

Chapter 2

Pre-bond Testing of the Silicon Interposer

In order to minimize the yield loss results from the stacking of good dies on a defective interposer, it is necessary to test the interposer before die stacking. In addition, the interposer is the least expensive component in the entire stack. As a result, pre-bond testing of the silicon interposer is needed to reduce cost; in this way, we can avoid a cheap, but faulty interposer rendering the expensive 2.5D IC.

In this chapter, we present an efficient solution to locate defects in the passive interposer at the pre-bond stage. The proposed test architecture uses e-fuses that can be programmed through voltage pulses outside the range of normal circuit operation. These e-fuses are inserted into the interposer. When the interposer is tested in the pre-bond stage, e-fuses are used to connect separated interconnects so that test paths can be formed for interconnect testing. Afterward, they are programmed to disconnect the functional interconnects as needed. Therefore, the functionality of the interposer is not affected once the dies are stacked on it. We also describe an assembly and test flow that facilitates pre-bond interposer test. This assembly and test flow has been validated in a production environment. The concept of a weighted critical area is introduced to reduce test time. The proposed solution therefore obviates the need for a more expensive active interposer. In order to fully determine the location of each e-fuse and the order of the functional interconnects in a test path, we present a test-path design algorithm. The proposed algorithm can generate all test paths for interconnect testing. We present HSPICE simulation results to demonstrate the effectiveness of the pre-bond test solution. Test-path designs are presented to highlight the efficiency of the test-path design algorithm. The advantage of using the weighted critical area is also analyzed.

The remainder of this chapter is organized as follows. Section 2.1 describes the structure of e-fuse. Section 2.2 first defines a “die footprint” and presents the proposed test architecture. Next, the assembly/test flow and test procedures are discussed. Finally, the concept of weighted critical area is introduced. In Sect. 2.3, we investigate test-path design and describe the proposed test-path design algorithm.

Section 2.4 presents HSPICE simulation and test-path design results. The benefits of using the weighted critical area is also analyzed for an interposer from industry. Finally, Sect. 2.5 concludes the chapter.

2.1 Background

The interposer cannot be tested easily before it is stacked with other dies due to several reasons [1]. Testing the interposer requires the targeting of both types of interconnects: horizontal and vertical. If both sides of the interposer can be probed at the same time, pre-bond interposer testing can be easily accomplished. However, double-sided probing of the interposer is not feasible today due to limitations related to wafer handling and probe-card design. In addition, it is difficult to probe the micro-bumps on the top side of the interposer due to their high density. Interconnect testing requires connecting the interconnects in a loop so that a logic value can be applied at one end, and the propagated value can be observed at the other end. However, interconnects are separated and independent from each other at the pre-bond stage. Therefore, new and innovative solutions are needed for pre-bond testing.

In this chapter, e-fuses are used for pre-bond testing of silicon interposers. E-fuses have been used extensively in a variety of applications due to their programmability [2]. They can be programmed using a voltage pulse; the schematic of an e-fuse is shown in Fig. 2.1. Before programming, the resistance of the e-fuse is small and it can be treated as an interconnect wire. When a high voltage is applied to point B, a high current blows open the e-fuse. The state of the e-fuse changes and it is programmed. After programming, the resistance of the e-fuse is high enough (above $10^4 \Omega$) that it can be treated as an open [3]. Due to this large resistance difference before and after programming, an e-fuse is used in the proposed test architecture to implement pre-bond testing.

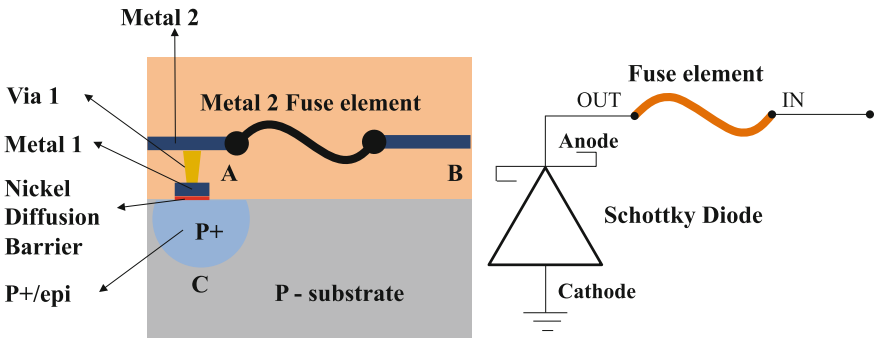


Fig. 2.1 Illustration of the structure of an e-fuse based on a Schottky contact

There are three kinds of e-fuse devices: (i) formed with a silicide gate poly-Si electrode material (gate-electrode-fuse), (ii) constructed from Cu wire (Cu-fuse), and (iii) consisting of a Cu-via (via-fuse) [4–6]. When we consider future device scaling, the gate-electrode-fuse may be problematic. Therefore, Cu-fuse and via-fuse devices are widely used in industry [5]. Our industry collaborators report that the resistance of the Cu-fuse is as small as $25\ \Omega$ before programming. After programming, its resistance is as large as several $G\Omega$ s [5]. The low on-off resistance ratio of the Cu-fuse makes it an excellent candidate for the proposed test method. As a result, we consider a Cu-fuse in this work.

In order to program an e-fuse in the interposer, we must have a discharge path to the substrate. However, since the interposer is a passive device, no programmable field-effect transistor (FET) can be used as a discharge path. Industry practice today is to form a substrate tap/tie using one or two additional masks such that a Schottky contact can be formed to act as a discharge path [7]. As shown in Fig. 2.1, consider the e-fuse to have 2 terminals IN and OUT. OUT is connected to the anode of the Schottky contact. When testing the interconnects/TSVs, the e-fuse is not programmed and signals flow through it. After testing, a high current pulse is applied at IN to the e-fuse such that it is programmed open at OUT and the current is discharged through the cathode of the Schottky contact into the silicon substrate. With the use of Schottky contact, the interposer is still passive after the inclusion of e-fuses. It is especially important to enable the pre-bond testing of passive interposers since active interposers increase production cost. Our proposed method is therefore targeted at passive interposers.

2.2 Proposed Test Architecture and Procedures

In this section, we define the concept of die footprint and introduce the proposed test architecture. The test flow is described and test procedures are listed. Finally, the weighted critical area is introduced to save test time.

2.2.1 Definition of Die Footprint

Although the dies have not been mounted on the interposer at the pre-bond stage, the design and fabrication of the interposer interconnects is typically based on the information (i.e., layout) of dies that will later be placed on the interposer [8]. Therefore, each die has a corresponding footprint on the interposer for subsequent bonding. The die footprint refers to the micro-pillar area to which the top die connects to on an interposer. It contains all interconnects to and from the specific die. The directionality of interconnects is not taken into consideration because they are typically just back-end of line (BEOL)/metal lines and can be treated as direction-independent

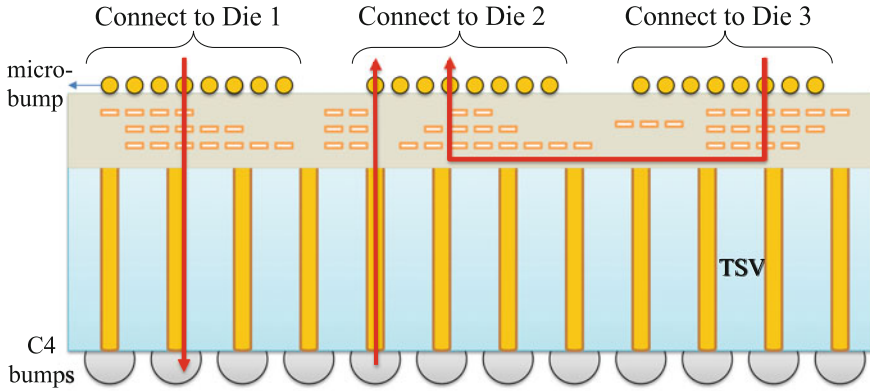


Fig. 2.2 Illustration of an interposer example with functional paths

before the top die is assembled. As a result, although dies are not considered in the pre-bond stage, the corresponding die footprint should be taken into consideration.

Before the dies are stacked on the interposer, each interconnect in the interposer connects two separate die footprints. In the proposed test architecture, separated interconnects are connected via e-fuses. Based on the structure of interposer shown in Fig. 2.2, it is clear that interposer testing requires the following: (1) testing of horizontal interconnects and (2) testing of vertical interconnects. Since it is difficult to probe both sides of the interposer simultaneously, these two types of interconnects are tested separately in the proposed method.

The length of each e-fuse is restricted in order to save space for functional interconnects. In particular, for horizontal die-to-die interconnects, consider a test path through four die footprints as $2 \rightarrow 1 \rightarrow 4 \rightarrow 3$. Since the interconnect $2 \rightarrow 1$ ends at Die 1 footprint and interconnect $1 \rightarrow 4$ starts at Die 1 footprint, connecting them using an e-fuse within the Die 1 footprint involves less interconnect length than doing it outside the die footprint. Therefore, each e-fuse can only be located within a single die footprint for testing horizontal interconnects; thus, for two horizontal interconnects to be connected by an e-fuse, they must share at least one die footprint. For vertical interconnects, the TSVs are spread out through the interposer. Connecting them through e-fuses outside the die footprint is a cheaper option.

2.2.2 Test Architecture

The general test architecture to target horizontal interconnects is shown in Fig. 2.3. An interposer example is utilized to illustrate the proposed test architecture. E-fuses are inserted inside the interposer, and horizontal interconnects, which are not connected in functional mode, are now connected for testing. In Fig. 2.3, two test paths are

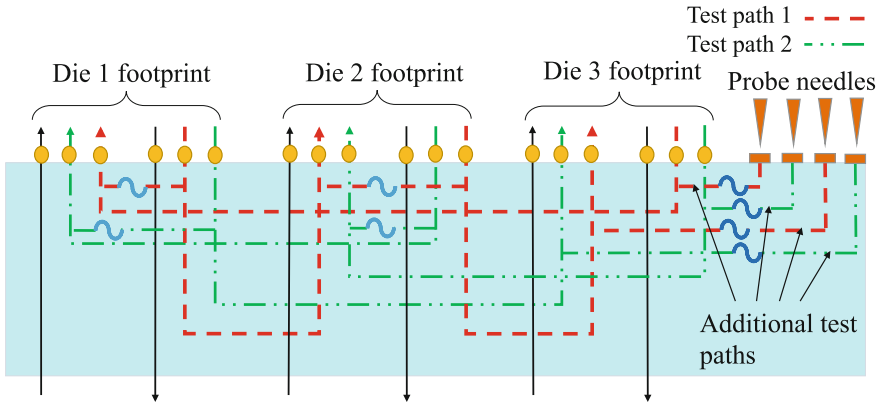


Fig. 2.3 Illustration of test paths for targeting horizontal interconnects

formed that start from the I/O ports in Die 3 footprint, pass through all three die footprints, and end in Die 3 footprint. It can be seen that the e-fuses are only located within a single die footprint; there are no e-fuses that span two die footprints. Test Path 1 starts from Die 3 footprint, passes through Die 1 footprint and then Die 2 footprint, and ends in Die 3 footprint. Similarly, Test Path 2 starts from Die 3 footprint, passes through Die 2 footprint and then Die 1 footprint, and ends in Die 3 footprint. Once the test paths are formed, test patterns are applied to the test paths and the horizontal interconnects can be tested. After all horizontal interconnects are tested, the e-fuses will be programmed and their resistance will increase to a significantly large value. As a result, the e-fuses can be viewed as opens in the interposer. When the dies are later mounted on the interposer, these programmed e-fuses will not affect chip functionality.

Because the micro-bumps on the top of the interposer have very high density, it is difficult to use them to probe the interposer. Therefore, additional test paths are inserted into the interposer for probing purposes. As shown in Fig. 2.3, each additional test path is composed of a probe pad and an e-fuse. These probe pads and e-fuses are referred to as additional pads and additional e-fuses, respectively. In order to probe the interposer using standard probe needles, additional pads are fabricated with a larger pitch. These pads are placed at the boundary of the interposer and test paths are routed to probe pads based on shortest distance.

In the test path design algorithm (Sect. 2.3), interconnect selection is specifically used to determine how to form the test path, and test paths are appropriately routed to probe pads based on shortest distance. This means that nets at the center of the interposer are connected together to form a long test path and then routed to probe pads at the die edges. Because the additional e-fuses are different from the e-fuses that connect the functional interconnects, these two types of e-fuses are represented in different ways in Fig. 2.3. The additional e-fuses are also programmed after horizontal

interconnect testing is complete so that the functionality of the 2.5D IC is not affected once the dies are stacked.

Interconnects with fanouts can also be easily incorporated in the proposed approach. Each fanout can be viewed as multiple individual interconnects during test-path design. Then, these interconnects will be operated as consecutive interconnects included in one test path. For example, consider a net that has three terminals, A, B, and C. Then, they are taken as two separate interconnects: AB and BC. Subsequently, a test path A-B-C can be formed. This test path will be combined with other test paths generated via the test-path design method. In this way, there will be no more misleading test outcomes.

The general test architecture for vertical interconnects is shown in Fig. 2.4. E-fuses are inserted inside the interposer and separated vertical interconnects in the same die footprint are then connected. Once the test paths are formed between vertical interconnects, test patterns can be applied to these test paths from the bottom of the interposer. Since C4 bumps at the bottom of the interposer can be probed directly with standard probe needles, no additional test paths are required to test vertical interconnects. Once all vertical interconnects have been tested, e-fuses will be programmed and treated as opens, once again disconnecting separate functional interconnects inside the interposer.

Note that it is not necessary for only two TSVs to be connected to each other. Multiple TSVs can also be connected together: one of these TSVs will be selected as a master TSV and the other TSVs can be taken as slave TSVs. All slave TSVs are connected to the master TSV using e-fuses while no connections are added between two slave TSVs. Then, these TSVs can be tested by applying patterns to the master TSV and observing responses from different slave TSVs. The TSVs will be connected using e-fuses in the upper-most metal layer of the interposer. This covers all the BEOL from the micro-bumps to the TSVs. By connecting the TSVs for vertical testing, we

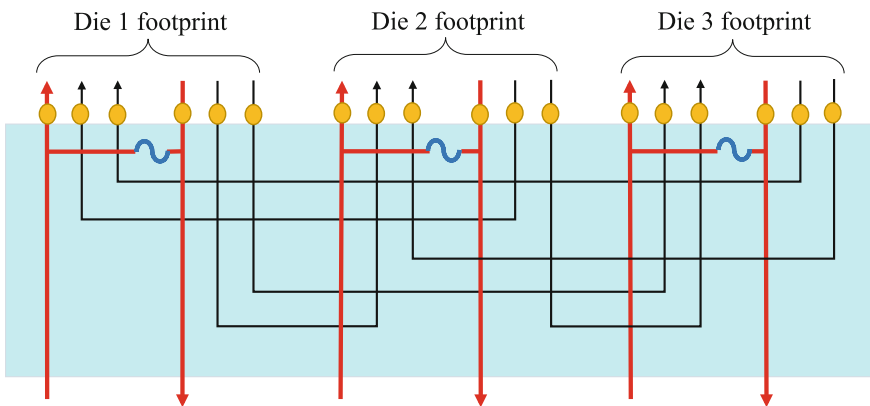


Fig. 2.4 Illustration of test paths for targeting vertical interconnects

are able to probe from the C4 bump side to determine whether the entire TSV path is a pass or a fail.

In summary, there are three types of e-fuses in the proposed test architecture. The first two types of e-fuses are used to form the test paths, one for horizontal interconnects and the other for vertical interconnects. The third type of e-fuse is used in the additional test paths in order to probe the top side of the interposer. The various types of e-fuses utilize different control signals and are programmed at different times.

2.2.3 Assembly and Test Flow

In contrast to 2D ICs, which require only one wafer-level test insertion, we need both final thick-wafer test and thinned-wafer test steps in order to determine a known good interposer (KGI) for 2.5D ICs. The assembly process flow is shown in Fig. 2.5. Thick-wafer test is used to test horizontal interconnects. The BEOL layers such as Cu and Al are first plated on the thick wafer; then, the horizontal interconnects are tested prior to wafer fab-out. If standard probe cards are used, then additional probe pads are required. However, a fine-pitch probe card with compliant probes can be used to test directly on the micro-bump pads having a NiAu finish. Research is currently underway and progress has been reported on such fine-pitch probe cards [9]. This eliminates the need for additional probe pads and access to program the e-fuses that is used to form longer test paths. Afterward, wafers are bonded to temporary glass carriers from the front.

The testing of thinned wafers is a significant challenge, but it can be addressed using techniques that are currently deployed in industry [1, 10, 11]. Compared to previous methods, e-fuses are less costly and thereby can be used more easily to form test paths. Thin-wafer test is implemented after TSV reveal to test vertical interconnects. Since all e-fuses are connected to the silicon substrate using the top

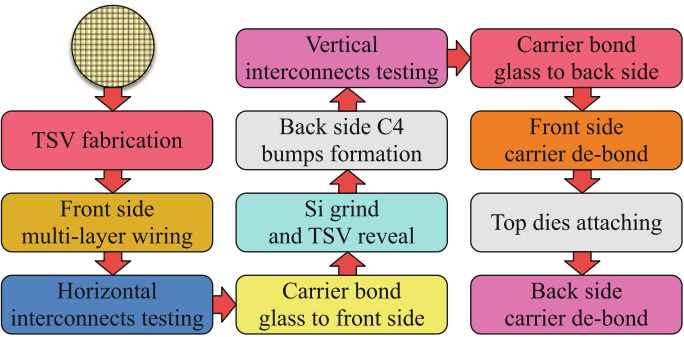


Fig. 2.5 Assembly and test flow for pre-bond testing

0.5/1 μm region, they will not be impacted during the thinning process. The C4 bumps are completed prior to thin-wafer test to prevent probing on backside Cu pads. Then, vertical interconnects are tested via e-fuses; afterward, thinned wafers are bonded to another temporary glass carrier from the back. This approach has been verified and implemented in a production environment [12]. Finally, two glass carriers are de-bonded separately; top dies are attached to the front side of the interposer and the stack is assembled onto a package substrate. In order to support this two-sided test for interposers, the preferred assembly process would be with a wafer support system so that the interposers are on a carrier wafer and can be probed on either side.

During this process, the additional cost is a well implant mask to form the Schottky contact. This solution has been evaluated for the upcoming interposer process flow to form the substrate connection. The mask cost is not a significant factor compared to the yield improvement that can be achieved. The rest of the e-fuse fabrication is standard BEOL processing. The size of an individual e-fuse is smaller than 0.366 μm^2 . With thousands of e-fuses fabricated on the interposer, the total area is negligible compared to a typical interposer size of $25 \times 25 \text{ mm}^2$. Even if the e-fuses are placed at area with high-density interconnects, the e-fuse-based routing can be easily carried out and the signal integrity is not affected.

2.2.4 Test Procedures

Three types of faults can typically occur in the horizontal interconnects: open faults, inter-bridge faults, and inner-bridge faults. *Open faults* refer to any hard or resistive opens, regardless of the fault location. *Inter-bridge faults* refer to bridge faults that occur between two test paths; for example, an interconnect in one test path is shorted with another interconnect in a different test path. *Inner-bridge faults* refer to bridge faults that occur inside a single test path; e.g., an interconnect in a test path is shorted with another interconnect from the same test path. In order to identify the type and location of these faults, specific test procedures are required.

Since each test path can be viewed as a single interconnect, a traditional interconnect test algorithm (True/Complement algorithm) [13] is used here. Similarly, the detection of open faults and inter-bridge faults does not depend on whether a test path or a functional interconnect is viewed as a single interconnect. Therefore, open faults and inter-bridge faults can also be detected by the True/Complement algorithm. With this algorithm, test patterns are applied to one end of a test path and test responses are observed at the other end. However, this method is not applicable for the detection of inner-bridge faults. For example, suppose a logic 1 is applied to a test path at one end. No matter whether the test path is fault-free or contains inner-bridge faults, a logic 1 will be observed at the other end. Therefore, the faulty response is identical to the correct response. As a result, a new test procedure has been developed for inner-bridge faults.

A test path is shown in Fig. 2.6. This test path is composed of four functional interconnects, and these interconnects are connected by e-fuses *a*, *b*, and *c*. To detect

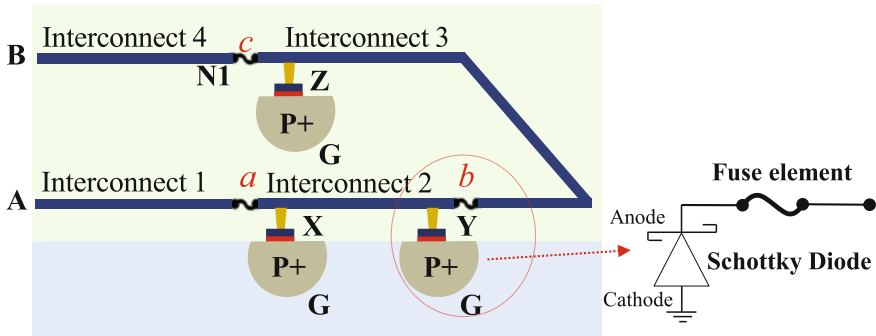


Fig. 2.6 Illustration of a test path

the inner-bridge faults, the e-fuse that is closest to point A is different from the other e-fuses; specifically, its diode is at the end rather than at the head. The proposed test procedures are listed:

(1) Apply logic 1 at point A and observe the response at point B. If there are no faults in the test path, a logic 1 will be observed at point B.

(2) Program the e-fuse *a* so that the test path is open at *a*'s position. Apply logic 1 at point A and interconnect 2 is discharged through point X. The e-fuse is connected to the silicon substrate through a Schottky contact; the Schottky contact provides a discharge path only when high current is applied during programming to form an Ohmic contact. Since the currents used for testing are much smaller in magnitude, the Schottky contact ensures that the e-fuse will not provide a discharge path prior to the e-fuses being programmed. Therefore, points Y and Z are not discharge paths when e-fuse *b* and e-fuse *c* are not programmed. If a logic 1 is observed at point B, it indicates that an inner-bridge fault exists between interconnect 1 and the remaining interconnects. Otherwise, point B is grounded via point X and interconnect 1 does not have an inner-bridge fault.

(3) Apply logic 1 to point B and program the e-fuse *b* so that the test path is open at *b*'s position. Interconnect 2 is discharged through point Y. Then, point B is taken as an observation point and the voltage level is measured at point B. If logic 0 is observed at point B, it indicates that an inner-bridge fault exists between interconnect 2 and the remaining interconnects. Otherwise, the charge in the remaining interconnects should remain high and a weak logic 1 should be observed at point B with a slow discharging rate.

(4) Apply logic 1 to point B and program e-fuse *c* so that the test path is open at *c*'s position. Interconnect 3 is discharged through point Z. Then, the voltage level is measured at point B. The same method used in step 3 is utilized to detect whether there is an inner-bridge fault between interconnect 3 and interconnect 4.

Since different e-fuses can be programmed using different programming currents, e-fuses in a path can be programmed one by one, and each time the desired e-fuse can be selected.

Then, above procedure can easily be generalized to the case of any number (n) of interconnects. All these interconnects are connected using e-fuses in the same manner as described for the special case of four interconnects. The e-fuse that connects interconnect i , $1 \leq i \leq n$, is first programmed so that interconnect i is disconnected from the rest of the test path. Then, interconnect i is tested. If there are no inner-bridge faults between i and the rest of the test path, the e-fuse that connects interconnect $i + 1$ will next be programmed so that $i + 1$ is disconnected from the rest of the test path. With this procedure, all inner-bridge faults for a single test path are efficiently tested and the n interconnects are separated through e-fuse programming after testing. Following this line of reasoning, we can easily see that all inner-bridge faults are tested.

Based on data from GLOBALFOUNDRIES, the pitch of the vertical interconnects is large ($240 \mu\text{m}$). Therefore, bridge faults are unlikely to occur in vertical interconnects, and thus, the True/Complement algorithm is not necessary for vertical interconnect testing. During the TSV processing, there are three types of faults that can occur in vertical interconnects: break faults, void faults, and pinhole faults. Figure 2.7 shows images of the three faults. The physical mechanisms underlying these three faults are discussed in [14]. Break faults and void faults increase the TSV resistance by different amounts based on the defect dimensions. Pinholes create a conduction path from the TSV to the substrate, resulting in a leakage fault. In order to detect these faults, a high voltage is applied to the vertical test paths. The three types of faults can be detected and identified based on the differences in response voltage levels.

Note that only the e-fuses involved in inner-bridge fault testing are required to be programmed one by one. For all other faults, the e-fuses can be programmed at the same time and do not have cross-dependencies. For the e-fuses involved in the testing of inner-bridge faults, we do not require additional scheduling to determine the programming order. The programming order has already been determined by the test path: The ordering of the interconnects in a single test path is the programming order of the e-fuses connected between two interconnects.

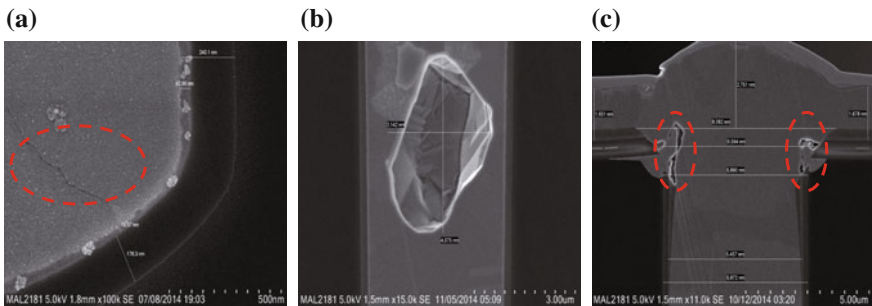


Fig. 2.7 TSV defects: **a** break, **b** void, **c** pinhole

2.2.5 Weighted Critical Area

Because the interconnects are not uniformly distributed on the interposer, some areas of the interposer may have a higher density of interconnects. An area with more interconnects, which can be flagged based on technology, process, and yield considerations, is referred to as weighted critical area. If a test path contains functional interconnects that connect to the same weighted critical area, it is referred to as a dense test path; otherwise, it is called a nondense test path. In nondense test paths, no two interconnects are in the same area, and hence, inner-bridge faults are less likely to occur; thus, it is not necessary to program e-fuses in that test path serially and programming time is reduced.

In a typical interposer, the location of micro-bumps can be easily tracked from the design netlist. Note that some micro-bumps (dummy micro-bumps) are not connected to any interconnect. Therefore, a weighted critical area is only determined by the information of active micro-bumps. We have developed a systematic method to determine a weighted critical area. Let the length and the width of an interposer be L_{int} and W_{int} . The interposer is divided into $n \times n$ subareas, where n is defined as the division resolution of the interposer. The number of active micro-bumps is accounted in each subarea (num_i); the average value (avg) and standard deviation (σ) are calculated. If num_i is greater or equal to $avg + \sigma$, then the subarea i is defined as a weighted critical area. As shown in Fig. 2.8, the interposer is divided into 16 subareas; avg and σ are calculated as 1.5 and 1.414. Therefore, subareas 1, 2, 4, 5, and 6 are determined to constitute the weighted critical area.

In order to analyze the influence of weighted critical area, we use the acronym “WCA” to refer to the testing approach that considers the weighted critical area; “nWCA” refers to the testing approach that does not consider weighted critical area. In nWCA, since bridge faults may exist between any two interconnects, all test paths must be tested for inner-bridge faults. In contrast, only dense test paths are tested for inner-bridge faults in WCA. Therefore, the use of WCA can help us to reduce the testing time.

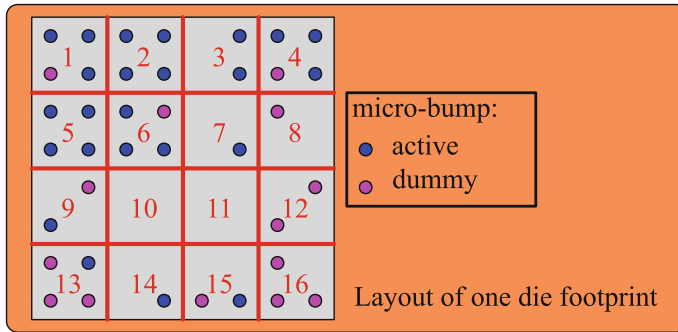


Fig. 2.8 Illustration of the subarea division for a die footprint

2.3 Test-Path Design

A test path both begins and ends at a die footprint. It contains several functional interconnects, which are connected by e-fuses. However, the positions of the e-fuses and the order of the functional interconnects in a test path have to be determined. In order to minimize the total test cost, the test paths must be carefully designed.

2.3.1 Optimization Problem

Consider an interposer with a set of M die footprints. The number of interconnects from Die i footprint to Die j footprint is w_{ij} . The objective is to select an optimal test-path design that minimizes the total test cost. The total test cost is determined by two parameters: the number of e-fuses and the number of additional probe pads. Therefore, the objective of minimizing the total test cost can be achieved by using fewer e-fuses and probe pads. As shown in Fig. 2.9, if a test path is broken into two separated test paths, e-fuse 1 connecting interconnect 1 and interconnect 2 will be eliminated. However, the total number of e-fuses and probe pads increases due to the two pairs of additional e-fuses and probe pads. As a result, for a given interposer with a fixed number of functional interconnects, a small number of test paths would require fewer e-fuses and probe pads. Therefore, our objective can be viewed as minimizing the total number of test paths.

In order to minimize the number of test paths, each test path should include as many functional interconnects as possible. However, the functional interconnects may not be randomly connected to form a single long test path because the e-fuse locations are restricted to be within a single die footprint. The following theorem provides a necessary condition for a single test path to include all functional interconnects.

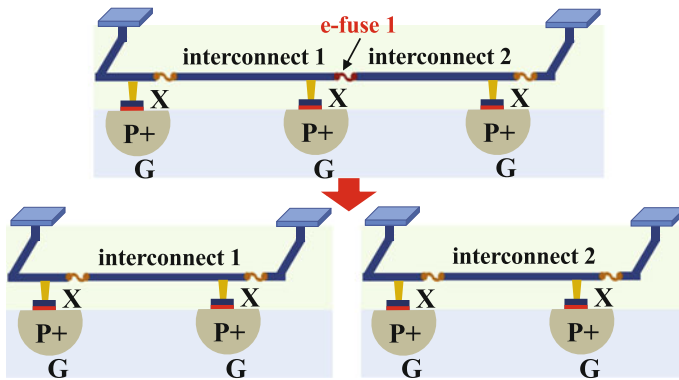


Fig. 2.9 A single test path broken into two test paths

Theorem 1 *If all functional interconnects can be included in a single test path, the number of input interconnects must be equal to the number of output interconnects for each die footprint, with the exception of the starting and ending die footprints.*

Proof A die footprint may appear multiple times in the test path. Each appearance of a die footprint is referred to as a node in the test path. Each node in the test path, except the starting and ending nodes, must have one input interconnect and one output interconnect. If this is not true (i.e., the number of input interconnects is not equal to the number of output interconnects), there must be a node in the test path that has only an input interconnect or only an output interconnect. In this case, all functional interconnects cannot be included in one test path.

Note that Theorem 1 does not provide a sufficient condition. In other words, even if the number of input interconnects is equal to the number of output interconnects for each die footprint (except for the starting and ending footprints), it is still possible that a single test path cannot be formed that will include all functional interconnects. Note that we are dealing with directed graphs in this chapter. It is well-known that a directed graph has an Euler path if and only if it is connected and all but two vertices have the same in-degree and out-degree; moreover, one of those two vertices has out-degree that is one greater than its in-degree (this is the start vertex of the Euler path), and the other vertex has in-degree that is one greater than its out-degree (this is the end vertex) [15]. Therefore, if the start die footprint has two more input interconnects than output interconnects, the problem can no longer be handled in terms of the Euler-path problem. In addition, it is unlikely that a commercial interposer will satisfy the assumption, i.e., the number of input interconnects is typically not equal to the number of output interposers for a single die footprint. Thus, a single test path is unlikely to include all interconnects.

In this situation, a test path may not necessarily include all functional interconnects. Therefore, the objective of minimizing the number of test paths is equivalent to the problem of searching for test paths until all functional interconnects are included. However, it is time-consuming to directly search for the longest test path.

For example, consider an interposer with four die footprints. The potential long test paths are shown in Fig. 2.10a. The test path begins at Die 1 footprint, and then, it can pass through any of Die 2, 3, or 4 footprints as long as there is an interconnect that connects the two. If the second node in the test path is Die 3 footprint, it can then pass through any of Die 1, 2, or 4 footprints for the third node. The only constraint on the test path is that no two consecutive nodes can be the same. Note that a test path can start from any die footprint on the interposer. Therefore, the number of possible test paths can be as large as $M \cdot (M - 1)^n$, where M is the number of die footprints and n is the number of interconnects between two die footprints. For large values of n , the number of possible test paths becomes extremely large and it becomes difficult to find the longest test path.

We next show that the test-path optimization problem is computationally intractable. As a result, efficient heuristics are needed to solve this problem for practical scenarios. The proof of intractability provides justification for the use of heuristic algorithms for this problem.

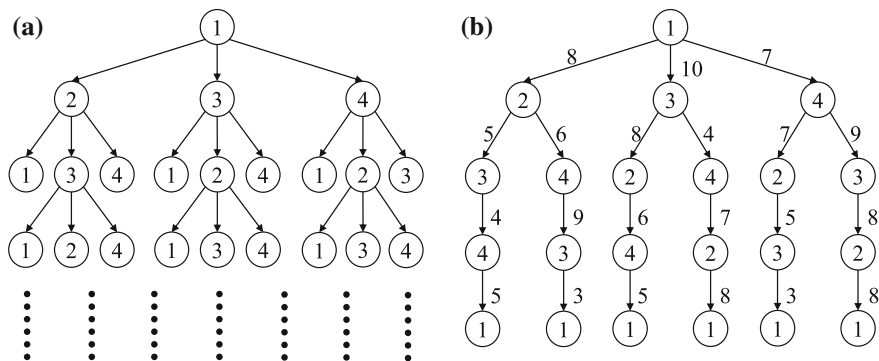
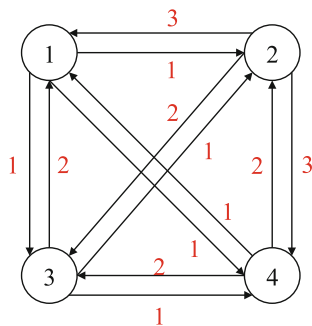


Fig. 2.10 **a** All possible test paths for an interposer example, **b** illustration of the initialized tree

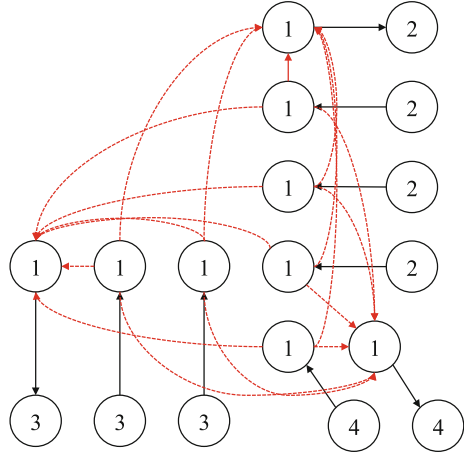
Fig. 2.11 Connection diagram for an example interposer



Theorem 2 *The test-path optimization problem is NP-hard.*

Proof Without loss of generality, we take an interposer with four die footprints as an example. The connection diagram is shown in Fig. 2.11. During the search for test paths, a test path may go through any given die footprint multiple times. Therefore, we attempt to expand one die footprint into multiple vertices. For instance, there are nine interconnects to and from Die 1 footprint. Hence, Die 1 footprint is expanded into 9 vertices, as shown in Fig. 2.12. The other die footprints are expanded in the same way and only the vertices connected to the Die 1 footprint are shown in Fig. 2.12. Since a test path is composed of functional interconnects and e-fuses, two types of edges are considered for the graph in Fig. 2.12. The first type of edges corresponds to functional interconnects, shown as dark lines in Fig. 2.12. Each edge represents one functional interconnect to or from Die 1 footprint. The second type of edges corresponds to e-fuses, shown as dash lines in Fig. 2.12. In order to search for the test paths, all possible e-fuses are added to the graph. Every Die 1 vertex that has an output interconnect must be connected to all Die 1 vertices that have an input interconnect. Figure 2.12 shows the expansion results for one die footprint.

Fig. 2.12 Expansion of Die 1 footprint



In a realistic graph, all die footprints must be expanded in the same way. Therefore, the search for the longest test path can be stated in terms of the following decision problem **P**:

INSTANCE: A directed graph G with n vertices.

QUESTION: Does there exist a path in G of length k ($k \leq n$) such that each vertex on the path is visited exactly once?

The parameter k refers to the length of the test path and the goal of the optimization problem corresponding to **P** is to maximize k . It is trivial to prove that **P** \in NP; we can (nondeterministically) “guess” an ordering of vertices of length k and verify in polynomial time that this ordering corresponding to a valid path in G . Likewise, we can (nondeterministically) “guess” a path in G and verify in polynomial time that this path is of length k .

We next consider a restriction of **P** whereby $k = n$. This special case of **P** is now equivalent to the well-known Hamiltonian path problem [16], which is known to be NP-complete. Therefore, we can conclude that **P** is NP-complete.

If a path as described above exists, then it can be chosen as the longest test path. Otherwise, we delete vertices in a systematic manner (e.g., one at a time) from the graph and consider subgraphs. We continue this procedure until we obtain a path that visits each vertex in the subgraph exactly once. As a result, our optimization problem is equivalent to the systematic (and repeated) invocation of **P** with $k = n$, $n-1$, $n-2$, and so on.

A heuristic solution is therefore needed to solve this problem in an efficient manner. In order to efficiently search for the longest test path, we propose a test-path design algorithm. The input to this algorithm is the interconnect matrix $W = [w_{ij}]$, where element w_{ij} represents the number of interconnects from Die i footprint to Die j footprint.

2.3.2 Proposed Algorithm

The first step in the proposed algorithm is to specify the constraint that each test path can pass through any given die footprint only once. Then, test paths that satisfy this constraint are selected. Afterward, the constraint is relaxed and the selected test paths are combined to form the “real” long test paths. When none of the remaining test paths can be combined, the algorithm terminates.

Under the specified constraint, a longest test path should begin at one die footprint, pass through all of the remaining die footprints, and finally return to the first die footprint. All possible test paths can be represented by a tree, where the nodes represent die footprints, and the edges represent the interconnects between different die footprints. Therefore, a tree can be created for any given die and interconnect layout. In such a tree, a longest test path is represented by a path from the root to a leaf of the tree. Because the majority of the test paths are likely to begin and end at the die footprint with the largest number of interconnects, both the root and the leaves of the initialized tree represent particular die footprint. The number of interconnects between different die footprints is defined as the *width* of the corresponding edge in the tree. The width of a particular test path is defined as the minimum width for all edges included in the path. An example of an interposer with four die footprints is used to illustrate the proposed algorithm. The tree for this example is shown in Fig. 2.10b. Die 1 footprint is taken as the root because it has the largest number of interconnects. The number within each node represents the index of the die footprint. The arrow illustrates the direction of a single test path. The test paths are selected using the following steps.

(1) Select test paths:

In view of weighted critical area, it is not necessary to test inner-bridge faults in nondense test paths so that programming time is reduced. As a result, during test path selection, our first priority is to select nondense test paths. When we consider multiple nondense test paths, the second priority is to select the longest test path. If two test paths have the same length, the third priority is to select the path with the largest width. After a test path is selected, the interconnects included in that test path are not considered in the subsequent steps; they are eliminated from W . After W is updated, we continue to select test paths until there are no more paths whose width is greater than 0 from the root of the tree to the leaves.

In Fig. 2.10b, all the root-to-leaf paths have the same length and the widest test path is $1 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$ with width 7. Therefore, seven test paths in that order are selected. Meanwhile, the interconnect matrix W is updated based on the path selection, which is indicated by the red path in Fig. 2.13a. As a result, all edges related to the four included interconnects are updated. Then, we continue to select the test paths: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1$ (width 4); $1 \rightarrow 2 \rightarrow 4 \rightarrow 3 \rightarrow 1$ (width 2); $1 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 1$ (width 1). The updated tree is shown in Fig. 2.13b.

(2) Remove all 0 edges:

As shown in Fig. 2.13b, many edges in the tree have a width of 0. This type of edge prevents us from selecting new test paths from those interconnects, and all 0 edges

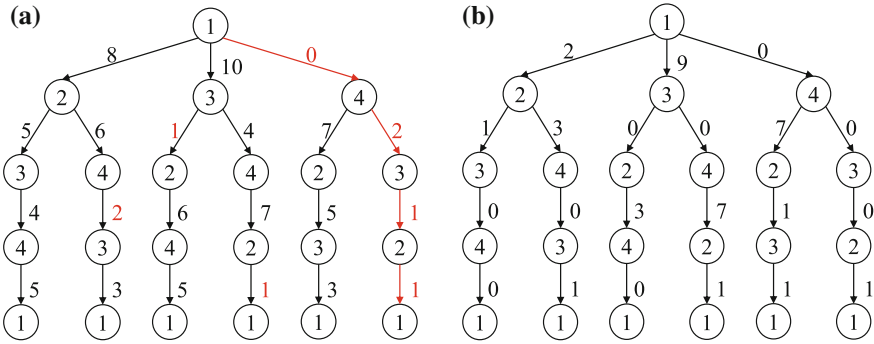


Fig. 2.13 Tree after the selection of **a** the first test path, and **b** all test paths

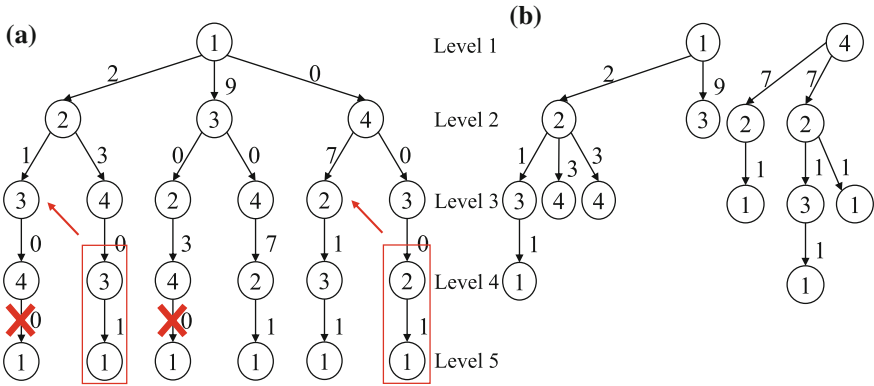


Fig. 2.14 **a** Operations for three types of edges, **b** tree after removing all edges with zero width

should therefore be removed from the tree. The nodes of the tree can be divided into several levels according to their positions: The root is at the lowest level (level 1), and all leaves are at the highest level. The tree edges can be classified into three types, and the 0 edges are removed at the highest levels before we move to the lower levels. In Fig. 2.13b, the edges between level 4 and level 5 are first pruned.

The first type of edge is an edge of zero width. This type of edge can be removed from the tree. For the given example, $4 \rightarrow 1$ can be removed, as indicated in Fig. 2.14a. The second type is an edge whose width is nonzero and parent edges¹ do not have zero width. This type of edge is kept in the tree without any changes. In the given example, $2 \rightarrow 1$ (parent edge $4 \rightarrow 2$) and $3 \rightarrow 1$ (parent edge $2 \rightarrow 3$) belong to this type.

The third type is an edge that has nonzero width, but it has a parent edge with zero width. This type of edge is first disconnected from its parent edges and then moved

¹A parent edge is an edge that is one level higher in the tree and shares a node with the edge of interest.

one level up. Next, the nodes at the end of this edge are merged with their parents' sibling nodes. In the given example, $3 \rightarrow 1$ (parent edge $4 \rightarrow 3$) and $2 \rightarrow 1$ (parent edge $3 \rightarrow 2$) belong to this third type. Consider the path $4 \rightarrow 3 \rightarrow 1$ in Fig. 2.14a. The edge $3 \rightarrow 1$ is disconnected from node 4 and moved one level up; node 3 is merged with the sibling of node 4: a different node 3 that is on level 3.

The same operations are applied to the edges between other levels. When the edges between level 2 and level 3 are updated and an edge belongs to the third type, a node on level 2 cannot be merged with its parent's sibling because the root does not have siblings. In this situation, this node and its corresponding descendants are detached from the original tree and they will form a new tree. The updated tree for the given example with all edges of width zero removed is shown in Fig. 2.14b.

(3) Merge redundant edges:

Once all-0 edges are removed from the tree, some edges in the tree may be redundant due to the multiple merging operations in the previous steps. For the example shown in Fig. 2.14b, node 2 on the left has two "node 4"s as its children, and node 4 on the right has two "node 2"s as children. These redundant edges must be merged; the final tree is shown in Fig. 2.15.

Once the redundant edges have been combined, the initialized tree is updated and one or more trees are derived. Then, the same steps are applied to each tree: select the test paths, remove all 0 edges, and merge the redundant edges. Meanwhile, the functional interconnects selected in test paths are continuously omitted from the interconnect matrix. Once the interconnect matrix is null, we can conclude that all functional interconnects have been included in test paths.

(4) Combine test paths:

Every test path requires two additional probe pads; however, probe pads introduce overhead. In order to minimize the number of probe pads, the number of test paths must be further reduced. Therefore, test paths obtained in the previous steps should be combined in order to form longer test paths.

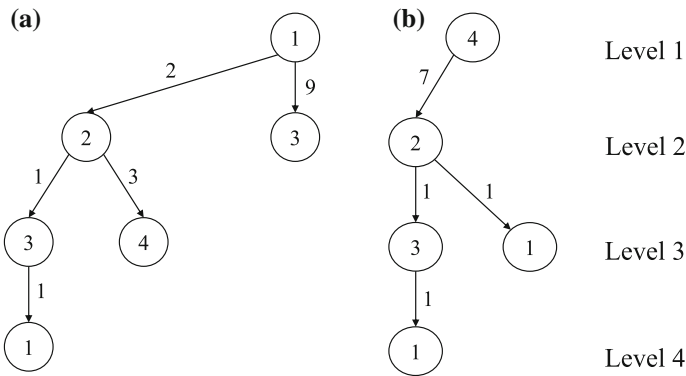


Fig. 2.15 Tree after merging redundant edges

Each test path has four parameters: start node, end node, length, and width. A set of test paths is selected through the previous three steps, and this set is referred to as a path set. The sum of all widths of the test paths in the path set is referred to as the total width. This number is equal to two times the number of probe pads required for the interposer.

When we combine the test paths, the shortest test path is first chosen from the path set. If there are several shortest test paths, the widest of the shortest test paths is chosen. This test path is referred to as a *base path*. Note that not all test paths in the path set can be combined with the base path; only the test paths that share the same start or end node with the base path can be combined. From these test paths, the shortest one is chosen as another base path, and it is combined with the first base path. Then, a new test path is derived and added to the path set. The four parameters of the new test path are generated based on the two base paths. The path set is updated and the same procedures are applied to it until the total width of the path set is minimized.

For each test path set, if its width is greater than one, this means multiple test paths share the same direction. As a result, in order to further reduce the total number of test paths, these multiple test paths can also be combined. This method is referred to as self-combination. For instance, a test path is finalized as $1 \rightarrow 2 \rightarrow 3$ with the width of two. Without self-combination, six e-fuses and four probe pads are required for the final test paths. However, if these two test paths are self-combined together as $2 \rightarrow 1 \rightarrow 3 \rightarrow 3 \rightarrow 1 \rightarrow 2$, the number of required probe pads and e-fuses can be reduced to two and five, respectively. In this way, the test-path design is accomplished and the total number of test paths is minimized.

(5) Interconnect selection for test paths:

Once the test paths are generated, how to select functional interconnects for each test path remains undiscussed. In Fig. 2.16, five functional interconnects and two test paths are shown in an interposer example: (1) $1 \rightarrow 2 \rightarrow 3 \rightarrow 2 \rightarrow 1$ and (2) $2 \rightarrow 3$. There are a total of twelve options for interconnect selection. For instance, both $FI1 \rightarrow FI3 \rightarrow FI4 \rightarrow FI2$ and $FI2 \rightarrow FI3 \rightarrow FI5 \rightarrow FI1$ (FI is short for functional interconnect) can form Test Path 1. Since multiple options are available for each test path, a method for interconnect selection is proposed for all test paths.

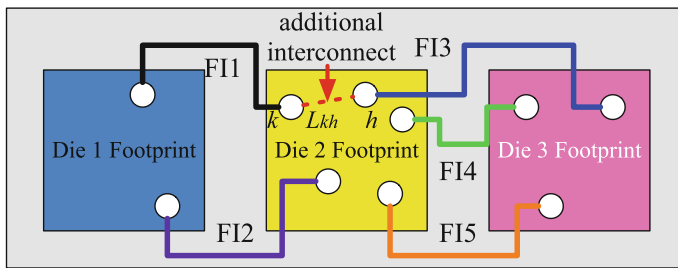


Fig. 2.16 Illustration of a test path example

The selection criterion is based on the distance between two consecutive interconnects in a test path. If the distance is shorter, it is much easier for e-fuse to connect those two interconnects because of the shorter length of additional interconnects. Since two interconnects are connected by e-fuse and additional interconnects within one die footprint, the interconnect selection problem can be solved within each die footprint.

The test-path design problem can be formally defined as follows. For each target die footprint, suppose there are a total number of P ports. The distance between any two ports is represented by L_{kh} , where k and h range from 1 to P . Whether port k is connected to die i , footprint is represented by a binary parameter C_{ik} . The physical information of the interposer is provided by the foundry; it contains all the interconnect location information. As a result, the values of L_{kh} and C_{ik} can be easily derived from the interconnect location information. Once the test paths are generated, the width of a test path that starts at die i footprint, goes through the target die footprint, and ends at die j footprint is represented by T_{ij} . Our goal is to select interconnects for each test path such that the total length of additional interconnects is minimized in the target die footprint. In order to obtain optimal results, we use integer linear programming (ILP) to solve this problem.

A binary variable x_{kh} is defined. The variable x_{kh} is equal to 1 if port k and port h are connected by e-fuse and additional interconnects, $1 \leq k, h \leq P$. Otherwise, if there is no e-fuse between port k and port h , then x_{kh} is equal to 0. Constraints on variable x_{kh} are defined as follows:

$$x_{kk} = 0, \forall k; x_{kh} = x_{hk}, \forall k, h \quad (2.1)$$

$$\sum_{k=1}^P \sum_{h=1}^P x_{kh} \cdot C_{ik} \cdot C_{jh} = T_{ij}, \forall i, j \quad (2.2)$$

The two sets of constraints in (2.1) indicate the self-characteristics of x_{kh} . Constraints in (2.2) indicate the location and number of e-fuses in the target die footprint should be based on the results of test-path design algorithm. With the constraints defined above, the total length of additional interconnects within the target die footprint is defined as follows:

$$\sum_{k=1}^P \sum_{h=1}^P x_{kh} \cdot L_{kh}$$

After the interconnect selection method is applied to each die footprint, the total length of additional interconnects for the entire 2.5D IC is minimized.

2.4 Experimental Results

In this section, we present simulation results and an evaluation of the test-path design method with benefits of weighted critical area. The simulations are carried out using HSPICE. The test-path design algorithm is implemented using Perl. Experiments are carried out on a Linux workstation with an Intel Xeon 2.53 GHz CPU and 64 GB memory. The technology parameters (Table 2.1) reflect the state of the art of commercial interposer technology.

2.4.1 Testing the Horizontal Interconnects

The circuit model of Fig. 2.17a is simulated in order to analyze the effectiveness of the e-fuses. Two 1700- μm -length horizontal interconnects are connected by an e-fuse and a sequence of signals is applied to N1, shown as V(N1) in Fig. 2.17b. Before the e-fuse is programmed, its resistance is as small as 25 Ω . Therefore, the e-fuse can be viewed as a short circuit and Interconnect 1 and Interconnect 2 are connected together. As a result, the signals applied to N1 are transmitted through the interconnects and are observed at N4, shown as V(N4-np) in Fig. 2.17b. After the e-fuse is programmed, its resistance is as large as 4 G Ω . Therefore, the e-fuse can be viewed as an open circuit and Interconnect 1 is separated from Interconnect 2. As a result, no signals are transmitted to N4 and N4 is dangling, shown as V(N4-p) in Fig. 2.17b.

Another set of simulations is carried out to verify that the programmed e-fuse does not affect the functionality of the interposer. Specifically, signals from N1 are applied to Interconnect 1 and signals from N3 are applied to Interconnect 2. With the programmed e-fuse, the responses from N1 are observed at N2, shown as V(N2) in Fig. 2.17b, and the responses of N3 are observed at N4, shown as V(N4-p1) in Fig. 2.17b. It can be seen that the two interconnects can operate independently. Therefore, the programmed e-fuse does not affect the normal functionality of the interposer.

We next simulate the test procedure for inner-bridge faults. The schematic in Fig. 2.18a is simulated. In the simulated condition, e-fuse *a* has already been programmed. Point X is grounded and point Z is isolated from the ground. Figure 2.18b

Table 2.1 Technology parameters used for simulation

TSV	Diameter	Height	Pitch	t_{ox}
	10 μm	100 μm	100 μm	230 nm
	Width	Thickness	Resistance	Capacitance
M1	45 nm	105 nm	9.435 $\Omega/\mu\text{m}$	0.2173 fF/ μm
M2	45 nm	100 nm	9.467 $\Omega/\mu\text{m}$	0.2035 fF/ μm

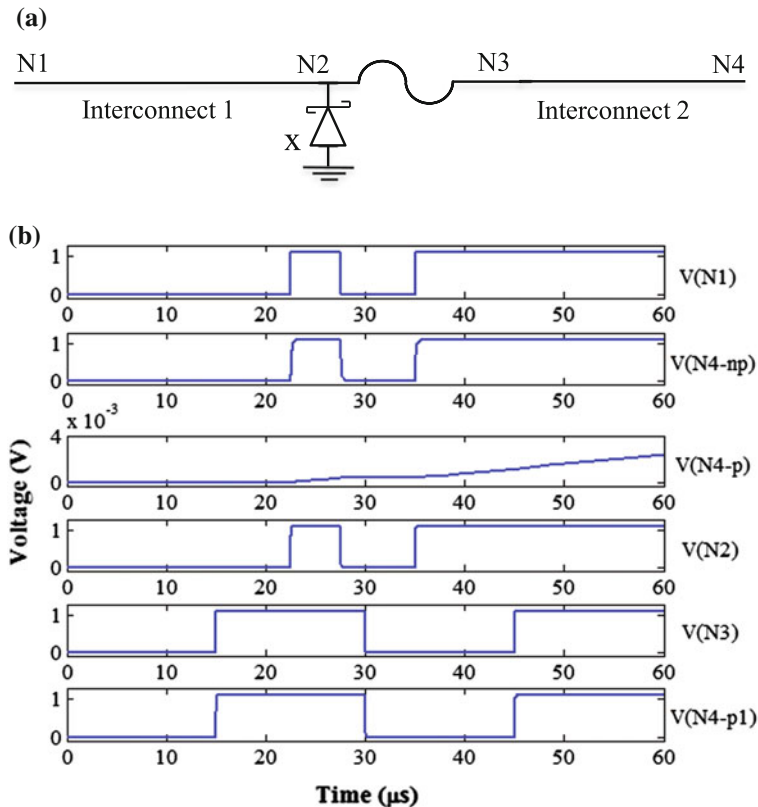


Fig. 2.17 Horizontal interconnects with an e-fuse: **a** circuit model, **b** simulation results

shows the responses at the observation point N1, which is one end of interconnect 3. Meanwhile, a logic 1 is applied to B until 45 μs . If there is no inner-bridge fault between Interconnect 2 and the remaining interconnects, the responses at N1 are shown as $V(N1\text{-T})$ in Fig. 2.18b. In region 1, the N1 voltage is slightly lower than 1.1 V because the circuit is shorted to ground through point X. Then, the e-fuse b is programmed in region 2. In this region, since a discharge path is temporarily formed through point Y, N1 voltage decreases slightly. After e-fuse b is completely programmed, the circuit is open at position b . Since there is no path between “interconnect 3—e-fuse c —interconnect 4” circuitry and ground, N1 voltage increases to 1.1 V in region 3. At 45 μs , no signal is applied to B, and B is dangling. Then, N1 voltage decreases slowly in region 4 due to discharge through capacitors.

If there is an inner-bridge fault between interconnect 2 and the remaining interconnects, the responses at N1 are shown as $V(N1\text{-F})$ in Fig. 2.18b. In this situation, regardless of whether e-fuse b is programmed, there is always a path between N1 and ground. Therefore, once B is dangling, N1 voltage drops to zero quickly. Based

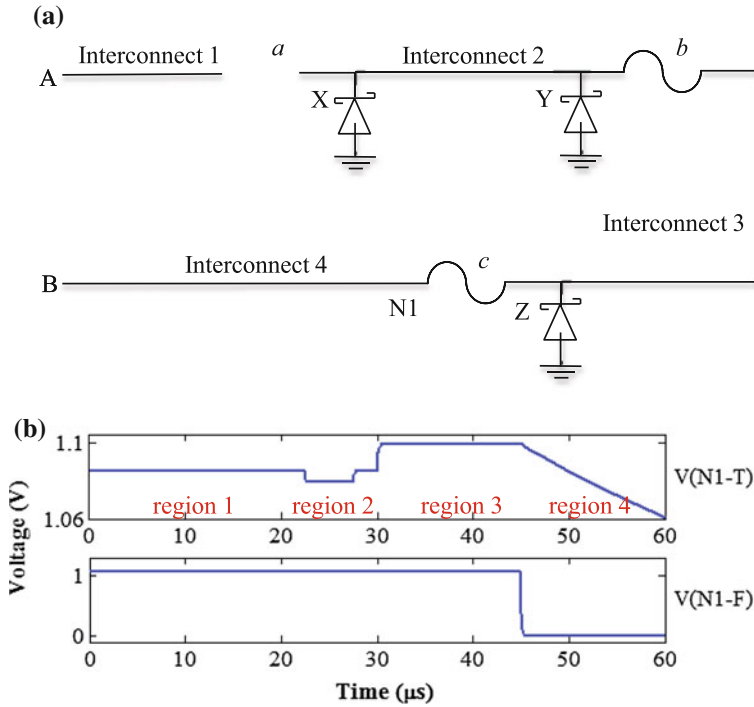


Fig. 2.18 Inner-bridge fault testing: **a** circuit model, **b** simulation results

on the difference between $V(N1-T)$ and $V(N1-F)$, it can be seen that the proposed test procedure for inner-bridge faults is effective.

2.4.2 Testing for Vertical Interconnects

The circuit model in Fig. 2.19a, which has two vertical interconnects connected by an e-fuse, is simulated. Each vertical interconnect is composed of TSV, M1, and V1 vias. The length of M1 is 4.5 μm ; the via V1 is typically 45 nm in diameter and 80 nm in height. Each M1 is connected to 6 V1s in parallel. In addition, each vertical interconnect is connected to an input or output driver through a horizontal interconnect. A sequence of signals are applied to N2, shown as $V(N2)$ in Fig. 2.20. Before the e-fuse is programmed, TSV 1 and TSV 2 are connected by the e-fuse. Therefore, signals applied to N2 are transmitted through the TSVs and the e-fuse, and are observed as N5, shown as $V(N5\text{-np})$ in Fig. 2.20. After the e-fuse is programmed, TSV 1 is separated from TSV 2. Therefore, no signals are transmitted to N5, and it is dangling, shown as $V(N5\text{-p})$ in Fig. 2.20.

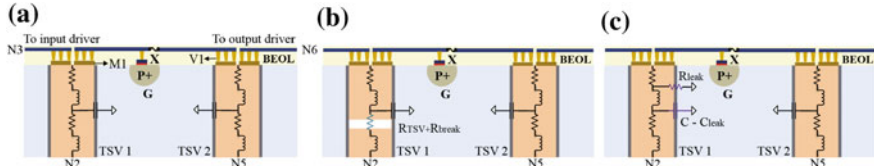


Fig. 2.19 Circuit models: **a** fault-free, **b** break faults, and **c** pinhole faults

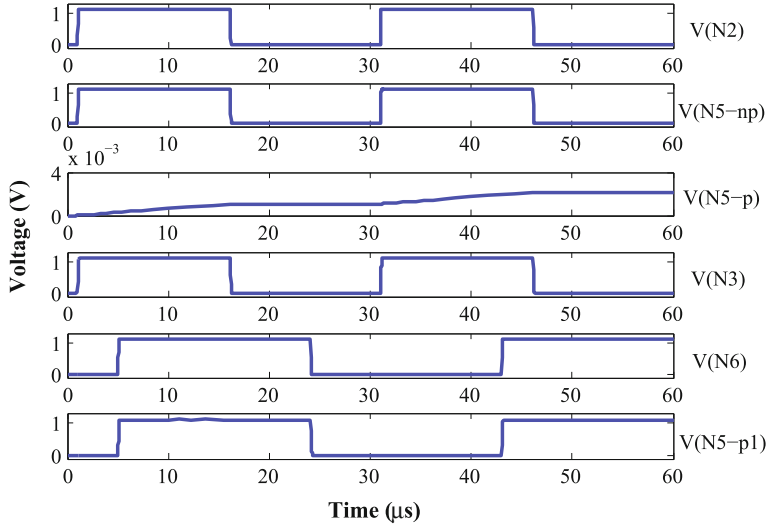


Fig. 2.20 Simulation results for vertical interconnects with e-fuses

Once dies are stacked on the interposer, the input driver (N3) will receive signals from TSV 1, and the output driver (N6) will send signals to TSV 2. With this in mind, another simulation is conducted. Signals from N2 are applied to TSV 1, and signals from N6 are applied to TSV 2. After the e-fuse is programmed, the responses to N2 are observed at N3, shown as $V(N3)$ in Fig. 2.20; the responses to N6 are observed at N5, shown as $V(N5-p1)$ in Fig. 2.20. It can be seen that the two vertical interconnects work independently as desired. Therefore, the programmed e-fuse does not affect the normal functionality of the interposer.

The three TSV fault models are next analyzed. The break fault models a full-open defect, as shown in Fig. 2.19b, and the TSV resistance is modeled as extra resistance, which can be as large as 1 G Ω . The void fault is modeled in a manner similar to the break fault. Instead of imposing the significant resistance change as used in the break model, the resistance increase in the void model is caused by a reduction of the effective conducting area. As a result, it is significantly smaller than the output resistance of a typical driving gate [17]. Therefore, the void faults can be neglected in interposer pre-bond testing.

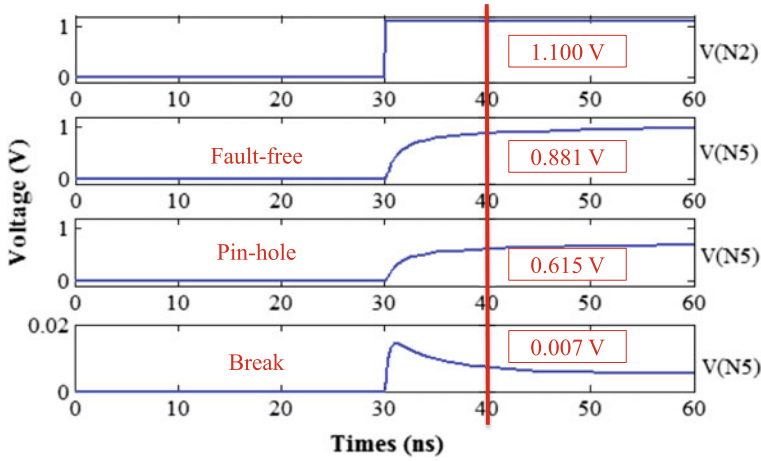


Fig. 2.21 Simulation results for the testing of TSV defects

The pinhole fault is modeled as leakage, as shown in Fig. 2.19c. A leakage resistance R_{leak} is placed in parallel with the TSV capacitance. Meanwhile, the TSV capacitance is also affected by the leakage, resulting in a decreased capacitance. The simulation results for the fault-free and faulty cases are shown in Fig. 2.21. A low to high transition is applied to N2, and the responses are observed at N5. At 40 ns, the fault-free response is 0.881 V. The responses of the pinhole and break faults are 0.615 V and 0.007 V, respectively. Since they are significantly different from the fault-free value, these two faults can be easily detected.

2.4.3 Evaluation of the Test-Path Design Method

In order to evaluate the proposed optimization technique, a small problem instance with a simple 3×3 interconnect matrix is first utilized to compare with an exhaustive-enumeration solution. There are a total of 7 interconnects from Die 3 footprint to Die 2 footprint. Any two other die footprints are connected by 5 interconnects. With the proposed technique, the test paths can be organized as “12 – 23 – 31 – 13 – 32 – 21” with width 5 and “32” with width 2, which is the same as that obtained by exhaustive enumeration. However, the computation (CPU) time for the heuristic is significantly lower (0.1 s vs. 241 s). This difference is more evident when we consider larger problem instance. We could not solve the 4×4 problem instance using exhaustive enumeration because we ran out of memory.

The successful integration of up to four dies on a passive interposer has been reported for a 2.5D IC [18, 19], and the stacking of even larger numbers of dies on interposers has also been discussed [20, 21]. A total of eight dies on an interposer has been reported in [22]. Several interconnect matrices are generated assuming differ-

ent number of die footprints on the interposer, which reflect recent and forthcoming 2.5D IC designs. Then, the experimental results for an interposer design provided by GLOBALFOUNDRIES are also presented. Based on the data from GLOBALFOUNDRIES, the time from applying a test pattern to observing the test response (t_p) is 25 μ s; the time to program e-fuse (t_e) is 5-10 μ s. In this chapter, t_e is set to be 7.5 μ s. Since e-fuses involved in the inner-bridge testing are programmed one by one while the other e-fuses can be programmed at the same time, the testing time can be estimated as $(t_p + t_e) \cdot n$, where n is the number of e-fuses in the longest test path.

(1) Simulation Results for Test-Path Design:

Several interconnect matrices are first analyzed when the number of die footprints is varied from three to eight. The matrix density is set to 80%: 80% elements in the matrix are nonzero elements. Different from the matrix density, which represents the percentage of nonzero elements, the area density is defined as the percentage of nonzero elements in the interconnect matrix that pass through weighted critical areas. For example, if 10% of the elements in the interconnect matrix pass through one or more weighted critical areas, the area density of the matrix is 10%. The area density is set to 30% for these matrices. The simulation results are shown in Fig. 2.22, where Test1 to Test6 represents the number of die footprints ranging from three to eight. The number of test paths in nWCA is smaller than the corresponding values for WCA because constraints on test-path design are less in nWCA and more functional interconnects can be included in a single test path. Since all test paths must be tested for inner-bridge faults in nWCA, its testing time is higher than those in WCA. The number of e-fuses remains the same for both WCA and nWCA.

The influence of the matrix density on the test-path design is next analyzed. An 8×8 interconnect matrix with 30% area density is first considered. The matrix density is varied from 30% to 70%, and the test-path design results are shown in Fig. 2.23. No matter the matrix density, WCA always has significant influence on the number of test paths and testing time, while it has limited influence on the number of e-fuses.

The influence of the area density on the test-path design is also analyzed. The matrix density is set to 100% and the area density is varied from 10% to 100%. The test-path design results are shown in Fig. 2.24. It can be seen that WCA can help to reduce the testing time, though WCA could lead to a relative larger number of test paths. In particular, when the area density is small (e.g., 10% 30%), WCA can significantly reduce the testing time.

The computation time is also analyzed as the area density and matrix density are varied from 10% to 100%, respectively. The simulation results are shown in Fig. 2.25. It can be seen that the area density has limited influence on the CPU time. In contrast, the matrix density has a significant influence on the CPU time. In particular, the CPU time increases with an increase of the matrix density. This is because more functional interconnects are operated in the test-path design algorithm.

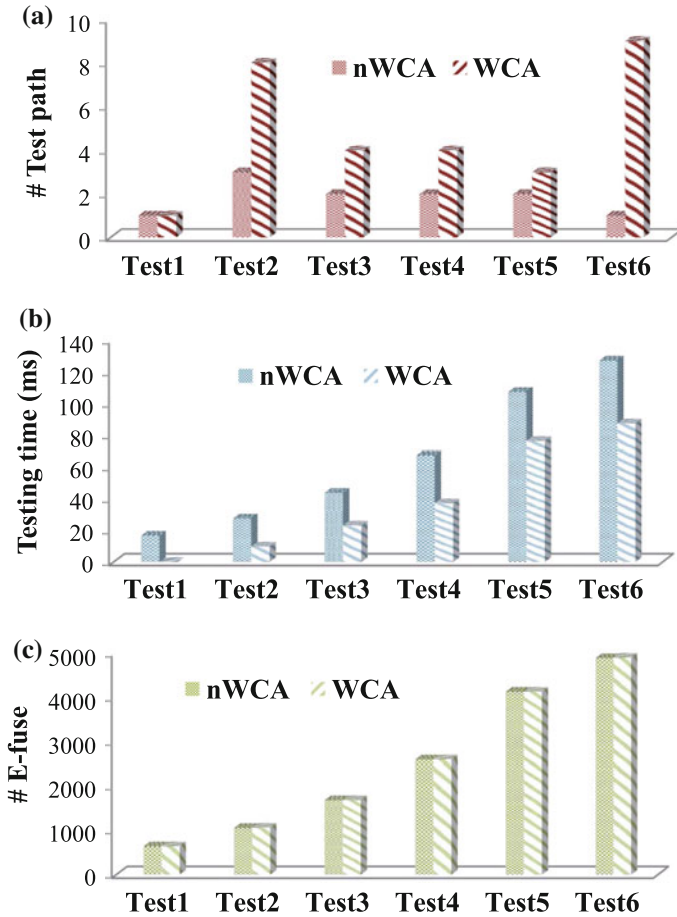


Fig. 2.22 Simulation results for interposers with different number of die footprints

(2) Experimental Results for Test-Path Design:

We have analyzed a commercial interposer from GLOBALFOUNDRIES; this interposer is referred to as X5. Five dies are stacked on the interposer: one ASIC die and four high-band-memory (HBM) dies. Interconnects are between the ASIC die and each HBM die.

The influence of weighted critical area is analyzed, where division resolution (n) is varied from 2 to 10. Therefore, the interposer is divided from 2×2 to 10×10 subareas. Then, the pre-bond testing is implemented on them. The experimental results are shown in Fig. 2.26. The number of probe pads for WCA is larger than those for nWCA when n is small. With the increase of n , the number of probe pad finally reach the same value for WCA and nWCA. In Fig. 2.26b, WCA can

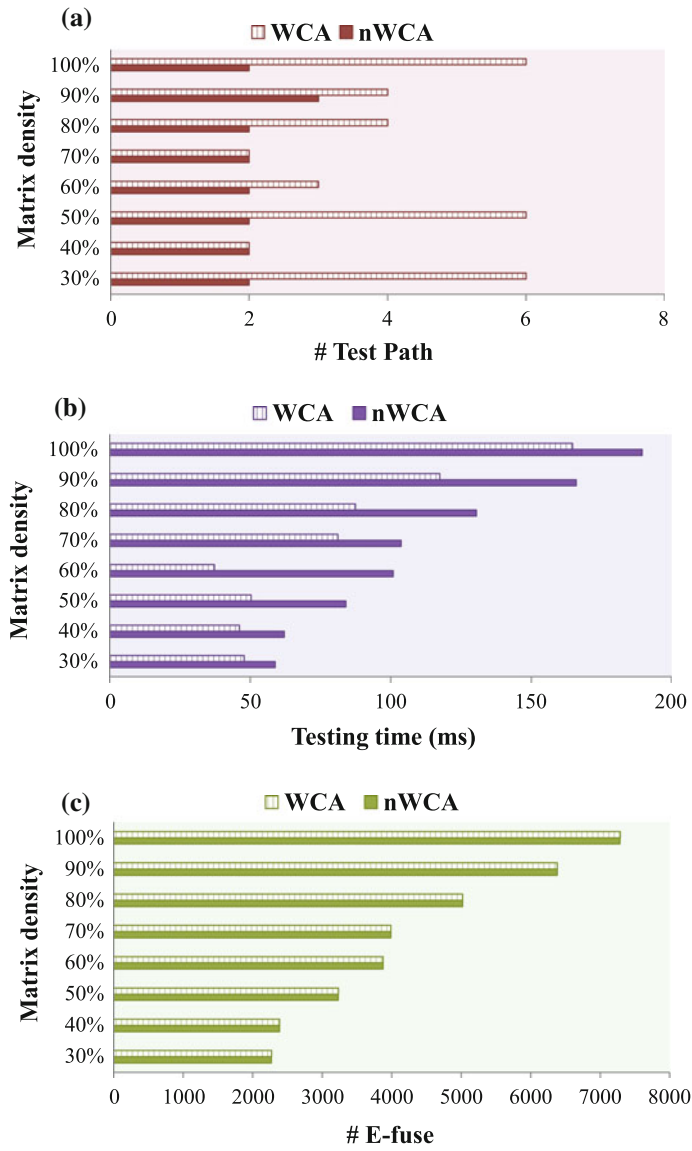


Fig. 2.23 Simulation results with different matrix density

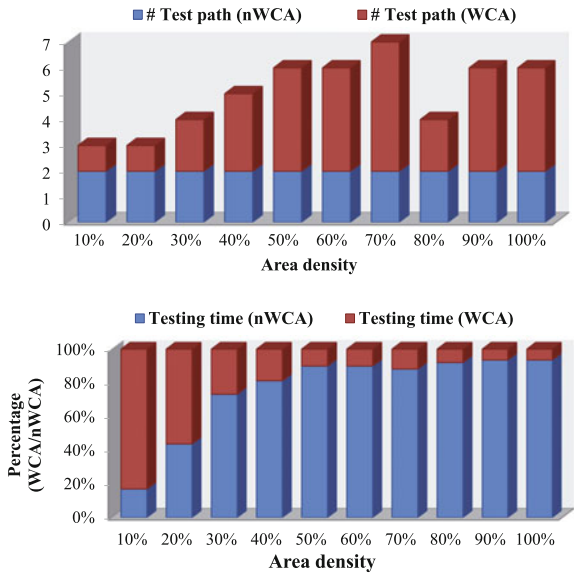


Fig. 2.24 Simulation results with different area density

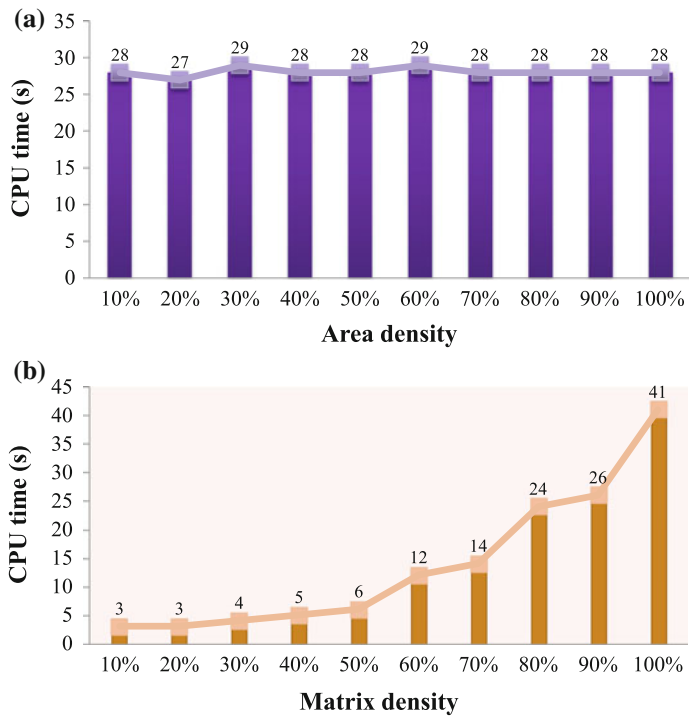


Fig. 2.25 CPU time with different area density and matrix density

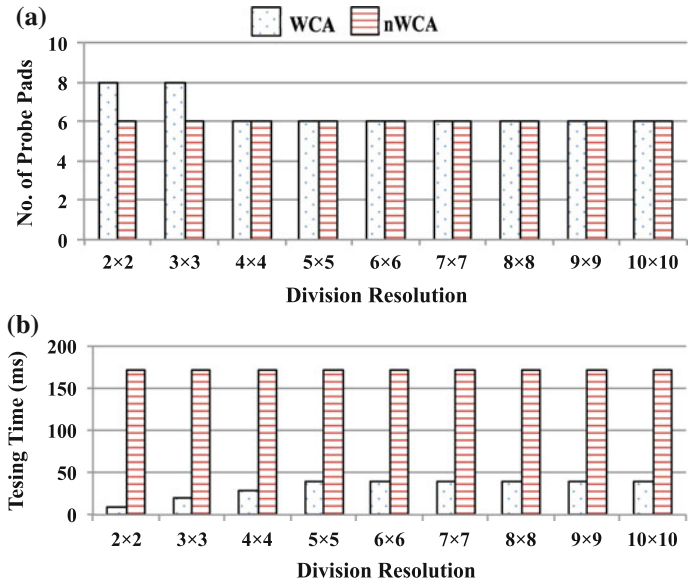
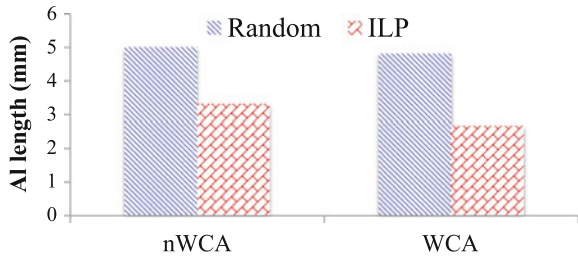


Fig. 2.26 Experimental results with different division resolutions

Fig. 2.27 Experimental results with interconnect selection method



significantly reduce the testing time, which is at most 20% of the testing time in nWCA. Therefore, it is advisable to include weighted critical area for pre-bond testing.

Finally, the interconnect selection method (ILP) is applied to X5. Since the ILP method is applied within each die footprint, it will not introduce more e-fuses or probe pads. The selection results are presented in Fig. 2.27, where “AI length” represents “additional interconnect length to place e-fuses.” In Fig. 2.27, a random selection method (Random) is taken as a baseline and both WCA and nWCA conditions are considered. It can be seen that the ILP method can always outperforms the random method in both WCA and nWCA conditions. Therefore, the simulation results conclusively demonstrate the benefits of the ILP method.

2.5 Conclusion

We have tackled one of the most challenging problems in 2.5D IC test and introduced a new test architecture that allows pre-bond interposer testing for 2.5D ICs. When the interposer is under test, e-fuses are used to connect separated interconnects so that test paths can be formed to test both horizontal and vertical interconnects. After testing and interposer qualification, the e-fuses are programmed to disconnect the interconnects so that the functionality of the interposer will not be affected. The concept of die footprint is utilized for interconnect testing, and the overall assembly and test flow has been described. In order to reduce test time, the concept of weighted critical area has been defined and utilized. In addition, a test-path design algorithm is proposed that minimizes the number of test paths. We have presented HSPICE simulation results to demonstrate the effectiveness of the pre-bond test solution. The benefit of using the weighted critical area has been demonstrated using a commercial interposer as a realistic test case. Finally, we also provide the evaluation of test-path design to demonstrate the effectiveness of the proposed approach.

References

1. S.K. Goel, S. Adham, M.-J. Wang, J.-J. Chen, T.-C. Huang, A. Mehta, F. Lee, V. Chickermane, B. Keller, T. Valind, S. Mukherjee, N. Sood, J. Cho, H. Lee, J. Choi, S. Kim, Test and debug strategy for TSMC CoWoSTM stacking process based heterogeneous 3D IC: a silicon case study, in *IEEE International Test Conference*, 2013
2. C. Kothandaraman, S.K. Iyer, S.S. Iyer, Electrically programmable fuse (eFUSE) using electromigration in silicides. *IEEE Electron Device Lett.* **23**(9), 523–525 (2002)
3. H. Suto, S. Mori, M. Kanno, N. Nagashima, Systematic study of the dopant-dependent properties of electrically programmable fuses with silicided poly-si links through a series of I-V measurements. *IEEE Trans. Device Mater. Reliab.* **7**(2), 285–297 (2007). June
4. C. Kothandaraman, S.K. Iyer, S.S. Iyer, Electrically programmable Fuse (eFUSE) using electromigration in silicides. *IEEE Electron Device Lett.* **23**(9), 523–525 (2002). Sept
5. T. Ueda, H. Takaoka, M. Hamada, Y. Kobayashi, A. Ono, A novel Cu electrical fuse structure and blowing scheme utilizing crack-assisted mode for 90–45 nm-node and beyond, in *Symposium on VLSI Technology Digest of Technical Papers*, 2006
6. H. Takaoka, T. Ueda, H. Tsuda, A. Ono, A novel via-fuse technology featuring highly stable blow operation with large on-off ratio for 32 nm node and beyond, in *IEEE International Electron Devices Meeting*, 2007
7. J. Ryckaert, E.J. Marinissen, D. Linten, Two-step interconnect testing of semiconductor dies, 2014. US Patent App. 14/247,019
8. K. Kumagai, Y. Yoneda, H. Izumino, H. Shimojo, M. Sunohara, T. Kurihara, M. Higashi, Y. Mabuchi, A silicon interposer BGA package with Cu-filled TSV and multi-layer Cu-plating interconnect, in *IEEE Electronic Components and Technology Conference*, 2008, pp. 571–576
9. F. Cros, L. Namburi, T. Hu, Fine pitch probes for semiconductor testing and a method to fabricate and assemble same. US Patent 9000793, 2015
10. M.A. Christo, J.A. Maldonado, R.D. Weekly, T. Zhou, Silicon interposer testing for three dimensional chip stack. US Patent 7863106, 2011
11. K.S.-M. Li, S.-J. Wang, J.-L. Wu, C.-Y. Ho, Y. Ho, R.-T. Gu, B.-C. Cheng, Optimized pre-bond test methodology for silicon interposer testing, in *IEEE Asian Test Symposium (ATS)*, 2014, pp. 13–18

12. S. Kannan, R. Agarwal, A. Bousquet, G. Aluri, H.-S. Chang, Device performance analysis on 20 nm technology thin wafers in a 3D package, in *IEEE International Reliability Physics Symposium*, 2015
13. P.T. Wagner, Interconnect testing with boundary scan, in *IEEE International Test Conference*, 1987, pp. 52–57
14. L.-R. Huang, S.-Y. Huang, S. Sunter, K.-H. Tsai, W.-T. Cheng, Oscillation-based prebond TSV test. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **32**(9), 1440–1444 (2013)
15. D.B. West et al., *Introduction to Graph Theory*, vol. 2 (Prentice Hall, Upper Saddle River, 2001)
16. M.R. Garey, D.S. Johnson, L. Stockmeyer, Some simplified NP-complete graph problems. *Theoret. Comput. Sci.* **1**(3), 237–267 (1976)
17. R. de Orio, H. Ceric, S. Selberherr, Electromigration failure in a copper dual-damascene structure with a through silicon via. *Microelectron. Reliab.* **52**(9), 1981–1986 (2012)
18. M. Sunohara, T. Tokunaga, T. Kurihara, M. Higashi, Silicon interposer with TSVs (Through Silicon Vias) and fine multilayer wiring, in *IEEE Electronic Components and Technology Conference*, 2008, pp. 847–852
19. B. Banijamali, S. Ramalingam, K. Nagarajan, R. Chaware, Advanced reliability study of TSV interposers and interconnects for the 28 nm technology FPGA, in *IEEE Electronic Components and Technology Conference*, 2011, pp. 285–290
20. H.H. Jones, Technical viability of stacked silicon interconnect technology. Xilinx. White Paper, <http://www.xilinx.com/publications/technology/stacked-siliconinterconnect-technology-ibs-research.pdf>, 2010
21. P. Dorsey, Xilinx stacked silicon interconnect technology delivers breakthrough FPGA capacity, bandwidth, and power efficiency. White Paper, http://www.xilinx.com/support/documentation/whitepapers/wp380_Stacked_Silicon_Interconnect_Technology.pdf, 2010
22. C.-C. Chi, E.J. Marinissen, S.K. Goel, C.-W. Wu, Post-bond Testing of 2.5D-SICs and 3D-SICs containing a passive silicon interposer base, in *IEEE International Test Conference*, 2011

Testing of Interposer-Based 2.5D Integrated Circuits

Wang, R.; Chakrabarty, K.

2017, XIV, 182 p. 118 illus., 102 illus. in color.,

Hardcover

ISBN: 978-3-319-54713-8