Block-Level Fault Isolation Using Partition Theory and Logic Minimization Techniques*

C.-J. Richard Shi
ECE Department, University of Iowa, Iowa City, Iowa 52242, U.S.A

Abstract—Multichip modules are emerging as a key packaging technology for mixed-signal circuits and systems. In this paper, we consider how to localize a failure within a chip boundary as rapidly as possible in order to expedite the rework process and to minimize its overall impact on manufacturing throughput and cycle time. A key contribution of this paper is to provide a unified block-level fault isolation framework for analog and digital circuits, and to show that optimum fault isolation reduces to set covering. This allows us to apply directly powerful set covering techniques and solvers developed recently in logic minimization. In addition, we present a greedy peeling heuristic with performance bound computation. Some preliminary experimental results are included to demonstrate the feasibility and performance of the proposed approach.

I. Introduction

The work here is motivated primarily by the need for efficient fault isolation in mixed-signal Multichip Modules (MCMs). Multichip modules is an attractive packaging technology for many applications. A bottleneck that limits the availability of economic MCMs is the high cost of testing. Typically, an MCM may contain over 40 large very large scale integrated circuits (VLSIs), analog and digital, interconnected by wire bonds. Because of the complexity, size and high cost of VLSI dice, a problem of interest in the manufacturing and testing of high density MCMs, especially large MCM-D types, is to locate faulty dice and then to perform a repairing or replacement [11]. Such MCM rework-diagnostic procedures are used frequently and extensively throughout the manufacturing process [2]. A primary goal is to localize a failure within a chip boundary as rapidly as possible in order to expedite the rework process and to minimize its overall impact on manufacturing throughput and cycle time. In this paper, we use *blocks* to refer to smallest replaceable units, and use *block-level fault isolation* to refer to the problem of locating a faulty block in a malfunction system.

A viable approach to digital diagnosis is to use the precalculated fault dictionary produced during test generation [1]. Unfortunately, it is difficult to apply this approach to analog and mixed-signal circuits. There are three problems. First, analog fault models are not clearly defined. Second, it is well known that analog simulation is time consuming. Even a single fault-free system-level simulation of an analog circuit is generally considered prohibitive at the transistor-level. Third, although research effort has been directed to test generation for analog and mixed-signal circuits [7], a common practice is to either generate tests randomly or based on the designer's prior experience; this leads to many unnecessary tests.

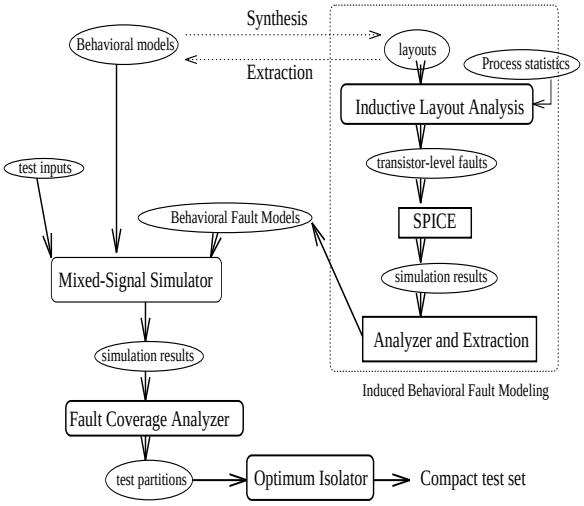

Fig. 1: The block-level fault isolation methodology.

To address the first two problems, we have developed a new fault isolation methodology, as illustrated in Fig. 1. The key idea of this methodology is to use net-oriented fault analysis for fast extraction of realistic transistor-level faults, and to use hardware description language (HDL) based behavioral models to replace both faulty and good circuit blocks for fault simulation [16].

The aim of this paper is to solve the third problem—

*This work is sponsored by U.S. Defense Advanced Research Projects Agency (DARPA) under grant number F33615-96-1-5601 from the U.S. Air Force, Wright Laboratory, Manufacturing Technology Directorate.

ASP-DAC '97
0-89791-851-7$5.00

how to find a minimum set of tests that can isolate a faulty block. For example, in Fig. 2, we assume that die 3 and 4 are faulty-free by using known good die (KGD) and all interconnects are free of faults using known good substrate (KGS). Associated with die 1 and die 2 are faults {f2,...f6} and {f7,f8,f9}, respectively. We assume that {f1} is associated with passive components (a resistor). We would like to generate a minimum set of tests that identifies the faulty block if the system fails. We note that if only one fault appears in each block, then this is the classical fault diagnosis problem. Techniques exist for constructing a small fault dictionary for digital testing [1, 14]. Unfortunately, they are tailored specifically for digital circuits. Analog fault dictionary reduction was previously studied by others [8, 12, 13] and by us [15]. In this paper, the framework originally presented in [15] is further developed to cope with block-level fault isolation.

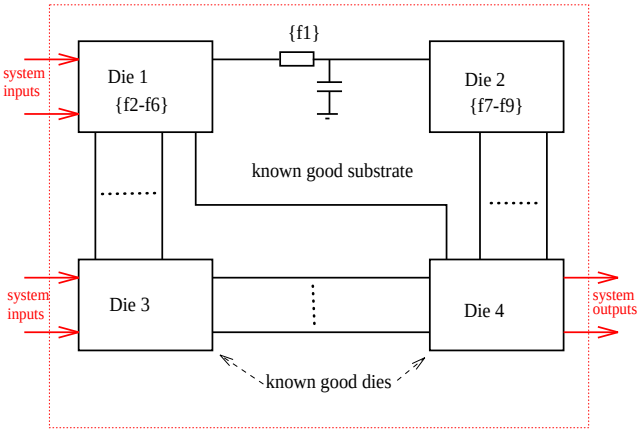

Fig. 2: Block-level fault isolation.

This paper is structured as follows: Section II presents a unified fault isolation framework for analog and digital circuits, and shows that optimum fault isolation problem reduces to the set covering problem. This enables us to use directly the powerful exact set solver developed recently for logic minimization [6] directly to our problem. An efficient greedy heuristic with performance bound computation is described in Section III. Some preliminary experimental results are reported in Section IV. Section V concludes the paper.

II. Problem Formulation

In this section, we present formal notations and some partition-theoretic results. We define a new representation for *diagnostic resolution*, and formulate the optimum fault isolation problem as the set covering problem.

A. Partition-Theoretic Background

Let E be a set of elements. A *partition* on a set E is a grouping of the elements in the set into disjoint subsets, called *groups*, such that every element belongs to exactly one group. For example, we may define a three-group partition π of the set {1,2,...,6} as $\{\{1,3,5\},\{2\},\{4,6\}\}$. The partition with just one element in each group is called the 0-partition; where the partition lumping all elements into one group is called the I-partition.

By $a \overset{\pi}{=} b$ (partition *equates* a and b), we mean that elements a and b are in the same group of partition. In the previous example, $1 \overset{\pi}{=} 3 \overset{\pi}{=} 5$, and $4 \overset{\pi}{=} 6$.

Similarly, by $a \overset{\pi}{\neq} b$ (partition π *nonequates* a and b), we mean that elements a and b are in the different groups of the partition. The *characterization set*, denoted as $\epsilon(\pi)$, of partition π is defined as the set of element-wise nonequalities. In the previous example,

$$\epsilon(\pi) = \{1 \overset{\pi}{\neq} 2, 3 \overset{\pi}{\neq} 2, 5 \overset{\pi}{\neq} 2, 1 \overset{\pi}{\neq} 4, 1 \overset{\pi}{\neq} 6, 2 \overset{\pi}{\neq} 4, 2 \overset{\pi}{\neq} 6\}$$

If x and y are two partitions (unless otherwise noted, it is assumed that partitions mentioned together are on the same set), we say that $x \geq y$ if each group of y is contained in a single group of x. For example, $\{135,2,46\} \geq \{15,3,2,46\}$.

Proposition 1 *Let x and y be two partitions. Then $x \geq y$ if and only if $\epsilon(x) \subseteq \epsilon(y)$.*

The *product*, or *intersection*, of two partitions x and y, denoted as xy, is the partition that equates a and b if and only if both x and y equates a and b. For example,

$$\{135,2,46\}\{1246,3,5\} = \{1,2,3,46,5\}$$

The observation on the following property of the intersection operation is a key to our results in this paper:

Proposition 2 *Let x and y be two partitions, and $\epsilon(x)$ and $\epsilon(y)$ be their respective characterization sets. Then $\epsilon(xy) = \epsilon(x) \cup \epsilon(y)$.*

Proposition 3 *Let x, y and w be three partitions such that $x \geq w$ and $y \geq w$. Let $z = xy$. Then $z \geq w$.*

Therefore xy is sometimes referred to as the *greatest lower bound* of x and y.

B. Test Partition and Diagnostic Resolution

We are now ready to model fault isolation using the partition-theoretic notion introduced above. Suppose that for a given circuit, the preselected fault list consists of $n-1$ faults designated $f_1, ..., f_{n-1}$. The good circuit condition is designated f_0. Let $F = \{f_0, ..., f_{n-1}\}$.

Following the convention of Simpson and Sheppard [17], a test refers to any means that can group faults in F into disjoint subsets. Thus a test introduces a partition of

F, called a *test partition*. Faults in each group of a test partition are *non-distinguishable*, or *ambiguous*, under the corresponding test. Groups in a test partition have been traditionally called *ambiguity sets* [8].

Intuitively, the test partition for a given test reflects the *diagnosability* of the test. Therefore, test partition has been previously defined as the *diagnostic resolution* of a test [1]. In this paper, the *diagnostic resolution* of a test is defined as the characterization set of its corresponding test partition.

For example, Fig. 1 shows the DC voltages measured at a test node for a given circuit after applying particular direct voltages under the good (f_0) and five fault (f_1 to f_5) conditions. To take into account the measurement noise, voltage values within the range of 0.7 Volt are considered to be indistinguishable. The test partition introduced by this test node is

$$\alpha = \{\{f_0, f_5\}, \{f_3, f_2, f_4\}, \{f_1\}\}, \tag{1}$$

where f_1 is uniquely identified at this test node. Clearly, the concept test partition is general enough to modeling the results from digital testing, AC measurement, and other types of failure analysis [17]. This allows us to treat both analog and digital circuits in the same manner.

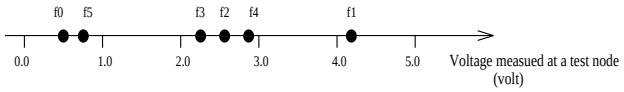

Fig. 3: A test partition and ambiguity sets.

Then the *fault resolution of a test sequence* is defined to be the union of the characterization sets of all the tests involved; this reflects precisely the capability to distinguish among faults. Intuitively, if $a \neq b$ does not appear in any set, then we cannot distinguish a and b. But if any test can distinguish a and b, then the whole test sequence can distinguish a from b.

A test is said to *achieve* a given diagnostic resolution G if the characterization set of its corresponding test partition is a superset of G. The *maximal fault resolution* is given by the following set $\mathcal{F}$ where each element corresponds to a pair of distinct faults in F. Given F with n elements, there are $\frac{1}{2}n(n-1)$ elements in $\mathcal{F}$. The *complete fault isolation* is to have a test sequence that achieves $\mathcal{F}$.

Remark: Very often, two faults in a digital system may cause the same change to the system functionality. Such faults are said to be *functionally equivalent*. For such a system, the *maximal fault resolution* is a partition where each group represents a set of *functionally equivalent* faults. For analog systems, such a situation rarely occurs. Therefore, in this paper, we assume that all the functionally equivalent faults are presented by a single fault.

C. Block-Level Fault Isolation

We are interested in locating faults downto each individual block. For the simplicity of our discussion, all the faults are assumed to be associated with blocks; interconnect faults can be treated in the same way as in [4]. Hence the structure of a multichip module, i.e., how VLSI dice and other components (if any) connect naturally defines a *structure partition* S of F. The characterization set $\epsilon(S)$ of partition S consists of pairwise nonequalities among faults in different blocks. A test (sequence) that can locate every faulty block if its characterization set is a superset of $\epsilon(S)$, i.e.,

$$\epsilon(T) \supseteq \epsilon(S).$$

Then test T is said to be a *complete faulty block isolation test*.

The optimum fault isolation problem can be stated as follows:

- Given a fault set F.
- Given a structure partition S on F.
- Given a set of test partitions T_i, $i = 1, ..., m$ on F.
- Find a minimum number of test partitions such that their intersection is no greater than S.

D. Set Covering Formulation

The characterization set $\mathcal{F}$ of the 0-partition of F consists of elements corresponding to all the pairs of distinct faults from F. Clearly $|\mathcal{F}| = n(n-1)/2$,

$$\epsilon(S) \subseteq \mathcal{F}$$

and

$$\epsilon(T_i) \subseteq \mathcal{F}, i = 1, ..., m.$$

Then the problem is to find a minimum number of $\epsilon(T_i)$, $i = 1, ..., m$ such that their union covers every element in $\epsilon(S)$.

We observe that they may exist many elements in E which never appear in $\epsilon(S)$. This is particularly true, since for multichip module fault isolation, there are not too many blocks but each block may have a significant number of faults. We can reduce the problem significantly by removing all the elements in F but not appearing in S, i.e., replacing $\epsilon(T_i)$, $i = 1, ...m$, by $\epsilon(T_i) \cap \epsilon(S)$, $i = 1, ...m$. Each $\epsilon(T_i) \cap \epsilon(S)$, $i = 1, ...m$ is a subset of $\epsilon(S)$. With this reduction, the optimum block-level isolation problem can be formulated as the following *standard* set covering problem: Given a set $\epsilon(S)$, and a collection of subsets, called clusters, $\epsilon(T_i) \cap \epsilon(S)$, $i = 1, ...m$, of $\epsilon(S)$ find a minimum number of clusters that all together cover every element in $\epsilon(S)$. A set of clusters that all together cover every element in $\epsilon(S)$ is called a *cover* of $\epsilon(S)$, where the number of clusters is the *cost* of a cover.

Remark: The classical test point selection problem in analog fault diagnosis [12] is a special case of our fault isolation problem, where the structural partition is replaced

by a partition I, and each test partition is a test node partition. Therefore, our algorithm here will be directly applicable to the classical test point selection problem, which may have increasing importance in analog design for testability.

Remark: In practice, they may exist some elements in $\epsilon(S)$ that do not appear in any of those clusters, then those elements cannot be satisfied by any solution. That means, there exists a set of faults that are not-distinguishable under the given set of tests. This can be handled by a pre-processing process.

III. Exact and Heuristic Algorithms

The reduction of optimum fault isolation to set covering enables us to apply directly powerful set covering techniques. In particular, a Zero-Suppressed Binary Decision Diagrams (ZBDD) based solver, originally developed for logic minimization [6], can be used to solve substantially large problem instances *exactly*. The focus of this section is to describe a very simple but effective greedy heuristic.

Greedy peeling is perhaps a simplest idea to set covering. It proceeds by picking the largest cluster, and then remove those elements in the cluster, and repeat the process until all the elements have been covered. Each repeat is called an ıiteration of greedy peeling.

The idea is very simple, however, Johnson proved that the number of clusters needed by greedy peeling is at most $H(|P|_{max})$ times the minimum number of clusters, where

$$H(|P|_{max}) = \sum_{i=1}^{|P|_{max}} \frac{1}{i},$$

and $|P|_{max}$ is the largest cluster [9]. It can be verified that $H(|P|_{max}) \leq ln|E| + 1$, where E is the set of elements to cover. This is a *worst case* theoretical bound, and is usually very loose for a typical problem instance. In the following, we describe a variant of the greedy peeling algorithm, which not only finds a greedy solution for set covering, but also computes a performance bound that is no greater than (usually much smaller) than $H(|P|_{max})$ in practice. Our algorithm is based on an idea of Chvátal [5].

To motivate the algorithm, we consider the following set cover example: given $E = \{e_1, ...e_7\}$ and a set $\mathcal{P}$ of clusters $P_1 = \{e_1, e_2, e_3\}$, $P_2 = \{e_1, e_4\}$, $P_3 = \{e_2, e_3, e_7\}$, $P_4 = \{e_3, e_5, e_7\}$, $P_5 = \{e_3, e_6\}$, $P_6 = \{e_4, e_5, e_7\}$, and $P_7 = \{e_4, e_6\}$. We apply greedy peeling to find a complete cover of E.

Each time when a cluster $P_i \in \mathcal{P}$ is peeled off by greedy peeling, we remove those elements in P_i from the set of clusters in $\mathcal{P}$. For convenience, we use $P_i^{(k)}$ to present the cluster P_i after $k-1$ iterations of greedy peeling. The set of non-empty cluster $P_i^{(k)}$ after $k-1$ iterations of greedy peeling is denoted by $\mathcal{P}^{(k)}$. We use C to denote the cover. Initially $k = 1$, $C = \{\}$, and $P_1^{(1)} = P_1$, $P_2^{(1)} = P_2$, $P_3^{(1)} = P_3$, $P_4^{(1)} = P_4$, $P_5^{(1)} = P_5$, $P_6^{(1)} = P_6$, and $P_7^{(1)} = P_7$. Since $P_1^{(1)}$, $P_3^{(1)}$, $P_4^{(1)}$ and $P_6^{(1)}$ all have the same largest size, greedy peeling picks an arbitrary one, say $P_6^{(1)}$, and adds it to the cover C. After removing elements of $P_6^{(1)}$ from all the clusters in $\mathcal{P}^{(1)}$, we have the following set $\mathcal{P}^{(2)}$ of non-empty clusters: $P_1^{(2)} = \{e_1, e_2, e_3\}$, $P_2^{(2)} = \{e_1\}$, $P_3^{(2)} = \{e_2, e_3\}$, $P_4^{(2)} = \{e_3\}$, $P_5^{(2)} = \{e_3, e_6\}$, and $P_7^{(2)} = \{e_6\}$. Next, $P_1^{(2)}$ has the largest size, it will be added up to the cover C. After peeling off $P_1^{(2)}$ from clusters in $\mathcal{P}^{(2)}$, there are two non-empty clusters: $P_5^{(3)} = \{e_6\}$ and $P_7^{(3)} = \{e_6\}$. We can choose $P_7^{(3)}$ to add to the cover C. By now, all the elements of E have been covered, so we have obtained a cover $C = \{P_1, P_6, P_7\}$. Its *cost* is 3.

At each iteration, when a cluster, say $P_i^{(k)}$, is picked up and added to C, the size of C increases by 1. We may say that the algorithm incurs a cost of 1. This cost can be spread evenly among the elements in the cluster $P_i^{(k)}$, i.e., each element e_i in $P_i^{(k)}$ is associated with a cost of $c_i = \frac{1}{|P_i^{(k)}|}$. In our example, at the first iteration, $P_6^{(1)}$ is peeled off, each of the three elements e_4, e_5, e_7 in $P_6^{(1)}$ is associated with a cost of $\frac{1}{3}$. At the second iteration, $P_1^{(2)}$ is peeled off. Each of three elements e_1, e_2, e_3 in $P_1^{(2)}$ is associated with a cost of $\frac{1}{3}$. Finally, P_7 is peeled off. There is only one element e_6 covered in this iteration. Thus it has a cost of 1. In summary, we have $c_1 = \frac{1}{3}$, $c_2 = \frac{1}{3}$, $c_3 = \frac{1}{3}$, $c_4 = \frac{1}{3}$, $c_5 = \frac{1}{3}$, $c_6 = 1$, and $c_7 = \frac{1}{3}$.

We can view these costs as weights associated with elements. Then the weight $P_i.cost$ of a cluster P_i can be calculated as the sum of the weights associated with all its elements. Therefore we have, $P_1.cost = 1$, $P_2.cost = \frac{2}{3}$, $P_3.cost = 1$, $P_4.cost = 1$, $P_5.cost = \frac{4}{3}$, $P_6.cost = 1$, and $P_7.cost = \frac{4}{3}$.

Let $C^{'}$ be an arbitrary cover of E, then

$$|\cup_{P_i \in C'} P_i| \geq |E|,$$

and,

$$\begin{aligned} \sum_{P_i \in C'} P_i.cost &\geq \sum_{e \in E} c_e \\ &= |C| \end{aligned}$$

Since an optimal cover, denoted by C^*, is one of these $C^{'}$, we have,

$$\sum_{P_i \in C^*} P_i.cost \geq |C|.$$

Thus,

$$|C| \leq |C^*| \frac{\sum_{P_i \in C^*} P_i.cost}{|C^*|}$$

Note that $\gamma = \frac{|C|}{|C^*|}$ is the performance bound of greedy peeling. Hence,

$$\begin{aligned}\gamma &\leq \frac{\sum_{P_i \in C^*} P_i.cost}{|C^*|} \\ &= \text{average } P_i.cost \text{ in } C^* \\ &\leq max\{P_i.cost, P_i \in \mathcal{P}\}\end{aligned}$$

That means, if we can compute a cover C by greedy peeling, then we know that the cost of an optimum cover C^* must be no greater than $|C|$ times the maximum $P_i.cost, P_i \in \mathcal{P}$. In our example, the largest $P_i.cost$ is $\frac{4}{3}$. Thus $\gamma \leq \frac{4}{3}$, and $|C^*| \geq \frac{|C|}{\gamma} > 2$. Since $|C| = 3$, we *know* that *we have an optimal solution.*

Note that $P_i.cost$ can be computed during the process of greedy peeling. This gives rise to an algorithm, called GREEDY, depicted in the pseudo-code in Fig. 4.

```
GREEDY(E, P)
 1  E' ← {}; C ← {}; k ← 0
 2  for P ∈ P do
 3     P.cost ← 0
 4  while |E'| < |E| do
 5     k ← k + 1
 6     P_k ← largest cluster in P
 7     for P ∈ P and |P| ≠ 0 do
 8        P ← P − P_k
 9        P.cost ← P.cost + |P ∩ P_k|/|P_k|
10     E' ← E' ∪ P_k
11     C ← C ∪ P_k
12  γ ← max{P.cost, P ∈ P}
13  return (C, γ)
```

Fig. 4: Greedy peeling for set covering.

Theorem 1 *Let* $H(|P|_{max}) = \sum_{i=1}^{|P|_{max}} \frac{1}{i}$, *where* $|P|_{max}$ *is the size of the first cluster picked up by greedy peeling. Let* C *be a complete cover computed by greedy peeling, and* C^* *be an optimum cover. Then* γ *computed by algorithm* GREEDY *is a performance bound of greedy peeling, i.e.,*

$$\frac{|C|}{\gamma} \leq |C^*| \leq |C|.$$

Further,

$$\gamma \leq H(|P|_{max}).$$

In our example, $\gamma = 1.33$, whereas $H(|P|_{max}) = 1.83$ and $ln|E| + 1 = 2.36$. If we use either $H(|P|_{max})$ or $ln|E| + 1$ as a performance bound, we can only know that $|C^*| \geq 2$. They do not indicate $|C| = 3$ is optimum, whereas the use of γ does. The similar behavior has been observed on a set of randomly generated examples.

IV. Experimental Results

A software prototype that implements the proposed greedy peeling algorithm has been developed in the C programming language. It also incorporates the exact set cover solver *scherzo* [6].

Table 1: Experimental results on MCNC ckts.

example			tests		greedy				optimum	
ckt	#l	#f	#p	resol	ln	cpu	log	bnd	ln	cpu
s27	38	74	2775	0.97	9	0.0	6.5	3.5	7	0.3
s208	238	208	26637	0.95	22	0.0	8.1	3.1	20	2.4
s208	238	100	6665	0.95	16	0.0	7.1	3.0	15	0.4
s298	336	100	6035	0.99	18	0.1	7.1	3.2	17	0.5
s344	395	100	6314	1.00	13	0.0	7.1	3.3	12	0.8
s400	452	100	5027	1.00	18	0.0	7.0	3.4	16	0.5
s510	549	100	5835	0.99	23	0.0	7.1	3.2	22	0.6
s526	578	100	5179	0.95	18	0.1	7.0	2.7	18	0.4
s953	1051	100	4577	0.91	16	0.1	7.0	2.9	16	0.7
s1423	1594	100	5142	0.98	16	0.0	7.0	3.1	15	0.5
s5378	5738	100	4971	0.98	18	0.0	7.0	3.2	17	0.4
s9234	9732	100	4187	0.83	11	0.0	6.8	2.7	10	0.5
s510	549	150	12144	0.992	34	0.0	9.1	3.5	28	0.8
s510	549	200	23558	0.987	30	0.0	8.1	3.4	28	1.8
s510	549	250	37415	0.993	41	0.1	8.4	3.6	41	2.6
s510	549	300	65950	0.993	43	0.1	8.8	3.5	43	5.7
s510	549	350	65950	0.993	43	0.1	8.8	3.5	43	5.9
s510	549	400	149219	0.992	43	0.1	8.8	3.5	43	5.9

#l : #lines in the ckt.
#f : #faults injected.
#p : number of fault pairs distinguished by the chosen vector set.
resol: diagnostic resolution of the chosen vector set.
ln : number of vectors needed after reduction.
log : log bound of greedy peeling.
bnd : proposed bound of greedy peeling.
cpu : CPU seconds on a 75Mhz SuperSparc with 96MB

To test the *performance* of our algorithm for larger circuits, we have run our program on a set of ISCAS89 benchmark circuits for testing [3]. To simplify the issues of memory initialization, feedback lines are cut, and each circuit is used as a combinational circuit. A simple fault simulator is used to simulate all the faults for a set of *randomly* generated test vectors, and to generate test partitions. The results are summarized in Table 1. All CPU time includes reading and writing data.

Several observations can be made from Table 1. First, this set of benchmarks are randomly diagnosable. For most circuits, the randomly generated 100 test vectors can achieve diagnostic resolution above 90%. Second, for all the examples we tested, optimum solutions can be found in several CPU seconds. (We have not be able to run even large circuits or choose many faults, mainly due to the fault simulator we used is too slow. We are currently incorporating a fast fault simulator PROOFS [10]). Third, for all the examples, the greedy algorithm performs very well in comparison with exact algorithm. It is orders of magnitude faster, and obtained near optimum solutions. Fourth, the proposed performance bound improved the known theoretical bound (one third or a half). However, this bound is still too loose. The actual differences between greedy solutions and optimum solutions are very small. It is also interesting to note the results for *s510*. The minimum test lengths are longer than that for the rest of circuits. It is known that this circuit is very difficult to test (and diagnosis).

V. Conclusions

This paper presented a unified framework of block-level optimum fault isolation for analog and digital circuits and systems. A key idea is to use test partition and to define directly the diagnostic resolution of a test as the set of fault pairs that are distinguishable under the test, whereas previously the concepts of ambiguity set have been used (which are essentially sets of faults that are not distinguishable under the test). While two concepts are directly related, the use of "distinguishable" set (characterization set) reduces optimum fault isolation to operations on sets, and leads to the formulation of the optimum fault isolation as the set covering problem; this enables us to exploit a rich set of existing techniques and theory for both fast and optimum fault isolation.

The use of ambiguity set has led to a previous formulation of optimum fault isolation as the partition intersection problem (operations on partitions) [12]. The drawbacks of such a formulation are two-folds: not efficient, and not extendable to handle fault isolation to blocks (not to individual faults)— a problem of interest to multichip module repairing.

In addition to problem formulation and understanding, we also contributed to the generic problem solving techniques. We proposed a problem-instance-dependent performance ratio of greedy peeling. This ratio is interesting, since it can be computed very efficiently (as a by-product of the greedy peeling algorithm), it measures the performance of the algorithm for a specific problem instance, and may even indicate that the solution found is indeed optimum for small problem instances.

Acknowledgment: The author is grateful to Prof. Sudhakar Reddy for several helpful discussions on this research and Dr. Olivier Coudert for accessing to his exact set covering solver.

References

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*, Computer Science Press, 1990.

[2] R. W. Bassett, P. S. Gillis, and J. J. Shushereba, "High-density CMOS multichip-module testing and diagnosis", pp. 530-539 in *Proc. International Test Conf.*, 1991.

[3] F. Beglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits", pp. 1929-1934 in *Proc. Int. Symp. Circuits and Systems*, 1989.

[4] C.-Y. Chao, H.-J. Lin, and L. Milor, "Optimal testing of VLSI analog circuits", to appear in *IEEE Trans. Computer-Aided Design*.

[5] V. Chvátal, "A greedy heuristic for the set covering problem," *Mathematics of Operations Research*, vol. 4, pp. 233-235, 1979.

[6] O. Coudert, "Two-level logic minimization: An overview", *Integration: the VLSI Journal*, vol. 17, no. 2, pp. 97–140, Oct. 1994.

[7] G. Devarayanadurg and M. Soma, "Analytical fault modeling and static test generation for analog ICs", pp. 44-47 in *Proc. IEEE Int. Conf. on Computer Aided Design*, Nov. 1994.

[8] W. Hochwald and J. D. Bastian, "A d.c. approach for analogue fault dictionary determination", *IEEE Trans. Circuits and Systems*, CAS-26, pp. 523-529, 1979.

[9] D. S. Johnson, "Approximating algorithms for combinatorial problems," *J. of Computer and System Sciences*, vol. 9, pp. 256-278, 1974.

[10] H. K. Lee and D. S. Ha, "New methods of improving parallel fault simulation in synchronous sequential circuits", pp. 10-17 in *Proc. International Conf. on CAD*, Nov. 1993.

[11] J. J. Licari, *Multichip Module Design, Fabrication, and Testing*, McGraw-Hill, Inc., 1995.

[12] P. M. Lin and Y. S. Elcherif, "Analogue circuits fault dictionary — new approaches and implementation", *Circuit Theory and Applications*, vol. 12, pp. 149-172, 1985; Reprinted in *Selected Papers on Analog Fault Diagnosis*, R. W. Liu (ed.), IEEE Press, 1987.

[13] V. C. Prasad and N. S. C. Babu, "On minimal set of test nodes for fault dictionary of analog circuit diagnosis", *J. Electronic Testing: Theory and Applications*, vol. 7, pp. 255–258, Dec. 1995.

[14] I. Pomeranz and S. M. Reddy, "On the generation of small dictionaries for fault location", pp. 272–279 in *Proc. International Conf. on Computer Aided Design*, Nov. 1992.

[15] C.-J. Shi, "Finding a minimal test set for analog fault diagnostic dictionary", pp. 303-308 in *Proc. International Workshop on Computer-Aided Design, Test and Evaluation for Dependability*, Beijing, China, July 2-3, 1996.

[16] C.-J. Shi and N. Godambe, "Behavioral fault modeling and simulation of phase-locked loops using a VHDL-A like language", pp. 245-250 in *Proc. IEEE International ASIC Conference*, Rochester, N.Y., Sept. 23-27, 1996.

[17] W. R. Simpson and J. W. Sheppard, *System Test and Diagnosis*, Kluwer Academic Publishers, 1994.