Automatic generation of in-circuit tests for board assembly defects

Citation for published version (APA):

DOI:
10.1109/ETS.2018.8400714

Document status and date:
Published: 29/05/2018

Document Version:
Accepted manuscript including changes made at the peer-review stage

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Automatic Generation of In-Circuit Tests for Board Assembly Defects

Harm van Schaaijk1,2
Prodrive Technologies
Science Park Eindhoven 5501
5692 EM Eindhoven
The Netherlands
harm.vanschaaijk@prodrive-technologies.com
martien.spierings@prodrive-technologies.com

Martien Spierings1
Technische Universiteit Eindhoven
Den Dolech 2
5612 AZ Eindhoven
The Netherlands
h.k.v.struijs@student.tue.nl
e.j.marinalisen@tue.nl

Erik Jan Marinissen2,3
IMEC
Kapeldreef 75
3001 Leuven
Belgium
erik.jan.marinalissen@imec.be

Abstract — The components and the solder joints that are made during assembly to hold components to their printed circuit board can suffer from defects and therefore need to be tested. Many research papers on board-assembly testing focus on boundary scan test, processor-controlled test, or other powered digital testing techniques, covering the indispensable circuits and that can incur damage that could have been avoided by executing a non-powered test first. In-circuit testing is a non-powered test method that applies stimuli and measures responses using probe needles. However, often used self-learning solutions for designing these tests need a known-good-board, entailing significant disadvantages. In this paper, a software tool is described that automatically generates in-circuit tests based on the product design files, without requiring probe access on every net. Furthermore, the tool indicates where on the board fault coverage is not maximal, and hence where extra probe access will improve the test quality.

1 Introduction
In board assembly, electronic components are soldered onto a printed circuit board (PCB). Boards contain a mix of passive and active discrete components, as well as (digital) integrated circuits (ICs). We distinguish components based on through-hole technology (THT) and surface-mount technology (SMT). Active test, the tool indicates where on the board fault coverage is not maximal, and hence where extra probe access will improve the test quality.

1.1 Graph Pre-Processing
The test is generated based on the design files of the device-under-test (DUT). The data that is needed is component information (e.g. reference designator (refdes [5]), type, value, tolerance) and interconnection data, i.e. the nets that describe the connections between the components on the board. Since the possibilities to test components are limited by the capabilities of the board tester, the tool needs to know the capabilities of the board tester, such as the maximum voltage that can be used safely. These steps are explained in detail in Sections 3.1.1 to 3.1.3.

1.1 Graph Collapsing
The goal of the collapsing process is to combine unreachable isolated structures into reachable compound structures. It does by exhaustively combining structures into reachable compound structures. Since tests will be generated for structures instead of components, this paper uses the term structure-under-test (SUT) instead of CUT. In order to generate tests for the structures, first a pre-processing step is performed on the graph that determines all the information that is needed to generate the test.

For further computation, we convert the netlist into an undirected multigraph \( G = (V, E) \), where \( V \) denotes the set of vertices and \( E \) the set of edges. Contrary to what one perhaps intuitively would expect, we found that in our application representing components by edges and nets by vertices works best. However, multiple-pin components, such as transistors and ICs, do not fit this representation, and therefore these components are represented by a vertex as well, with edges representing the pins of the component. In the graph net-vertices are marked to either have probe access or not. The edges are provided with a label that contains the refdes, component type, and value and tolerance.

1 Test Generation for Passive Components
After the graph is generated tests should be generated for the components in the graph. Since not every component can be tested in isolation, either due to components being in parallel or due to a lack of probe access, components will be grouped. In this paper these groups are defined as structures. A structure containing a single component is defined as an isolated structure, all other structures are defined as compound structures. Since tests will be generated for structures instead of components, this paper uses the term structure-under-test (SUT) instead of CUT. In order to generate tests for the structures, first a pre-processing step is performed on the graph that determines all the information that is needed to generate the test.

Figure 1: Netlist handling example. An example netlist and its corresponding graph are shown in Figure 1. Nets which have probe access are indicated as green vertices, while nets without probe access are shown as red vertices. The edge labels of the components are shown next to the corresponding edge.

2 System Inputs
The test is generated based on the the design files of the device-under-test (DUT). The data that is needed is component information (e.g. reference designator (refdes [5]), type, value, tolerance) and interconnection data, i.e. the nets that describe the connections between the components on the board. Since the possibilities to test components are limited by the capabilities of the board tester, the tool needs to know the capabilities of the board tester, such as the maximum voltage that can be used safely. These steps are explained in detail in Sections 3.1.1 to 3.1.3.

1.1 Graph Collapsing
The goal of the collapsing process is to combine unreachable isolated structures into reachable compound structures. It does by exhaustively combining structures into reachable compound structures. Since tests will be generated for structures instead of components, this paper uses the term structure-under-test (SUT) instead of CUT. In order to generate tests for the structures, first a pre-processing step is performed on the graph that determines all the information that is needed to generate the test.

Figure 2: Example of the collapsing process.
It might happen that a collapsed structure is still not reachable by probes on either end of a path of (s, m1) in Figure 2(c). If this is the case it can be concluded that probe access has to be added to reach the components within this structure.

3.1.2 Subgraphs

A structure is tested by connecting a source probe to one of the vertices of the structure, a measurement probe to the other, and executing a test routine. However, since the structures are tested while having been soldered onto a PCB, there might be paths other than the SUT that allow current to flow from source to measurement. These paths are called false paths.

Note that false paths can only be simple paths; a net cannot have multiple voltages at the same time and current cannot flow through it more than once. A closer look at these false paths reveals an interesting property, viz. that edges being in a false path of the SUT is both a reflexive and symmetric relation. If one observes that each false s, m-path of an edge \( e = \{s, m\} \) together form a simple cycle, the rest of the proof is trivial. From this it follows that the connected components can be split into subgraphs that for each edge can itself and all of its false paths. This split happens to be equal to the biconnected components [6] of the graph.

3.1.3 Minimal Guard Set

If a false path is not guarded it likely will influence the measurement. However, if too many nets are guarded in a false path, components are shortened between two guarded nets, which negatively influences the guard ratio. Moreover, since the impedance of these components might be dependent on the frequency, the ideal set of nets that are guarded is dependent on the selected frequency that will be used to test the structure. The goal of the minimal guard set algorithm is to find all possible sets of vertices that guard all false paths with as few components shortened between guards as possible. It does this by first discovering all paths. Next all vertices without probe access are removed since these cannot be guarded. These paths are then sorted on length to prevent that a path gets guarded twice because it coincides with a shorter path. Finally the algorithm recurses the paths, selecting new nets to guard if the path wasn’t already guarded by a previously selected net. All other vertices in the path are marked as "not-preferred" since selecting these would cause components to be shorted between guards.

3.1.4 Maximal Voltage

The maximum voltage that can be safely applied to a net is typically limited by the active components that can be reached from that net. As mentioned in Section 3.1 a maximal safe voltage can be assumed in order to prevent damaging the SUT. Between all paths between a source probe and active components are forced to a potential of \( 0 \) V, this is not necessary. Guard and measurement probes exactly do this. Therefore by discovering if there are any paths from the source probe to an active component, without crossing either an guarded or a measured vertex, the measurement can be determined to be limited by a safe voltage.

3.2 Test Mapping

Isolated structures contain only a single component, and therefore it is easy to create a library of tests that can be mapped on the isolated structures of the DUT to cover all the cases. However, for compound structures this is not true since these can consists of multiple different components, in multiple different combinations. To maximize the test coverage a generic test method is needed that tries to map a test on a structure if there is no test found in the test library. We deploy both solutions to get the best of both worlds.

3.2.1 Library Test Mapping

The library test mapping procedure maps tests from a pre-defined test library onto the structures of the DUT. This library contains product-independent tests, which all specify a so called mold which defines which combinations of components can be covered by the test, and a test mapping procedure.

Figure 3: Matching of a compound mold.

If an (expanded) structure and a mold are isomorphic [7] such that a mapping from the test measurement probes to the reachable vertices of the structure is preserved, and all of the labels of the mold and structure match, then the mold matches and the mapping procedure will continue. Figure 3 shows an example of a compound mold. Only structure 2 matches since it contains the right component types, the combined value of the capacitors is within their specified range, while the orientation is irrelevant.

3.2.2 Generic Test Method

All structures that could not be covered by a test defined in the test library are handed to the generic test method (GTM). The generic test method is a method that tries to map a test as effective as possible, regardless of what is inside that structure. The generic test method can only perform impedance measurements and is therefore more limited than the tests in the library.

To determine the frequencies to test a structure at, the generic test method has to determine at which frequencies the components in the structure are visible. Just like the components inside the structure, the impedance of the structure itself has a tolerance. Using modified nodal analysis [8] the currents through and voltages over each component can be determined. If the current through a component within the structure is large enough to not be covered by all the tolerances of the component, or the structure is large enough to not be covered by the tolerance of the structure, it is not current-dominated and its presence can be determined. Analogous, if a component is not voltage-dominated, it can be determined to be both correct and live.

Using this principle, the PCOLA-coverage can be determined for each of the test frequencies by scanning every frequency during test execution. By selecting as few as possible frequencies that result in maximal coverage, the efficiency can be optimized. In order to determine the guard ratio, the impedance of the false path is determined next. Finally a suitable voltage is selected that is lower than the determined maximal voltage, but high enough for the board tester to perform a reliable measurement. From all these potential tests, all tests that are superfluous are removed.

4 Experimental Results

We selected three boards for consumer and industrial applications from theProdive Technologies product catalog for our experiments: (1) the QA board, which is especially designed for qualification of the ICT tester hardware, (2) the EQDM board is designed towards maximum ICT access, (3) the ECU board is the first board that has a test running in production that is generated using this tool. For the largest board in this comparison, EQDM, that has over 3000 components, generating the test takes about 15 minutes on a PC with a Core i7-6820HQ CPU at 2.7 GHz and 8 GB of DDR4 RAM at 2.133 GHz.

The tests generated for the ECU board are executed using set of 15 boards. 252 of the 299 tests (84.28%) proved to be stable with a guard ratio \( \geq 2 \). 47 tests required a minor change to e.g., limits or frequency in order to get a stable test. The generated test for the QA board ran successfully, none of the tests steps proved to be unstable.

5 Conclusion

The graph representation of a board design has proven to be a versatile tool that allows for relative easy computation of the the false paths and the impedances between certain nets. It allows unreachable structures to be collapsed into reachable compound structures and decomposing the graph into its biconnected components greatly reduces the mapping complexity. The minimal guard set algorithm finds all possible combinations of vertices to guard. If these guard sets are stable tests, it can then be determined what the maximal voltage is that can be applied to a net.

Tests in a pre-defined library of tests can be mapped onto the structures in the DUT. These tests specify a so called mold, which determines what kind of structure can be covered by the test. The test mapping procedure continues. Structures not covered by tests are then handled by the generic test method. By utilizing the PCOLA-SQO properties the effectiveness of a proposed test can be optimize and guaranteed, and superfluous tests are automatically removed.

Future work entails adding support for: (1) active component, (2) shorts testing, (3) automatically selecting the most stable test.

References