupta: *ROSALIND - A Knowledge-Based System For Debugging Analogue In-Circuit Tests.* IEE Colloquim on "Intelligent Fault Diagnosis" London, Feb 1992.

[Loopik 91] Alex Loopik: *ROSALIND - An Analog Debug Environment, Feasibility Study and System Proposal.* Hewlett-Packard Laboratories Bristol (internal) report HPL-91-28, 1991.

[Preist&Eshghi 92] Chris Preist & Kave Eshghi: *Consitency-Based and Abductive-Based Diagnosis as Stable Models.* International Conference on Fifth Generation Computer Systems, Tokyo, Japan, 1992.[deKleer&Williams 87]