Wafer Testing with Pairwise Comparisons

Kaiyuan Huang
McGill University

Vinod K. Agarwal
McGill University

Laurence LaForge
University of Vermont

K. Thulasiraman
Concordia University

Abstract

A novel diagnosis scheme is proposed for wafer testing, in which the test access port of each die is utilized to perform comparison tests on its neighbors. A probabilistic diagnosis algorithm is presented, which correctly identifies almost all dies, even when the probability of failure of a die is larger than 0.5. The algorithm determines the status of a die according to the size of its faction, a set of dies which claim match of their responses. The algorithm is shown to be particularly suitable for constant degree structures, such as rectangular and octagonal grids. Moreover, the algorithm is designed for wafer scale structures, wherein the boundary dies do not have a complete regular (rectangular, for example) structure. The algorithm also allows for the fault coverage of the tests to be imperfect. In addition, diagnosis is done locally. Both the test time and the diagnosis time are invariant with respect to the number of dies on the wafer. Our algorithm can also tolerate some systematic errors. Finally, the saving of test cost could be significant as compared with probing, because in the existing schemes dies are probed one at a time while they are tested in parallel with our approach.

1 Introduction

The concept of built-in self-test (BIST), wherein a chip or board tests itself to determine its status (GO/NOGO), has become firmly established [1] in the semiconductor industry. A recent IEEE standard, 1149.1, on testability bus and boundary scan has led to the exploration of novel ways to incorporate BIST in digital systems [2]. Recently, Rangarajan, Fussell and Malek [3] investigated the possibility of performing probe-less testing at the wafer level.

This paper proposes a completely new approach to BIST at the wafer level with extremely high diagnostic resolution even when the yields are less than 50%. The paper also describes various implementation issues related to BIST on a wafer.

The basic idea in the new approach is to make each die compare its output with its neighbors and determine its own status based on the number of dies that agree with it. This count includes those dies which are not immediate neighbors of the die under consideration but on an agreement-chain from it. This new algorithm is proven to work extremely well even when more than 50% of dies are faulty. A detailed performance model of this approach is developed under both the binomial and negative-binomial distributions. However, due to the lack of space, the results based on the latter model will be described in another paper.

The concept of units testing each other for diagnosis purposes started from a seminal paper by Preparata, Metze and Chien [4]. Based on their model of test outcomes, wherein a good unit always had a valid outcome (i.e., tested another good (bad) unit to be so) and a bad unit could produce any outcome, these authors proved that $t < n/2$ in a n-unit system can be correctly diagnosed only if each unit is tested by at least t other units. Chwa and Hakimi [5] proposed the idea of comparison of outputs to perform testing.

Nonetheless, both the PMC and the CH approaches suffer from the fact that when these models are applied to constant degree systems, such as a rectangular grid or hypercube, t is very small (for instance, t is four in rectangular systems) compared to the size of the system. Somani, Agarwal and Avis explored the diagnosability of such systems and determined that a very large number of fault sets could be uniquely diagnosed even when their sizes are much larger than t. Further work in this direction stems from a probabilistic approach used first by Scheinerman [6] and then by Blough [7] which firmly establishes that as n tends to infinity, each unit can be correctly identified when each unit is tested by slightly more than $\log n$ units.

0731-3071/92 $03.00 © 1992 IEEE

In constant degree structures each unit is connected to less than $\log n$ other units except when n is very small. One therefore needs a different strategy for such structures. Fussell, Rangarajan and Malek [8, 3, 9] considered the concept of multiple comparison testing for constant degree structures and established that if the product of the number of test links per unit and the number of tests per link is larger than $\log n$, a very high degree of diagnostic resolution is possible. They also explored the application of this idea to wafer level testing [3]. However, certain assumptions made in their paper cannot be easily justified for real applications. First such assumption, which is implicitly made in [3], is that the test sets provide an aggregate coverage extremely close to 100% (see an explanation in Section 6). Another unrealistic assumption is that the two faulty dies almost never test each other to be "good". Finally, they assume that the coverage of each set of tests is uniform and independent of other sets of tests. It is also interesting to note that no implementation details are considered in their papers.

In an earlier paper [10], we developed the basic principles of a new diagnostic approach for constant degree structures. Our analysis was however limited to perfect test coverage, utilized more hardware than is actually necessary, and did not consider its application to wafer testing. The approach described in this paper removes all these limitations, and also analyzes those dies at the boundary of the wafer which have less than four neighbors (in the case of the rectangular grids).

Another novel aspect of the proposed approach is the ease with which it can be implemented as a BIST scheme on a wafer, thereby eliminating the need for the expensive and time consuming wafer probe testing. Using the new approach, the horizontal scribe area is used to lay the four wire IEEE testability bus along with the power and clock lines. It is assumed that the vertical scribe area is used for alignment marks needed for fabrication, and that all the new signal, power and clock tracks in the horizontal area have an order of magnitude more width than necessary to ensure their robustness. An extra signal track is used to transmit the same pseudo-random vectors to all the dies simultaneously [11]. After each transmission of a new pseudo-random vector, the resulting output response is scanned out one bit at a time from each die and compared with its four (rectangular grid connection) or eight (octagonal grid connection) neighbors' responses. Any resulting mismatch between any pair of neighbors is latched in a single flip-flop. The diagnosis is then performed locally based on the contents of these flip-flops at the end of the test procedure. At the end, a separate status flip-flop in each die contains the diagnosed GO/NOGO status of the die.

The remaider of the paper is organized as follows. In Section 2 we describe our test structure and comparison model. The diagnosis algorithm is given in Section 3 and its performance is analyzed in Section 4. Special considerations for boundary dies are given in Section 5. Finally, we compare our approach with other work in Section 6 and conclude the paper in Section 7.

2 Wafer Test Structure and Comparison Model

The main objective here is to design a test structure on each wafer that is easy to implement, inexpensive, and help provide a high quality testing of its dies. This structure is of course highly dependent on the algorithm used for testing. In later sections, we will describe our diagnosis algorithm which utilizes the test structure and analyze its performance. However, it is more convenient to start with a description of the test structure and the comparison model.

2.1 Test Structure

In [10] we proposed a test structure, wherein each unit has a comparator corresponding to each of its neighbors. Test data is broadcast to all units and each unit compares its result with those from its neighbors. Our new test structure, however, has a comparator corresponding to each pair of neighboring dies and therefore the number of comparators is reduced by half. In real implementations a comparator for a pair of neighboring dies can be placed in either of these two dies. We assume that each die is designed with internal scan [12] philosophy which allows it to be tested by pseudo-random or weighted pseudo-random patterns [13, 14]. We also assume that each die conforms with the IEEE 1149.1 testability standard with the test access port (TAP) and the associated test bus signals test_data_in (TDI), test_data_out (TDO), test_mode_select (TMS), and test_clock (TCLK). For further details on this standard, the reader is referred to [2]. With this standard, it should be possible to lay out the the four-wire testability bus in the scribe area of the wafer such that the dies can be accessed by simply accessing this bus on the periphery of the wafer.

Each pair of neighboring dies is connected by a comparator consisting of an Exclusive-OR gate and

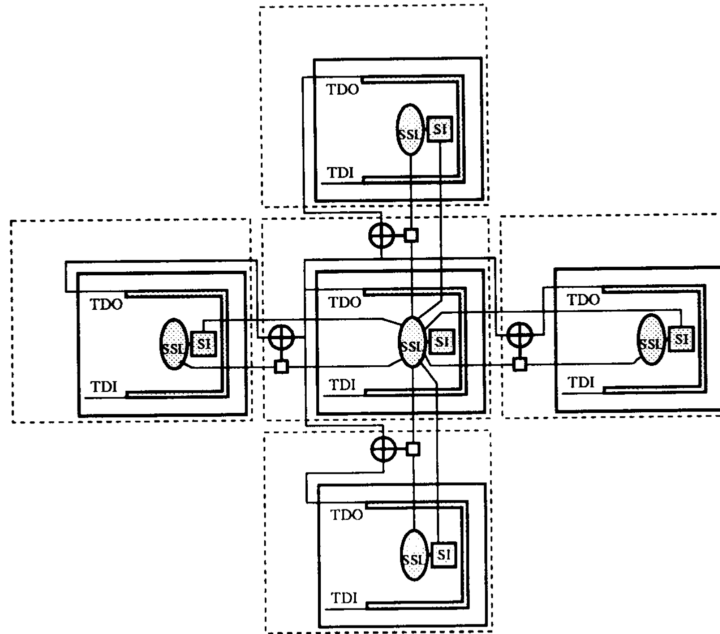

Figure 1: Comparison structure

a latch. For each test pattern supplied, the TDO outputs from the two dies are sent to the comparator and the comparison result is latched. The latch contains a 0 (1) representing agreement (disagreement) if the TDO outputs are identical (not identical) for all (at least one) test patterns. The test structure for the rectangular connection topology is shown in Figure 1, where a solid block represents a die. All dies are identical in their test structure. For the sake of clarity, only the central is shown with the complete structure. The extra signal pins, PSI, from all the dies are connected together for parallel feeding of test patterns [11]. In real implementation these comparators can be placed in the dies as shown by the dashed blocks. If we consider a dashed block to be a die, then the number of comparators per die is two. Each die has the comparators for the north and west neighbors only. This structure is a two-dimensional extension of the pair-wise compare scheme described in [11]. With this arrangement, the following test procedure is carried out in parallel by all the dies for each of the test patterns supplied:

1. Receive a pseudo-random pattern through the TDI wire and store it in the internal scan chain of each die.
2. Clock all the dies and capture the resulting test response in the internal scan chain.
3. Scan out the test response and send it through the TDO pin to all the adjacent comparators.

After receiving the test responses from the two adjacent dies, each comparator compares them bit by bit and the resulting agreement/disagreement bit is latched.

The above test-compare procedure can be executed as many times as the number of test patterns need to be applied. Note that the next test pattern is scanned in at the same time while the test response is scanned out.

At the end of the test-compare procedure, each of the comparator latches contains a 0(1) if the dies compared agree (disagree) in all (at least one) of the test patterns.

Note that the test procedure described above is executed with the help of the TAP and the testability bus.

As we can see from Figure 1, in addition to some extra pins for each die, there is also some additional circuitry, the ***status setting logic*** (SSL) and ***status indicator*** (SI). These are designed for determining the fault status of the die and their roles will be further explained in Section 3.

So far we have been describing the test structure for a die whose four neighbors all exist. For a die on the boundary of the wafer, some of its neighbors may not be available. This special situation will be addressed in Section 5.

Notice that since the test data is broadcast to all units simultaneously, all comparison tests are performed at the same time and therefore take a constant time proportional to the number of test patterns times the length of the internal scan chain. Furthermore, this arrangement does not require the storage of any fault-free data for comparison, as described in [11].

We have introduced the rectangular grid comparison topology. The comparison relationship can also take the octagonal grid topology, wherein each die compares responses with eight neighbors.

2.2 Imperfect Comparison Test Model

We assume that the distribution of failure of dies is binomial with probability p, $0 \leq p \leq 1$. The comparisons can be modeled by an undirected graph $G(V, E)$, called the ***comparison graph***, where V is the set of vertices representing the dies and E the set of edges representing comparisons. There is an edge between vertices u and v if and only if there is a comparator between u and v. The set of comparison outcomes is called the ***syndrome of the wafer*** and can be seen as a function w of edges. As we mentioned above, we will allow the comparison testing to be imperfect. We assume that the probability of $w(u, v) = 1$ for a fault-free vertex u and faulty vertex v takes on some value c, $0 \leq c \leq 1$, called the *coverage of the test.* For an easier analysis, we also assume that this probability is independent for the comparison outcomes. For any two fault-free vertices u and v, the comparison always leads to a match; in other words, we always have $w(u, v) = 0$. In addition, the probability that two faulty vertices produce all identical responses, or $w(u, v) = 0$, is assumed to take on some small value β, $0 \leq \beta \leq 1$.

3 Diagnosis Algorithm

The algorithm is similar to that presented in [10]. Unlike other probabilistic diagnosis algorithms which use various forms of local voting for determining the status of a unit, our algorithm considers the size of a bunch of units which produce identical responses and then determine the status of the whole bunch. Such a bunch of units is called a *faction.* More precisely, a *faction* is a subset V' of vertices which induces a connected subgraph of $G(V, E)$ such that 1) every edge joining two vertices in V' has the weight of 0 and 2) every edge joining a vertex in V' and a vertex outside V' has the weight of 1. A faction may take various geometric forms. In the case that a faction of fault-free vertices forms a narrow string, the probability that they are correctly identified is small with a majority voting based diagnosis algorithm because they have very few fault-free neighbors to give them credits. The geometric shape of a faction of fault-free vertices does not affect the probability of their correct identification in our algorithm, only does the number of vertices in the whole faction. As we allow the fault coverage of the test to be imperfect, a fault-free vertex may agree with a faulty one. As a result, fault-free and faulty vertices may be in the same faction. We use a threshold k for determining the status of a vertex. A vertex v is considered to be faulty if it lies in a faction of size less than or equal to k; otherwise, fault-free. The basic idea resembles that behind the t-fault diagnosis algorithm of Chwa and Hakimi [5].

Algorithm 3.1 (Diagnosis Algorithm)
For every vertex v, label v "faulty" if it is in a faction of size no larger than k; otherwise, "fault-free".

As we will see in the next section, we can have very good diagnosis performance with a small threshold value k. Therefore, the determination of the status of a vertex can be done in a small neighborhood of the vertex under consideration. This allows for a distributed implementation of the algorithm. For the threshold $k = 2$, syndrome decoding can be done locally as follows:

Algorithm 3.2 (Local Syndrome Decoding)

Step 1: *For each vertex v, label v fault-free if there are two edges incident on v with the weights of 0.*

Step 2: *For each vertex v not labeled in Step 1, label it fault-free if it has a neighbor labeled fault-free in Step 1 and the edge joining them has the weight of 0.*

Step 3: *For each vertex v not labeled, label it faulty.*

End

Theorem 3.1 *A vertex is identified to be faulty by Algorithm 3.2 if and only if it is in a faction of size no larger than 2.*

Proof: If a vertex is in a faction of size larger than 2, then either it has two neighbors in the faction or it

has a neighbor which has two neighbors in the faction. This means that it will be identified to be fault-free by Algorithm 3.2 in either Step 1 or Step 2. This implies that a vertex is identified to be faulty only if it is in a faction of size no larger than 2. On the other hand, if a vertex is in a faction of size no larger than 2, then it has at most one neighbor in the same faction and that neighbor, if it does have one, has the former as the only neighbor in the faction. This means that neither of them can be identified to be fault-free in Step 1. It is clear that they will not be identified to be fault-free in Step 2, either.

Q.E.D.

In Section 2 we described our test structure (see Figure 1). For each die, we have a latch SI, called the ***status indicator***. There is also a ***status setting logic*** SSL, which sets the status indicator according to the comparison outcomes regarding the neighbors and the contents of the SI's of the neighbors. With the above circuitry, the diagnosis algorithm can be elaborated on as follows.

Algorithm 3.3
Each unit performs the following steps in parallel:

Step 1: *Initialize each SI to 1.*

Step 2: *Perform, in co-operation with the adjacent comparators, the test-compare procedure described in Section 2.*

Step 3: *The SSL sets SI to 0 if two or more latches of the adjacent comparators have the 0 value.*

Step 4: *Send the SI value to and receive the SI values from the four neighbors.*

Step 5: *The SSL sets SI to 0 if it received a 0 SI value from any of its neighbors and if the corresponding comparison latch contains the 0 value.*

Step 6: *All units containing SI equal to 0 are declared fault-free (GO) and the remainder are declared faulty (NOGO).*

4 Performance Analysis

In this section we will analyze the performance of our diagnosis algorithm. Let $P_G(v)$ be the probability that, given v fault-free, v is correctly identified and $P_B(v)$ be the probability that, given v faulty, v is correctly identified. Later on in this section we will give analytical expressions for these two parameters. These two parameters only show the quality of diagnosis of an individual vertex. A more useful criterion for assessing an algorithm is the percentage of the vertices which are correctly identified by the algorithm. This translates to finding the ratio of the expected number of correctly identified fault-free (faulty) vertices to the expected number of fault-free (faulty) vertices in the whole system. For an easier and uniformed analysis, we will only consider torus connected grids in this section. An interesting result is that, for a torus connected grid, these ratios are identical to the probability of correct identification of a fault-free vertex and the probability of correct identification of a faulty vertex respectively. For non-torus connected grids, both $P_G(v)$ and $P_B(v)$ vary for dies v on or near the boundaries of the wafer. This will be discussed in the next section. Let g and f be the number of fault-free vertices and the number of faulty vertices, respectively. Let g_c and f_c be the number of correctly identified faulty-free vertices and the number of correctly identified faulty vertices. These numbers are random variables. We use $E[h]$ to represent the expectation of a random variable h.

Theorem 4.1 *For a torus connected grid, the following holds:*

1. $\frac{E[g_c]}{E[g]} = P_G(v)$.
2. $\frac{E[f_c]}{E[f]} = P_B(v)$.

where v is a vertex on the grid.

Proof: Due to space limitation, the reader is referred to [15] for the proof.

As we stated above, we assume that systems are torus connected. This keeps $P_G(v)$, as well as $P_B(v)$, the same for every vertex v and therefore simplifies the calculation of the above ratios. In the following we will deduce analytical expressions for $P_G(v)$ and $P_B(v)$ for torus connected grids. We will only consider the case of $k = 2$. We also assume that a grid has at least four columns and at least four rows.

4.1 Rectangular Grids

If vertex v is fault-free, there are three classes of local status-syndrome patterns corresponding to the event that v is in a faction of size no larger than 2. There is a single pattern in the first class in which v disagrees with all its four neighbors. As v is fault-free, the four neighbors must be all faulty. This status-syndrome pattern is depicted in Figure 2(a). In this and the following figures, a dark block represents a faulty vertex, a grey block a vertex possibly faulty or

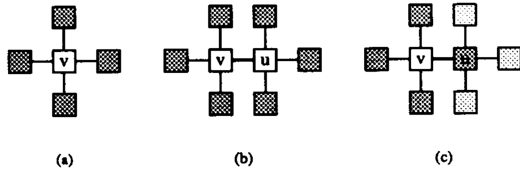

Figure 2: Status-syndrome patterns for a fault-free vertex

fault-free and a light block a fault-free vertex. An internal edge of a faction is represented by a thick line. In the second class of status-syndrome patterns v has a single fault-free neighbor u and they are surrounded by faulty vertices. Note that the fault-free neighbor can be anyone of the four neighbors of v. The pattern in which u is to the east of v is depicted in Figure 2(b). Rotating the picture round v 90 degrees at a time produces the other three patterns in this class. In the third class of status-syndrome patterns v is completely surrounded by faulty vertices but exactly one of them, say u, agrees with v and all the other three neighbors of u disagree with u. As u is faulty, these three neighbors may be either fault-free or faulty. The pattern in which u is to the east of v is depicted in Figure 2(c). Rotating the picture round v 90 degrees at a time produces the other three patterns. It is easy to see that these three classes of status-syndrome patterns exhaust all the possibilities that v is in a faction of size no larger than 2. Note that all these patterns are pairwise disjoint. This can be shown as follows. Assume, to the contrary, that there are two patterns which are identical. Overlay the two patterns and align them according to the positions of v. We can find either a vertex which is faulty in one pattern and fault-free in the other or an edge whose weight is 1 in one pattern and 0 in the other. This contradicts to the assumption that the two patterns are identical. Let $q_1^0(v)$, $q_2^0(v)$ and $q_3^0(v)$ be the probabilities that a fault-free vertex v is in a faction corresponding to a first class status-syndrome pattern, to a second class status-syndrome pattern and to a third class status-syndrome pattern, respectively. From the above discussions we have the following.

Lemma 4.1 $q_1^0(v) = p^4c^4$.

Proof: Consider Figure 2(a). Vertex v is given to be fault-free. There are four vertices which are faulty. This can happen with the probability p^4. Four edges each joining a fault-free vertex and a faulty vertex have the weights of 1. The probability that this event happens is c^4, according to our model. The probability of occurrence for the whole event is therefore p^4c^4.

Q.E.D.

Lemma 4.2 $q_2^0(v) = 4(1-p)p^6c^6$.

Proof: Consider Figure 2(b). There is another vertex which is fault-free. The probability that this vertex is fault-free is $1-p$. Six vertices are faulty and the probability of its occurrence is p^6. We can also note that six edges each joining a fault-free vertex and a faulty vertex have the weights of 1 and its probability of occurrence is c^6. Note that the weight on the edge joining v and u is definitely 0 given that they are both fault-free. The probability of occurrence for the status-syndrome pattern shown in Figure 2(b) is therefore $(1-p)p^6c^6$. As there are four symmetric patterns in the same class, we have the factor of 4.

Q.E.D.

Lemma 4.3 $q_3^0(v) = 4(1-c)c^3p^4((1-p)c+p(1-\beta))^3$.

Proof: The proof is similar to the proof of Lemma 4.2. Note that there are three edges each joining the faulty vertex u and a vertex which may be faulty or fault-free. The probability that such an edge has the weight of 1 is $(1-p)c+p(1-\beta)$. The first term corresponds to the case of the other end of the edge being fault-free and the second term to the case of the other end of the edge being faulty.

Q.E.D.

Theorem 4.2 $P_G(v) = 1-(q_1^0(v)+q_2^0(v)+q_3^0(v))$.

Proof: From the above discussions we know that the three classes of status-syndrome patterns are pairwise disjoint and exhaust all the patterns corresponding to the event that v is fault-free and is in a faction of size no larger than 2. Hence the theorem.

Q.E.D.

Using the above results, we calculated the probability $P_G(v)$ for wide ranges of p and c values with β set at 0.01. The results are listed in Table 1. Comparing the entries, we can see that the probability that a fault-free vertex is correctly identified is even higher when the fault coverage of the test is not perfect. We may even find from Table 1 that the lower the coverage, the higher is the probability $P_G(v)$. This phenomenon can be explained as follows. We can see from our algorithm that if a fault-free vertex agrees with more neighbors then it is more likely that it will be correctly identified. A fault-free neighbor will certainly produce the same responses and therefore they agree with each other. If the coverage is perfect, a fault-free vertex will always disagree with a faulty one. If

Table 1: Performance on identification of a good vertex with $\beta = 0.01$

p	$P_G(v)$									
	$c =$ 1	$c =$ 0.99	$c =$ 0.98	$c =$ 0.97	$c =$ 0.96	$c =$ 0.95	$c =$ 0.94	$c =$ 0.93	$c =$ 0.92	$c =$ 0.91
0.1	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
0.2	0.998	0.998	0.998	0.998	0.998	0.998	0.998	0.998	0.998	0.998
0.3	0.990	0.990	0.990	0.990	0.990	0.991	0.991	0.991	0.991	0.991
0.4	0.965	0.965	0.966	0.967	0.967	0.968	0.969	0.969	0.970	0.971
0.5	0.906	0.908	0.910	0.912	0.917	0.916	0.918	0.921	0.923	0.925
0.6	0.796	0.800	0.805	0.810	0.814	0.819	0.824	0.828	0.833	0.838
0.7	0.619	0.627	0.636	0.645	0.653	0.662	0.671	0.679	0.688	0.696

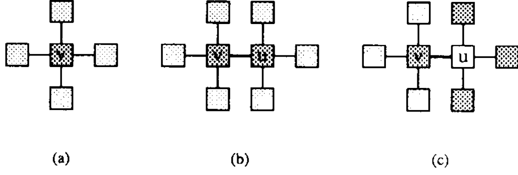

Figure 3: Status-syndrome patterns for a faulty vertex

the coverage is not perfect, then there is the possibility that a fault-free vertex may agree with a faulty one and therefore this fault-free vertex is more likely to be correctly identified. In the extreme case that the coverage is 0, fault-free vertices will always agree with faulty ones and therefore all fault-free vertices can be correctly identified. This, however, does not mean that the overall quality of diagnosis is higher when the coverage is lower. When the coverage is lower, the probability of correct identification for a faulty vertex will be also lower. What is interesting to see is that, for reasonable values of c, we can correctly identify fault-free and faulty vertices both with high probabilities. In the following, we will analyze the performance of our algorithm on the identification of faulty vertices.

Assume that vertex v is faulty. Similar to the case that v is fault-free, there are three classes of local status-syndrome patterns corresponding to the event that v is in a faction of size no larger than 2. The first class contains a single pattern and is shown in Figure 3(a). All the four edges incident on v have the weights of 1. The vertices adjacent to v may be either fault-free or faulty. In the second class of patterns v has a unique faulty neighbor u which agrees with v. The pattern with u to the east of v is depicted in Figure 3(b). Rotating the picture 90 degrees at a time produces the other three symmetric patterns. In the third class of patterns v has a unique fault-free neighbor u which agrees with v. The pattern with u to the east of v is depicted in Figure 3(c). Rotating the picture 90 degrees at a time produces the other three symmetric patterns in the class. Similar to the case that v is fault-free we discussed above, we can show that these three classes of patterns are pairwise disjoint and they exhaust all the patterns corresponding to the event that v is in a faction of size no larger than 2. Let $q_1^1(v)$ be the probability that, given that v is faulty, v is in a faction of size no larger than 2 corresponding to a status-syndrome pattern in the first class, $q_2^1(v)$ the probability corresponding to a status-syndrome pattern in the second class and $q_3^1(v)$ the probability corresponding to a status-syndrome pattern in the third class. The following conclusions follow from these definitions and discussions. The proofs are omitted due to space limitation.

Lemma 4.4 $q_1^1 = ((1-p)c + p(1-\beta))^4$.

Lemma 4.5 $q_2^1(v) = 4p\beta((1-p)c + p(1-\beta))^6$.

Lemma 4.6 $q_3^1(v) = 4(1-p)(1-c)p^3c^3((1-p)c + p(1-\beta))^3$.

Theorem 4.3 $P_B(v) = q_1^1(v) + q_2^1(v) + q_3^1(v)$.

The probability $P_B(v)$ is listed in Table 2 for a wide variety of values of p and c with β set at 0.01. We can see from the table that a faulty vertex can also be correctly identified with high probability. $P_B(v)$ even increases with the increase of p. This phenomenon can be explained as follows. To identify a faulty vertex, it is desirable that this faulty vertex disagrees with its neighbors. A faulty vertex disagrees with a fault-free neighbor with probability c while it disagrees with a faulty neighbor with probability $1-\beta$. How $P_B(v)$ varies with p depends on how c compares

Table 2: Performance on identification of a faulty vertex with $\beta = 0.01$

p	$P_B(v)$									
	$c =$ 1	$c =$ 0.99	$c =$ 0.98	$c =$ 0.97	$c =$ 0.96	$c =$ 0.95	$c =$ 0.94	$c =$ 0.93	$c =$ 0.92	$c =$ 0.91
0.1	0.99998	0.964	0.930	0.896	0.863	0.831	0.800	0.770	0.741	0.713
0.2	0.9999	0.968	0.938	0.907	0.878	0.849	0.821	0.794	0.768	0.742
0.3	0.9998	0.973	0.946	0.920	0.897	0.869	0.844	0.820	0.796	0.773
0.4	0.9997	0.977	0.955	0.930	0.911	0.890	0.869	0.848	0.827	0.807
0.5	0.9996	0.982	0.964	0.947	0.929	0.912	0.895	0.877	0.861	0.844
0.6	0.9994	0.986	0.973	0.960	0.947	0.934	0.921	0.908	0.895	0.881
0.7	0.9991	0.991	0.982	0.974	0.965	0.956	0.947	0.938	0.928	0.919

Table 3: Variation of $P_G(v)$ and $P_B(v)$ with β when $p = 0.5$, $c = 0.99$

	$\beta =$ 0.01	$\beta =$ 0.001	$\beta =$ 0.0001	$\beta =$ 0.00001	$\beta =$ 0.000001
P_G	0.908	0.908	0.908	0.908	0.908
P_B	0.982	0.983	0.983	0.983	0.983

with β. For the calculations of $P_B(v)$, we set β at 0.01. This implies that c is no larger than $1 - \beta$ for the values of c used in the table. We do not have any statistics on the possible value of β. The reason for setting β at this value is that we believe $1-\beta$ is likely to be larger than c. When a faulty vertex compares responses with another faulty vertex, two uncertainties are involved. When a faulty vertex compares responses with a fault-free vertex, only one uncertainty is involved. As a faulty vertex may be in one of many different failure modes, β may take a very small value. Table 3 shows how $P_G(v)$ and $P_B(v)$ vary with the change in the value of β. We can see from the table that $P_G(v)$ is immune to the change of β (to be accurate, there is a slight change of $P_G(v)$) while $P_B(v)$ increases a bit with the decrease of β. Rangarajan, Fussell and Malek's approach [3] requires that β be extremely small as we will show in Section 6.

4.2 Octagonal Grids

Our algorithm is also applicable to an octagonal grid. Due to space limitation, however, detailed analytic results are omitted. Table 4 shows $P_G(v)$ and $P_B(v)$ values for a variety of values of p with $c = 0.99$ and $\beta = 0.01$. As we can see from the table, $P_G(v)$ improves from that for a rectangular grid.

5 Boundary Considerations for Non-Torus Connected Grids

In the last section we considered torus connected grids, where every die was in the identical geographical position so far as the connection topology was concerned and a uniformed solution was reached for the probabilities of correct identification $P_G(v)$ and $P_B(v)$. As a wafer is a plate in nature, a non-torus connection topology is more natural and easier for implementation. However, the dies on the boundaries are not in an identical geographical position with the dies in the inner part of the wafer and therefore special considerations are needed. In the following we will consider two specific issues associated with non-torus connected grids. First, we will discuss a special circuit arrangement for the open pins from the dies on the boundaries arising from lack of neighboring dies to certain sides. Second, we will analyze the effects of lack of such neighbors on the probabilities of correct identification of boundary dies.

5.1 Arrangement for Open Pins

In principle, open output pins can be left open without any negative effects. For the side of a boundary die to which a neighboring die does not exist, there is either an open TDO pin (input to the comparator placed in the boundary die under consideration) or an open comparison outcome pin (supposedly input from a non-existing comparator latch placed in the non-existing neighboring die), depending on whether the comparator between the boundary die and its missing neighbor is placed in the boundary die or in the missing neighboring die, fictitiously assuming the missing die exists. There is also an open SI input pin supposedly from the missing neighbor. A simple way to handle these open pins is to fix these pins to the logical

Table 4: Performance on an octagonal grid with $c = 0.99$ and $\beta = 0.01$

	$p=0.1$	$p=0.2$	$p=0.3$	$p=0.4$	$p=0.5$	$p=0.6$	$p=0.7$
$P_G(v)$	0.99999999	0.999997	0.99992	0.9991	0.994	0.972	0.899
$P_B(v)$	0.930	0.937	0.944	0.951	0.958	0.965	0.973

"1". When the open TDO pin is connected to "1", a fixed logical value, in the case that the comparator is placed in the boundary die, the comparison outcome will be "disagreement", so will it when the comparison outcome input pin is fixed to "1" in the case that the comparator is supposedly placed in the missing neighbor of the boundary die. Fixing the open SI input pin to "1" is equivalent to assuming that there is a fictitious neighbor but this neighbor is not labeled "fault-free" in Step 1 of Algorithm 3.2.

With the above arrangement of the open pins on the boundary dies, the test procedure and local syndrome decoding algorithm described in the previous sections can be equally executed on all dies, whether boundary or inner dies. It is also worth noting that the layout of each die is kept the same and therefore the same set of masks can be used for all the dies during the manufacturing process.

5.2 Performance on Boundary Dies

As we stated above, the test procedure and local decoding algorithm can be executed on all dies on a non-torus connected grid with the arrangement of the open pins described above. It is easy to see that $P_G(v)$ degrades somehow while $P_B(v)$ improves somewhat for a die v on or near the boundaries of the wafer. For $k = 2$, the dies whose probabilities of correct identification are affected will be limited to those who are on the boundaries of the wafer or who have a neighbor on the boundaries. For a wafer with a large number of dies, the number of such dies only counts for a small fraction of the total number of dies and therefore the ratio of the expected number of correctly identified fault-free (faulty) dies to the expected number of fault-free (faulty) dies determined for torus connected grids can be taken as an estimate of the corresponding ratio for non-torus connected grids. For a wafer with a small number of dies, the deviations of $P_G(v)$ and $P_B(v)$ due to the irregularities on the boundaries count significantly and must be dealt with separately. We claim without elaboration, due to space limitation, that the deviations of $P_G(v)$ and $P_B(v)$ for a die which is not on the boundary but has a neighbor on the boundary from those for a die further inside the wafer are extremely small. For those dies right on the boundary we have analytic results on the performance deviation. However, due to space limitation, we only provide an example as given in Table 5, where β is set to 0.01 and c to 0.95, and the boundary die has only two neighbors.

6 Comparison with Other Work

In this section we show some of the advantages of our approach over Rangarajan, Fussell and Malek's [3].

First, our approach allows for a test set of moderately high fault coverage while theirs requires that the aggregate coverage of the test sets be extremely close to 100%. This can be shown as follows: In their approach the fault coverage of a test, c, can be small, for example, 0.1. However, a large number (r) of such small coverage tests are performed corresponding to each link. They assume that the total coverage of r such tests is at least $C = 1-(1-c)^r$. For $r = 500$ and $c = 0.1$ (these values were used in their examples), C is about $1 - 10^{-23}$. This is extremely close to 1.

Second, their approach cannot tolerate systematic errors but ours can to a large extent. Their results were based on the assumption that two defective dies almost never produce the same responses. As they pointed out in their implementation considerations section, systematic errors might invalidate this assumption and therefore their voting scheme. Every design has some week points which are more likely subject to failures in the event of processing parameter variations than other parts of the design. This is one source of systematic errors. Systematic errors may also occur if a flawed mask or dusted mask is stepped across the wafer for the exposure of each die. When systematic errors occur, the dies are likely to develop defects of the same failure modes and therefore they are likely to produce the same responses. Let β be the probability that two defective dies produce identical responses for all the r tests. From their assumptions we have $\beta = (1-C)^2 + (c/f)^{2r}$, where f is the number

Table 5: Performance on a boundary vertex

	$p = 0.1$	$p = 0.2$	$p = 0.3$	$p = 0.4$	$p = 0.5$	$p = 0.6$	$p = 0.7$
$P_G(v)$	0.990	0.958	0.902	0.817	0.702	0.559	0.397
$P_B(v)$	0.912	0.922	0.932	0.943	0.954	0.967	0.978

of failure modes a defective die can be in. The first item corresponds to the case that the faults in the two defective dies are not covered by the tests and the second item corresponds to the case that the faults in the two dies are covered by the tests but these two dies are in the same failure modes. For $c = 0.1$, $r = 500$ and $f = 1000$ (these values were used in their examples), β is about 10^{-46}. This is extremely close to 0. As we have seen in Section 4, our approach allows β to be many orders of magnitude larger.

7 Conclusion

We have presented our test structure and diagnosis algorithm intended for wafer testing. The test structure is very simple and easy for implementation. The algorithm is also simple and needs only local information for determining the status of a die. The test time and the diagnosis time are both invariant with respect to the number of dies on the wafer. Yet the algorithm is able to correctly identify the status of almost all dies as we have seen in the last sections. The algorithm is shown to be suitable for constant degree connection structures such as rectangular and octagonal grids. It also allows for the fault coverage of the test to be imperfect. As we have seen, we can achieve a high quality of diagnosis with tests of moderately high coverage. Since our algorithm works well when the probability of failure is larger than 0.5, it can cope with wafers of less than 50% yields. Another advantage of our algorithm is that it can tolerate systematic errors to a large extent. We believe our scheme is very practical and worth implementing.

References

[1] E. Eichelberger and T. Williams, "A logic structure for LSI testability," in *Proc. Design Automation Conf.*, pp. 462–468, 1977.

[2] *IEEE Standard Test Access Port and Boundary-Scan Architecture*, IEEE, May 1990. IEEE Std 1149.1.

[3] S. Rangarajan, D. Fussell, and M. Malek, "Built-in testing of integrated circuit wafers," *IEEE Trans. Comput.*, vol. 39, pp. 195–205, Feb. 1990.

[4] F. P. Preparata, G. Metze, and R. T. Chien, "On the connection assignment problem of diagnosable systems," *IEEE Trans. Electron. Comput.*, vol. EC-16, pp. 848–854, Dec. 1967.

[5] K. Chwa and S. L. Hakimi, "Schemes for fault-tolerant computing: A comparison of modularly redundant and *t*-diagnosable systems," *Information and Control*, vol. 49, pp. 212–239, Jun. 1981.

[6] E. R. Scheinerman, "Almost sure fault tolerance in random graphs," *SIAM J. Comput.*, vol. 16, pp. 1124–1134, 1987.

[7] D. M. Blough, *Fault Detection and Diagnosis in Multiprocessor systems*. PhD thesis, The Johns Hopkins University, 1988.

[8] D. Fussell and S. Rangarajan, "Probabilistic diagnosis of multiprocessor systems with arbitrary connectivity," in *Digest FTCS-19*, pp. 560–565, 1989.

[9] S. Rangarajan and D. Fussell, "Probabilistic diagnosis algorithm tailored to system topology," in *Digest FTCS-21*, pp. 230–237, 1991.

[10] K. Huang, V. K. Agarwal, L. LaForge, and K. Thulasiraman, "A diagnosis algorithm for constant degree structures and its application to VLSI testing," submitted to the IEEE Trans. Paral. and Dist. Syst.

[11] B. Nadeau-Dostie, P. S. Wilcox, and V. K. Agarwal, "A scan-based BIST technique using pair-wise compare of identical components," in *Proc. 4th CSI/IEEE Int. Symp. VLSI Design*, Jan. 1991.

[12] Williams and J. M. Angell, "Enhancing testability of large-scale integrated circuits via test points and additional logic," *ibid*, pp. 46–59, Jan. 1973.

[13] P. Agarwal and V. Agarwal, "Probabilistic analysis of random test generation method for irredundant combinational logic circuits," *IEEE Trans. Comput.*, pp. 691–695, Jul. 1975.

[14] H. D. Schnurmann, E. Lindbloom, and R. G. Carpenter, "The weighted random test-pattern generator," *IEEE Trans. Comput.*, pp. 695–700, Jul. 1975.

[15] K. Huang, *System Diagnosis with Application to VLSI Testing*. PhD thesis, McGill University, 1992. (in preparation).