DEFECT SITE PREDICTION BASED UPON STATISTICAL ANALYSIS OF FAULT SIGNATURES

A Thesis

by

MICHAEL ROBERT TRINKA

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2003

Major Subject: Computer Engineering

DEFECT SITE PREDICTION BASED UPON STATISTICAL ANALYSIS OF FAULT SIGNATURES

A Thesis

by

MICHAEL ROBERT TRINKA

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

M. Ray Mercer
(Chair of Committee)

A. L. Narasimha Reddy
(Member)

S. Bart Childs
(Member)

Chanan Singh
(Head of Department)

August 2003

Major Subject: Computer Engineering

ABSTRACT

Defect Site Prediction Based Upon Statistical Analysis of Fault Signatures.

(August 2003)

Michael Robert Trinka, B.S., Texas A&M University

Chair of Advisory Committee: Dr. M. Ray Mercer

Good failure analysis is the ability to determine the site of a circuit defect quickly and accurately. We propose a method for defect site prediction that is based on a site's probability of excitation, making no assumptions about the type of defect being analyzed. We do this by analyzing fault signatures and comparing them to the defect signature. We use this information to construct an ordered list of sites that are likely to be the site of the defect.

ACKNOWLEDGMENTS

There are so many people that have been essential to my success in pursuit of my graduate degree. First, thank you to Dr. Mercer for taking me on as a graduate student, and being such an excellent advisor. Then there is the whole Computer Engineering group, who gave me direction and contributed ideas, code, and debugging support. I am sure I will miss someone, but here are their names in no particular order: Mike Grimaila, Sooryong Lee, Jennifer Dworak, Jason Wicker, Brian Baer, Nathan Mickler, Jimmy Wingfield, Brad Cobb, Amy Wang, and David Dorsey. Next are Carolyn Warzon and Tammy Carda, who know how to get anything done. You two hold magical red tape scissors, and you have both helped me more than either of you can know. My friends at the Wesley Foundation have also played a role, always being supportive and encouraging me when I was down, and providing much needed distraction when I was feeling burned out. Of course, my mother and my in-laws have been there for me. Last but certainly not least, thank you to my wife Jenni for her constant support.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Integrated circuit manufacturers devote significant time and resources to improving the quality of the products they ship by filtering out those ICs that contain manufacturing defects. In some cases, especially when many circuits are found to contain defects, additional effort is expended in diagnosing precisely what types of defects are occurring. If the source of the defective behavior can be found, it may be possible to make improvements to the production process that will decrease the number of defective chips and improve the overall yield.

Unfortunately, a simple visual inspection of the defective integrated circuit is not an effective method for identifying the source of most manufacturing defects. Instead, analysis of the output responses of the failing chip must be used. For example, Ratford and Keating suggested in 1986 that the comparison of fault dictionary signatures to the failing circuit behavior should be implemented along with guided probe analysis to diagnose failing boards and devices [1].

These fault dictionary signatures are obtained through fault simulation of the circuit. A signature often contains information about which outputs fail when a fault is present in the circuit and a given test pattern is applied. Alternatively, the actual output values that result when the test pattern is applied may be stored. Regardless of which version is used, such a dictionary will essentially describe how the introduction of a fault affects

This thesis follows the style and format of *IEEE Transactions on Automatic Control*.

the circuit outputs for a given test pattern set.

A significant amount of previous work in the field of diagnosis has generally focused upon using fault dictionaries to find which *faults* best match the actual circuit behavior [2], [3], [4]. However, the fault model chosen may vary and often depends upon the assumptions made about what kind of defect is actually present in the circuit. This can present problems because the type of defect present in a circuit may be very different from the fault model chosen. It is therefore desirable to diagnose circuits without making any assumptions as to the types of defects that may occur.

Therefore, the most important information to obtain from diagnosis is not *fault* information, but *site* information. Once a defective site has been identified, physical examination or additional data collection from circuit simulation may be used to further classify the defect type and behavior, or to take corrective action on a faulty manufacturing process.

The common requirement for the detection of all defects is observation. Each time a site that contains a defect is observed, there is some probability that the defect occurring there is excited. We will show that we can use fault dictionary information to calculate the probability of exciting a defect present in the circuit given that the defect occurs at that site. Once this is done for all circuit sites, we will show that we can use that information to indicate which sites are most likely to be the actual source of the defective behavior with a high degree of accuracy and without the expenditure of a significant amount of resources beyond the creation of a traditional fault dictionary. Furthermore, once the site of the defect is identified, an estimate for the probability of exciting the defect given observation of the site where it occurs will also have been obtained.

PREVIOUS WORK

Over the years many researchers have suggested ways of using fault signatures and fault dictionaries to aid in circuit diagnosis. Specifically, many papers, such as [2], [3], and [4] compare the output signatures of a failing device to stuck-at fault signatures in a fault dictionary in order to identify faults that can be used to explain the failing circuit behavior. Those faults whose behavior most closely matches the behavior of the defective circuit are considered the most likely candidates.

For example, the authors of [3] introduced a technique they call partial intersection that counts the number of times faults are associated with failing bits and failing vectors. Faults with counts above a user-defined threshold are returned to the user as candidates for the identity of the defect.

Similarly, the authors of [4] rank faults based upon the number of failing outputs that cannot be caused by a fault *f* and the number of vectors for which fault *f* cannot explain either an observed failure in the circuit being diagnosed or the lack of an observed failure. These two conditions (one is based upon outputs and one is based upon vectors) are each assigned different weights. Since the merit ranks increase when the defective circuit behavior does not match that of the fault, those faults with the smallest values assigned to them are considered the most likely causes of failure.

Finally, Waicukauski and Lindbloom introduced another method of using fault dictionaries to diagnose faults in [2]. They introduced three classes of defects: those that behave exactly like stuck-at faults, those that occasionally behave like stuck-at faults at a single site, and those that affect multiple sites (such as AND and OR bridges). If the circuit behavior exactly matches that of a stuck-at fault, then the defect is considered to belong to the first class, and that fault is considered to be responsible for the defective behavior. Otherwise, the defect that explains the majority of the faulty behavior is identified. If necessary, other faults are then chosen to account for the as yet unexplained faulty behavior.

The inherent difficulty in using these models is that they try to match stuck-at fault behavior to defects that may in actuality be very different from stuck-at faults (although the authors of [2] do try to use their three categories to identify these different types of defects). In contrast, other researchers have tried to identify alternate fault models that they believe may be better able to describe many of the defects that occur in integrated circuits. For example, erroneous shorts between circuit sites often can be better modeled as bridging faults than as stuck-at faults. Both AND/OR and net-dominating bridging faults have been shown to be useful for studying these erroneous shorts [5], [6], [7], [8]. Thus, significant research has been done with respect to diagnosing bridging faults with fault dictionary information.

For example, both [9] and [10] specifically targeted bridging faults while doing diagnosis. In these cases, the authors considered the theoretical requirements for the detection of bridging faults and used these requirements to guide their diagnosis procedure. The authors of [9] noted that in order for a bridge to be detected, one of the four stuck-at faults associated with the two bridged nodes must be detected as well.

The authors of [10] expanded upon this work by taking into account additional requirements for the detection of a bridge. For example, a vector which detects both A stuck-at one and B stuck-at one cannot detect a bridge between sites A and B. Furthermore, if A stuck-at zero is detected by the same vector as B stuck-at one, then a bridge between sites A and B should cause an error to occur for that vector. These also introduce a ranking system to indicate which bridge candidates are most likely to have caused the defective behavior. While this algorithm was able to obtain good diagnosis results for the simulated bridges, it is unlikely to be very helpful when the actual defects are not bridges.

Some researchers have also tried to handle non-modeled defect behavior by returning sets of faults that together appear to explain the defective circuit behavior. For example, the authors of [11] introduced a "one-test-at-a-time" algorithm in which Dempster-Shafer statistical analysis is used to calculate which sets of faults are most likely to be the source of the defective behavior.

Our approach differs from the other approaches described here in that it uses the underlying requirements of the detection of any defect to evaluate the likelihood that a defect is located at a particular circuit site. This likelihood is calculated based upon an estimated probability of excitation given site observation. Because no assumption is made about the precise mechanism by which the defect affects circuit behavior, it is general enough to allow for the diagnosis of a variety of different types of defects. Furthermore, it calculates information not obtained by any of these other methods: the probability of excitation given site observation for the defect that is actually present in the circuit.

A NEW DIAGNOSIS ALGORITHM

No matter what type of defect occurs in an integrated circuit, two requirements must be simultaneously satisfied for that defect to be detected by a test pattern. These two requirements are *defect excitation* and *site observation.* Defect excitation refers to causing an incorrect logic value to be present at the location of the defect. This creates a difference at the defect site between the values in a defective and a non-defective circuit. For example, in order to detect a site P "stuck-at-one", the input values must be assigned in such a way as to place a logic zero at point P in the non-defective circuit. We would then say that those assigned input values *excite* the defect P stuck-at-one. In Fig. 1, this would correspond to setting the primary input A to a value of 0.

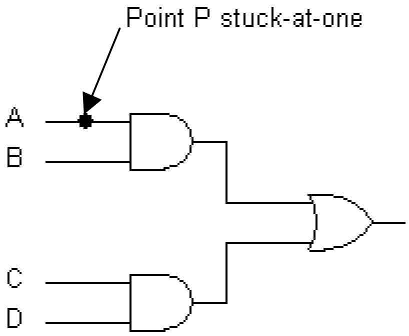

Fig. 1. Example circuit with stuck-at-one fault at point P

Unfortunately, a tester does not have access to all of the interior circuit points, and thus defect excitation is not enough to ensure detection. The incorrect value at the defect location will also need to be propagated to an output (or to a scan element) in order for the defect to be detected. In other words, the value at the defect site will need to be *observed*. In Fig. 1, setting input B to 1 and setting one or both of inputs C and D to 0 would accomplish this.

The observation requirement is common for all possible defects and faults. No matter what type of fault or defect is present in the circuit, the site where the incorrect logic value occurs must be observed at an output. In contrast, excitation requirements vary among different types of defects. For example, in order to excite a bridging defect such as the one depicted in Fig. 2, the two erroneously connected sites should be set to opposite logic values in a non-defective circuit. In this case, inputs B and C would have to be set to opposite logic values. In contrast, exciting the stuck-at fault described above merely involved setting the value of a single input.

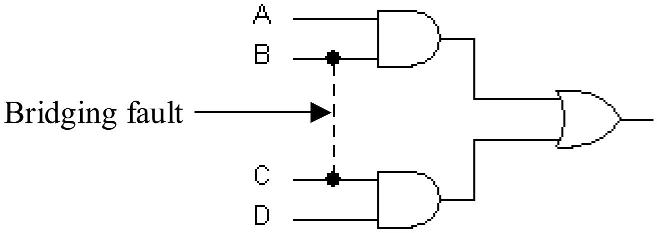

Fig. 2. Example circuit with a bridging fault

When diagnosis of a defective circuit is attempted, the type of that particular defect is not generally known a priori. Thus, we do not know what excitation requirements need to be satisfied. In contrast, we do know that a defective site will need to be observed for any pattern for which the circuit behaves incorrectly, no matter what type of defect is present. Thus, we propose a new method of circuit diagnosis based upon requiring site observation and calculating the probability of excitation.

We can use information contained within a fault dictionary to develop an estimate of the probability of exciting the defect given that that site is observed and given that the defect is actually present at that site. We will then use these probabilities of excitation to *indict* the site, or sites, which are most likely to be the site of the defect.

The set of circuit outputs where a fault is detected forms a *fault signature* for that fault. If there are N circuit outputs, then each *fault signature* is a number formed using N binary values, where zero indicates no detection at that output and one indicates detection at that output. In essence, every entry in our fault dictionary contains a *fault signature* – in the form of an integer value between 0 (no errors) and 2^N-1 (an error at every circuit output).

In certain cases, a real defect at a site can cause a good one to become a faulty zero for certain test patterns, and the same defect at the same site can cause a good zero to become a faulty one for other test patterns. Therefore, we combine stuck-at-one detections with stuck-at-zero detections to produce site detections. In particular, we combine the signature for the stuck-at-one fault at a site with the signature for the stuck-at-zero fault at that same site to form a composite signature for that site. Thus, the results from our fault simulation analysis are *site signatures* (for each test pattern) – as well as *fault signatures*.

When the actual testing of an IC is done via a tester, we capture the outputs that contain errors via *defect signatures*. Every applied pattern has a corresponding *defect signature*. A *defect signature* is compatible with a *fault signature* if the *defect signature* is equal to the *fault signature.* If this *defect signature* is compatible with the *fault signature* for one of the two stuck-at faults, F, at site S, then *defect site S is indicted* by the test pattern. Thus, the results from analysis of actual test results are *indicted sites* (for each test pattern).

Next, we combine actual test results with simulation results. For every site, we sum over all test patterns to calculate the total number of *site observations* (from simulation data) and the total number of *site indictments* (from tester data). For each site, we divide the number of *site indictments* by the number of *site observations*. The result is the *excitation probability* for the defect generating an error observed at that site (assuming excitation and observation are statistically independent).

Finally, the *excitation probabilities* for defects at each individual site in the network are analyzed. Sites are ranked according to their *defect excitation probabilities*. The site with the highest probability is the most likely candidate (assuming a single defect in the IC). Further, the relative probabilities for different sites can be used to estimate a confidence level for the predictions produced. This same information may give indications of the number of actual defect sites on the chip. Actual *defect excitation probabilities* for real defects of interest can be successively determined based upon the results from the physical failure analysis process. As more defective ICs are analyzed, and the probabilities of excitation of these actual defects are determined, predictions of defect types can be based upon characteristic *excitation probabilities*.

IMPLEMENTATION AND RESULTS

In order to conduct our experiment, it was necessary to develop a tool that would allow us to collect the necessary data. We developed a custom tool called Super Defect Analyzer, or SuperDA.

The first step in the process was to do normal stuck-at fault simulation. SuperDA implements an event driven, parallel pattern simulation technique capable of simulating 32 input vectors at a time. Once the faults are inserted, the faulty circuit outputs are compared to the good circuit outputs, obtaining the fault signature. The stuck-at one and stuck-at zero fault signatures for each site are then combined to form the site signatures. We also combine the number of times each fault for a given site was detected to find out how many times each site was observed during fault simulation. Fault simulation was done using a set of ATPG input vectors.

In order to simulate a defective part, we introduce a *surrogate* into our circuit. This surrogate can model one of three behaviors: an AND bridge, an OR bridge, or net domination. We performed three sets of analysis: one which only introduced AND and OR surrogates, one which introduced only net domination surrogates, and one which had a mixture of all three. In all cases, 5000 unique random pairs of non-feedback sites were chosen, and a random surrogate type was assigned. Once the circuit has been simulated with the surrogate inserted, the surrogate signature is obtained. The number of times each surrogate was detected was also stored.

After a surrogate signature has been collected, it is possible to compare that surrogate signature to the fault signatures collected during fault simulation. We then count the number site indictments: the number of patterns that cause an exact match between the surrogate signature and one of the two fault signatures at that site. An exact match is defined as detecting the site and the surrogate on exactly the same set of circuit outputs for a given input vector. This number can be used to find the excitation probability of a given site i by:

$$excitation\_prob[i] = \frac{\#site\_indictments[i]}{\#site\_observations[i]} \tag{1}$$

It is then possible to sort the list of all sites in the circuit based on this excitation probability. The sites at the top of the list should have the greatest probability of being the site of the actual defect. To be at the top of the list, a site must have exhibited defective behavior a large portion of the time that it was observed. Tables 1, 2, and 3 show the results of this ordering by excitation probability.

Table 1

AND/OR Surrogate Results

Circuit	Average Position	Percent in Top Ten	Percent Site is First in List	Average Position when in Top Ten
c432	5	94.57	45.63	2
c499	11	69.16	27.8	2
c880	5	88.94	31.68	2
c1355	17	63.05	15.5	4
c1908	11	70.82	19.84	3
c2670	20	66.96	22.23	3
c3540	19	79.63	25.41	3
c5315	10	77.4	25.81	3
c6288	54	38.57	8.22	4
c7552	18	65.12	18.76	3
Average	17	71.42	24.09	3

Table 2

Net Dominating Surrogate Results

Circuit	Average Position	Percent in Top Ten	Percent Site is First in List	Average Position when in Top Ten
c432	3	98.28	60.81	1
c499	9	74.08	26.69	2
c880	3	90.59	47.74	2
c1355	14	74.01	23.03	3
c1908	10	77.1	33.27	2
c2670	20	73.51	32.37	2
c3540	18	87.31	36.69	2
c5315	11	84.25	45.72	2
c6288	67	38.99	11.68	3
c7552	14	76.47	33.79	2
Average	17	77.46	35.18	2

Table 3

Mixed Surrogate Results

Circuit	**Average Position**	**Percent in Top Ten**	**Percent Site is First in List**	**Average Position when in Top Ten**
c432	4	96.03	53.28	2
c499	10	71.08	27.31	2
c880	4	90.21	40.39	2
c1355	15	69.57	20.48	3
c1908	11	71.91	24.88	3
c2670	18	71	28.92	2
c3540	17	83.45	31.81	3
c5315	10	82.3	36.59	2
c6288	59	40.72	10.5	3
c7552	15	71.99	27.43	3
Average	16	74.83	30.16	3

The average position column tells the average position of the top ranked defective site over all trials. We consider a diagnosis to be effective when the actual site of the defect is identified in the top ten suspected sites. This is both because previous researchers have used this metric, and also because of the impracticality of searching ICs for large numbers of suspected sites. Based on the number of surrogates that are detected by our simulation, we can determine how often the correct site is in the top ten indicted sites, yielding the Percent in Top Ten column. Obviously, the higher the actual defective site's ranking, the better. We therefore record how often the defective site is first on the list and the average position of the defective site when the site is in the top ten indicted sites.

As these tables illustrate, when a site is in the top ten, it is likely to be very high on the list. With the exception of c6288 (an extremely permeable circuit), we are able to identify the defective site in the top ten quite often. However, to be a viable option for real circuit diagnosis, we need to improve our results. Punishing sites that are not likely to be the defective site can do this. We can compute a "punishment probability" by using the percentage of times that a surrogate is detected but not matched exactly by the site to which we are comparing it:

$$punish\_prob = \frac{\#detections - \#matches}{\#detections} \tag{2}$$

This punishment probability can then be used to alter the excitation probability of the site by:

$$new\_prob = old\_prob - (weight * punish\_prob) \tag{3}$$

For our experiment, we ran several weighting factors, ranging from 0 to 3.6. Tables 4, 5, and 6 show the results of this new technique.

Table 4

AND/OR Surrogate Results With Punishment

Circuit	Average Position	Percent in Top Ten	Percent Site is First in List	Average Position when in Top Ten	Best Punishment Weight
c432	4	96.94	60.06	1	2.8
c499	7	94.99	37	2	0.8
c880	3	94.67	39.44	2	3.6
c1355	11	90.84	41.41	2	0.8
c1908	6	88.03	27.53	3	2
c2670	10	86.14	37.61	2	3.6
c3540	15	96.35	42.33	2	3.6
c5315	4	93.13	41.28	2	3.6
c6288	53	42.01	11.76	3	3.2
c7552	8	91.62	36.38	2	3.6
Average	12	87.47	37.48	2	2.8

Table 5

Net Dominating Surrogate Results With Punishment

Circuit	Average Position	Percent in Top Ten	Percent Site is First in List	Average Position when in Top Ten	Best Punishment Weight
c432	2	99.84	80.97	1	3.6
c499	4	98.94	44.82	2	3.6
c880	1	99.06	67.43	1	3.6
c1355	7	98.38	65.41	1	3.6
c1908	6	93.35	59.33	2	3.6
c2670	13	92.65	61.15	1	3.6
c3540	15	99.04	70.26	1	3.6
c5315	6	95.67	74.58	1	3.6
c6288	66	39.77	15.26	3	3.6
c7552	7	94.89	69.03	1	3.6
Average	13	91.16	60.82	1	3.6

Table 6
Mixed Surrogate Results With Punishment

Circuit	Average Position	Percent in Top Ten	Percent Site is First in List	Average Position when in Top Ten	Best Punishment Weight
c432	3	98.31	71.34	1	3.2
c499	5	96.82	39.02	2	0.8
c880	2	96.74	53.86	2	2.8
c1355	8	94.94	53.6	2	0.8
c1908	6	89.9	42.41	2	3.6
c2670	10	89.29	50.01	2	3.6
c3540	13	97.19	57.2	2	3.6
c5315	5	94.83	58.7	1	3.6
c6288	58	42.56	13.75	3	1.6
c7552	6	93.25	54.27	2	3.6
Average	12	89.38	49.42	2	2.7

When we compute the original excitation probability, we do not consider any additional data about the surrogate being detected. The punishment probability allows us to take in to consideration how observable the surrogate is. If the surrogate is detected, the real defective site should be the one that has the most matches, thus getting punished the least.

As these tables demonstrate, the results are dramatically improved by applying this type of punishment to the excitation probability of a site. As expected, the best weighting factor changes on a circuit-by-circuit basis. This can most readily be explained by the relative permeability of the different circuits. Also, since the net dominating surrogates can only affect one site, increasing the weight can only improve results. Therefore, a weighting factor that caters to the AND/OR surrogates would be most suited to real life, where the possible defects are of many different types.

With the addition of this punishment probability, we now have a method that consistently predicts the defective site with a high degree of accuracy.

CONCLUSION

Not knowing the type of defect present in a circuit makes deterministic excitation impossible. We have therefore presented a new algorithm for diagnosing a circuit that is based on the excitation probability of the sites in a circuit, computed from fault observation data. Our approach is unique because it is based on the fundamental requirement that a defect must be excited to be observed, and therefore is not sensitive to the type of defect being diagnosed. This is important because real circuits are not limited to a certain few types of defects. Our approach is also not sensitive to the layout of the circuit, only to its logical representation. This allows for fast processing and little more data collection than a traditional fault dictionary.

We are also able to rule out sites that are unlikely to be the defective site by using a punishment probability to get even better results. With this punishment in place, we were able to identify the defective site in our top ten list of indicted sites more than 89 percent of the time in all circuits but one. In addition, when the defective site was in the top ten, the average position of the site was 3 or less for all circuits, indicating very effective diagnosis.

REFERENCES

[1] V. Ratford and P. Keating, “Integrating Guided Probe and Fault Dictionary: An Enhanced Diagnostic Approach,” in *Proceedings International Test Conference 1986*, Washington, DC, September 1986, pp. 304-311.

[2] J. A. Waicukauski and E. Lindbloom, “Failure Diagnosis of Structured VLSI,” *IEEE Design and Test of Computers*, August 1989, pp. 49-60.

[3] R. P. Kunda, “Fault Location in Full-Scan Designs,” in *Proceedings ISTFA ’93: The 19th International Symposium for Testing and Failure Analysis*, Los Angeles, CA, November 1993, pp. 121 – 127.

[4] K. De and A. Gunda, “Failure Analysis for Full-Scan Circuits,” in *Proceedings International Test Conference 1995*, Washington, DC, October 1995, pp. 636-645.

[5] K. C. Mei, “Bridging and Stuck-at Faults,” *IEEE Transactions on Computers*, vol. C-23, no.7, 1974, pp. 720-727.

[6] S. D. Millman and E. J. McClusky, “Detecting Bridging Faults With Stuck-at Test Sets,” in *Proceedings International Test Conference 1998*, Washington, DC, September 1988, pp. 773-783.

[7] J. Saxena, K. M. Butler, H. Balachandran, D. B. Lavo, B. Chess, T. Larrabee, and F. J. Ferguson, “On Applying Non-Classical Defect Models to Automated Diagnosis,” in *Proceedings International Test Conference 1998*, Washington, D.C., October 1998, pp. 748-757.

[8] S. Ma, I. Shaik, and R. S. Fetherston, “A Comparison of Bridging Fault Simulation Methods,” in *Proceedings International Test Conference 1999*, Atlantic City, NJ, September 1999, pp. 587-595.

[9] S. D. Millman, E. J. McCluskey, and J. M. Acken, “Diagnosing CMOS Bridging Faults with Stuck-at Fault Dictionaries,” in *Proceedings International Test Conference 1990*, Washington, DC, September 1990, pp. 860-870.

[10] D. B. Lavo, B. Chess, T. Larrabee, and F. J. Ferguson, “Diagnosing Realistic Bridging Faults With Single Stuck-at Information,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 3, March 1998, pp. 255-268.

[11] D. B. Lavo, I. Hartanto, and T. Larrabee, "Multiplets, Models, and the Search for Meaning: Improving Per-Test Fault Diagnosis," in *Proceedings International Test Conference 2002*, Baltimore, MD, October 2002, pp. 250-259.

VITA

Michael Robert Trinka was born on May 28, 1976 and raised in Fort Worth, TX. He lived his first 18 years there until leaving for Texas A&M University (College Station) in the fall of 1994. It was in his sophomore year at A&M that he took his first digital design course, and it happened to be taught by Dr. M. Ray Mercer. A few years later when it was time for Michael to graduate, he remembered that class and thought it would be fun to study digital design concepts more closely. After receiving his Bachelor of Science from Texas A&M in computer engineering in May of 1999, he stayed in College Station to pursue a master's degree. It was during this time that he married his wife, Jenni, and when she was ready to graduate, they moved back to Fort Worth, where, at the time of this writing, Michael is employed by Lockheed Martin. Michael received his Master of Science degree in August of 2003. His permanent address is 2901 Sandage Ave. Apt. 306, Fort Worth, TX 76109. His email address is miketrinka@yahoo.com.