IPSJ Transactions on System LSI Design Methodology Vol.7 56–73 (Aug. 2014)

[DOI: 10.2197/ipsjtsldm.7.56]

Invited Paper

# Test and Design-for-Testability Solutions for 3D Integrated Circuits

Krishnendu Chakrabarty[1,a)] Mukesh Agrawal[1] Sergej Deutsch[1] Brandon Noia[1] Ran Wang[1] Fangming Ye[1]

Received: January 25, 2014, Released: August 4, 2014

**Abstract:** Despite the promise and benefits offered by 3D integration, testing remains a major obstacle that hinders its widespread adoption. Test techniques and design-for-testability (DfT) solutions for 3D ICs are now being studied in the research community, and experts in industry have identified a number of hard problems related to the lack of probe access for wafers, test access in stacked dies, yield enhancement, and new defects arising from unique processing steps. We describe a number of testing and DfT challenges, and present some of the solutions being advocated for these challenges. Techniques highlighted in this paper include: (i) pre-bond testing of TSVs and die logic, including probing and non-invasive test using DfT; (ii) post-bond testing and DfT innovations related to the optimization of die wrappers, test scheduling, and access to dies and inter-die interconnects; (iii) interconnect testing in interposer-based 2.5D ICs; (iv) fault diagnosis and TSV repair; (v) cost modeling and test-flow selection.

**Keywords:** cost modeling, repair, retiming, through-silicon via (TSV), wafer sort

## 1. Introduction

To overcome the challenges introduced by technology scaling, in particular long interconnects, the semiconductor industry has recently started investigating 3D stacked ICs (3D ICs). By designing circuits with more than one active device layer, large 2D circuits can instead be created as 3D circuits with significantly shorter interconnects. This not only leads to large reductions in latency and power consumption, but also higher bandwidth and circuits with a higher packing density and smaller footprint. Since dies in a 3D stack can be manufactured separately, there are also benefits to the heterogeneous integration of different technologies into a single 3D stack.

In order to make 3D ICs commercially viable, new test solutions are required to keep costs low. Compared to the testing of 2D ICs, 3D ICs introduce many new challenges for testing. Yield loss for each die in a 3D IC is compounded during stacking, so stacking of untested die leads to prohibitively low product yields. This motivates the need for pre-bond testing, or the testing of dies prior to being bonded to a 3D stack. Pre-bond testing allows for the stacking of dies that are known to be defect-free and it also enables die-matching, so that dies in the same stack can be chosen based on metrics such as speed or power consumption. It is also important to perform post-bond tests, or testing of either a partial stack to which all dies have yet to be bonded, or testing of the complete stack. Post-bond testing ensures that the stack is functioning as intended and that no defects have been introduced during the processing steps of thinning, alignment, and bonding.

Since die stacking can be performed in different ways, including wafer-to-wafer stacking, there is also potential to optimize the yield by wafer matching. Moreover, test insertions during stacking of dies can be included to improve the yield. However, these tests introduce extra cost, hence cost modeling is necessary to optimize a 3D test flow based on various parameters such as costs of test insertions, as well as die and stacking yields. Finally, 3D stacking introduces a new level of design hierarchy in addition to that in conventional SoCs, hence new test architectures are required to provide test access to each die in a stack, as well as new methods to optimize 3D test schedule.

The remainder of this paper is organized as follows. In Section II, we review pre-bond TSV testing and repair techniques. Section III focuses on improving stacking yield by wafer matching. Section IV provides an overview of 3D design-for-test and test-schedule optimization techniques. In Section V, we present cost models to optimize a 3D test flow. Finally, Section VI concludes the paper.

## 2. Pre-bond TSV Testing and Repair

TSV testing can be separated into two distinct categories: pre-bond and post-bond test [1], [2]. Pre-bond testing allows for the detection of defects that are inherent in the manufacture of the TSV itself, such as impurities or voids, while post-bond testing detects defects caused by thinning, alignment, and bonding. Successful pre-bond defect screening can allow defective dies to be discarded before stacking. Because methods to “unbond” die are yet to be realized, even one faulty die can cause a stacked IC to be considered *bad*, including all good dies in the stack.

TSVs play the role of interconnects, hence there are a number of pre-bond defects that can impact chip functionality [3]. These

1 Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA

a) krish@duke.edu

include pinhole defects in the TSV sidewall insulator that create leakage paths to the substrate and microvoids or cracks in the TSV pillar that result in resistive-open faults. This section covers a variety of techniques that are utilized for the detection and repair of TSV faults prior to die stacking.

Pre-bond TSV tests aim to detect manufacturing faults in TSVs, usually through parametric measurement of one or more TSV electrical properties. **Figure 1** (a) shows a defect-free TSV modeled as a lumped RC circuit, similar to a wire. The resistance and capacitance of the TSV depends on the geometry and material of the TSV pillar and sidewall insulator. Defects in the TSV pillar or the sidewall insulator alter the electrical characteristics of the TSV. Figure 1 (b) shows the effect of a microvoid in the TSV pillar. These microvoids increase the resistance of the TSV pillar and, depending on the severity of the defect, can manifest as anything from a small-delay defect to a resistive open. Figure 1 (c) shows a pinhole defect of the sidewall insulator. Depending on the severity of the defect, the leakage of the TSV may significantly increase due to the leakage path to the substrate.

### 2.1 BIST Techniques for TSV Fault Detection

Pre-bond TSV testing is difficult due to a variety of factors. Pre-bond test access is severely limited due to TSV pitch and density. Current probe technology using cantilever or vertical probes requires a minimum pitch of 35 $\mu$m, but TSVs have pitches of 4.4 $\mu$m and spacings of 0.5 $\mu$m [5]. Without the introduction of large probe pads onto a TSV or similar landing pads on the face side of the die [6], current probe technology cannot make contact with individual TSVs. Adding many landing pads is undesirable, as they significantly decrease the pitch and density of TSVs and TSV landing pads. Thus, the number of test I/O available to a tester during pre-bond test is significantly reduced compared to the I/O available during post-bond test. Furthermore, TSVs are single-ended before bonding, meaning that that only one side is connected to die logic and the other side is floating (or tied to the substrate in the case of a deposited TSV without bottom insulation). This complicates TSV testing because standard functional and structural test techniques, for example stuck-at and delay testing, cannot be performed on the TSV.

Due to these difficulties, a number of innovative BIST techniques have been introduced in the literature to perform pre-bond TSV test [3], [4], [7], [8]. BIST techniques require only a few external signals to perform test, including, but not limited to power/ground, clock, and BIST enable contacts. Thus, BIST techniques are well suited for the test-access constraints of pre-bond test. Furthermore, they do not require expensive testers or probe cards to interface with the circuit and provide test data or analyze test responses.

An example of a BIST technique for detecting TSV capacitance is shown in **Fig. 2** [8]. This BIST method treats TSVs like DRAM cells, using read and write operations relying on charging and discharging the TSV capacitance. A test is performed by first precharging each TSV to $V_{DD}$. Then, TSVs are isolated during a hold phase. Next, a charge-sharing circuit is created between a TSV and a charge-sharing capacitance $C_{load}$. The voltage after charge sharing is compared to reference voltages via sense amplification. In the circuit of Fig. 2, transmission gates are used to select each TSV under test (TUT) as well to bias $V_{sense}$ to $V_b$ during hold operations. A tri-state buffer is used for the write driver to write to each TSV cell. A sense amplifier is used to compare the voltage after charge sharing to $V_{ref}$, where $V_{ref}$ is either $V_{RL}$ or $V_{RH}$.

BIST techniques such as the DRAM-like method described above suffer from drawbacks. No current BIST architecture for pre-bond TSV test is capable of detecting resistive defects near or at the end of the TSV furthest from the active device layer. This is due to the fact that these resistive defects result in no significant increase or decrease to the TSV capacitance because the bulk of the TSV closest to the test architecture is intact. Furthermore, BIST techniques do not provide an avenue for TSV burn-in tests, and so cannot screen for TSVs that would experience infant mortality failures. The test circuits utilized by BIST techniques, such as voltage dividers or sense amplifiers, cannot be calibrated before hand and are subject to process variation on the die. This can impact the accuracy of parametric tests.

### 2.2 Pre-Bond Parametric TSV Test Through Probing

TSV probing offers an alternative to BIST techniques for pre-bond TSV testing. Novel probe technologies have been introduced in recent years to enable pre-bond probing. Cascade Microtech Inc. has introduced a pyramid probe card that has been demonstrated at a 40 $\mu$m array pitch [9]. Form Factor Inc. has introduced the NanoPierce$^{TM}$ contact head for 3D TSV probing, similarly demonstrated at a 40 $\mu$m array pitch [10]. The probe needles are grown from many dense nanofibers that act together to make a contact. Both probe cards utilize low-force probing and show minimal microbump damage.

Despite these new advances in probe card technology, the demonstrated pitches and array placement of needles limit TSV placement and density if individual contact with each TSV required. Furthermore, scaling these probe cards is yet to be demonstrated, and it appears that micro-bumps may scale faster

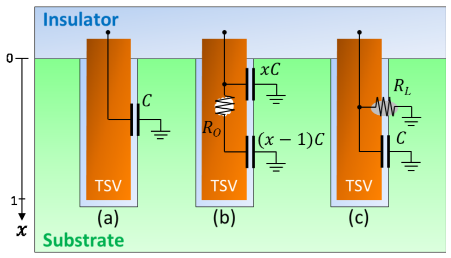


**Fig. 1** TSV models: (a) fault-free, (b) micro-void, (c) pinhole [4].

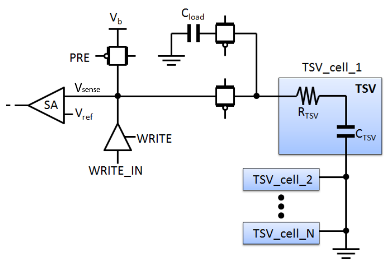


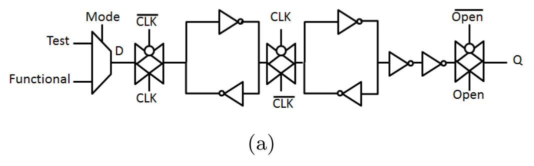
**Fig. 2** Detailed test module circuit for DRAM-like blind TSV testing [8].

than probe technology. For example, in Ref. [11] micro-bumps have already been manufactured at sizes of 5 $\mu$m with a 10 $\mu$m pitch. Furthermore, even if every TSV on a die can be contacted individually, the issue of routing test data to and from a probe card with the thousands of probe needles required to contact every TSV is likely to be prohibitive. If the number of probe needles is instead reduced to a more manageable number, then many touchdowns may be needed to test every TSV on a die, significantly increasing test time and the likelihood of damaging the thinned die or wafer.

This section presents a new technique for pre-bond TSV testing that is compatible with current probe technology and leverages the on-die scan architecture that is used for post-bond testing. It utilizes many single probe needle tips, each to make contact with multiple TSVs, shorting them together to form a single "TSV network." The proposed approach highlights the relevance of today's emerging test standards and test equipment, and the important role that tester companies can play in 3D IC testing. Because the proposed method requires probing, it is assumed in this chapter that the die has already been thinned and that it is supported by a rigid platter (carrier) to prevent mechanical damage during probing. During test, the probe needle must be moved once to allow for testing of all TSVs in the chip under test. This method also allows for the concurrent testing of many TSVs to reduce overall test time. Furthermore, significantly fewer probe needles are required to test all TSVs, which reduces the cost and complexity of probe equipment.

For post-bond external tests, a 1500-style die wrapper with scan-based TSV tests has been proposed [12] that utilizes wrapper boundary scan flops between TSVs and die logic. To enable pre-bond TSV probing, the standard scan flops that make up the die boundary registers between die logic and TSVs are modified to be gated scan flops (GSFs), as shown in **Fig. 3**. The gated scan flop accepts either a functional input or a test input from the scan chain; the selection is made depending on operational mode. A new signal, namely the "open signal," is added; it determines whether the output Q floats or takes the value stored in the flip-flop. In the GSF design, shown at gate-level in Fig. 3 (a) and at transistor-level in Fig. 3 (b), two cross-coupled inverters are used to store data. Transmission gates are inserted between the cross-coupled inverters and at the input (D) and output (Q) of the flop itself. An internal inverter buffer is added before the output transmission gate such that the gated scan flop can drive a large capacitance on its output net without altering the value held in the flop. The "open" signal controls the final transmission gate.

To determine the capacitance of the TSV network and the resistance of each TSV, the probe needle must be equipped with an active driver and a method of detection. In order to keep this circuitry simple, a design such as the one shown in **Fig. 4** can be used. This design consists of a DC source with a voltage on the order of the circuit under test. A switch, $S2$, is used to connect or disconnect the source from a capacitor ($C_{charge}$) of known capacitance. The voltage across the capacitor, $V1$ is continuously monitored through a voltmeter. A second switch, $S1$, allows the capacitor to be connected or disconnected from the probe needle itself.



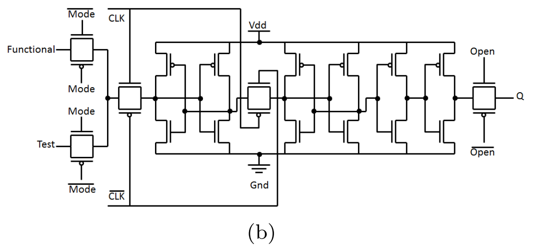


**Fig. 3** Design of a gated scan flop: (a) gate-level and (b) transistor-level [13].

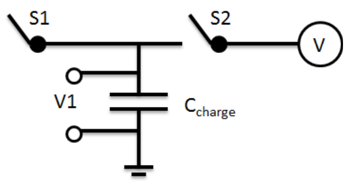
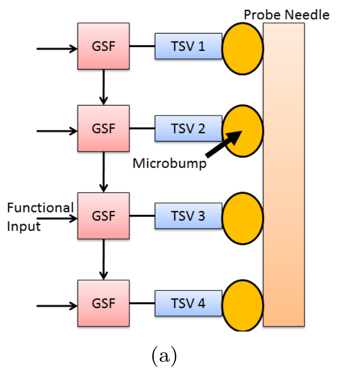


**Fig. 4** A charge sharing circuit [13].



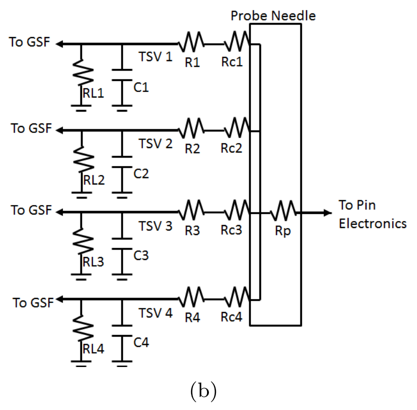


**Fig. 5** TSV network model [13].

A probe needle makes contact with a number of TSVs at a time, as seen in **Fig. 5** (a). The TSVs are connected to gated scan-flops, which are connected to form a scan chain. This circuit is modeled as seen in Fig. 5 (b). The probe needle has a known resistance $R_p$ and a contact resistance ($Rc_1$-$Rc_4$) with each TSV. The contact resistance depends on the force with which the probe needle contacts each TSV, and may differ per TSV. Each TSV has an associated resistance ($R_1$-$R_4$) and capacitance ($C_1$-$C_4$). Furthermore, a leakage path modeled by a resistance ($RL_1$-$RL_4$) exists between each TSV and the substrate. The net capacitance $C_{net}$ is the combined capacitance of all of the TSVs in parallel.

Parametric tests, including capacitance, leakage, and resistance tests, can be performed for the TSVs in a network. The bulk of TSV defects that can be tested pre-bond result in increased TSV resistance. For this reason, it is important that a pre-bond test is capable of accurately measuring TSV resistance. To measure resistance for each TSV, the capacitor $C_{charge}$ will be charged through each TSV, and the time needed to charge the capacitor to a chosen voltage (for example, 99% of $V_{dd}$) is recorded. Long charge times increase the resolution in resistance measurement, but they lead to longer test time. As a tradeoff, smaller voltage levels (such as 90% of $V_{dd}$) can be used to reduce test times if the resolution is acceptable. The test begins by loading all of the gated scan-flops with 1 and discharging the TSV network using the probe. Switch $S2$ is opened and switch $S1$ is closed such that the capacitor $C_{charge}$ is discharged as well. One of the gated scan-flops is then set to its low-impedance output, allowing the scan-flop to charge $C_{charge}$ through its connected TSV. When $V1$ reaches the pre-determined voltage, the time to charge $C_{charge}$ is recorded. It is then compared to a calibrated charge curve for a non-faulty TSV. This process of charging and discharging continues for each TSV, which can be completed quickly by incrementing the controlling counter to open each subsequent TSV.

The robustness of TSV resistance measurements in a 20-TSV network under process variations is shown to be accurate [13]. The TSV under test is considered to have a resistive fault with a total resistance of 50 Ω. Resistances on the other TSVs in the network are simulated with a Gaussian distribution in which 3-$\sigma$ is a 20% spread from the nominal value of 1 Ω. All TSV capacitances are simulated with a similar Gaussian distribution using a nominal value of 20 fF, and leakage resistances are distributed around a nominal 1.2 TΩ. Charge times are then compared to a calibrated curve. As can be seen from a 100-trial Monte Carlo simulation in **Fig. 6**, the resolution of resistance measurements remains high under process variations, with a mean measurement of 51.2 Ω and a standard deviation of 6.6 Ω. Similarly accurate results are shown for capacitance and leakage tests as well.

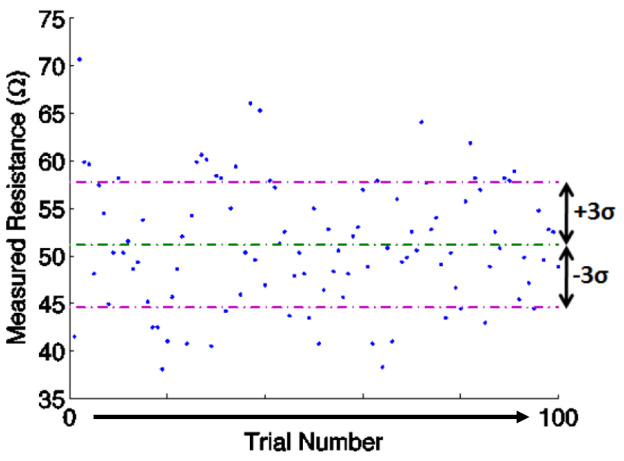


**Fig. 6** 100-point Monte Carlo simulation of TSV resistance measurements with 20% variation in the TSV resistance, leakage resistance, and capacitance of fault-free TSVs [13].

## 2.3 Ring Oscillators for TSV Test and Diagnosis

In the previous section, we outlined a pre-bond TSV-test method using TSV probing. However, despite recently reported success in mechanical probing at fine array pitches [9], it still remains to be seen how easily they can be used in practice. As an alternative solution, a non-invasive pre-bond TSV test method has been proposed [4]. In this work, two types of faults are targeted: resistive-open faults and leakage faults. For each case, as well as the fault-free case, an electrical TSV model is used as shown in Fig. 1. Deviations in TSV parameters due to defects lead to variations in the propagation delay of the net connected to TSV. These variations are measured by ring oscillators (ROs) at different voltage levels. A method that uses ROs for TSV test was proposed in Ref. [14]; it was originally designed for post-bond TSV diagnosis. The technique presented below, however, focuses exclusively on pre-bond TSV test. An RO for pre-bond test is created by reconfiguring $N$ I/O segments as shown in **Fig. 7**. Each I/O segment includes a TSV and a bidirectional I/O cell connected to the front side of the TSV. The I/O cells are assumed to be part of the functional circuitry, which is common in industrial designs.

The number of TSVs in a group ($N$) can be selected based on the desired oscillation frequency. In the extreme case, if $N = 1$, the RO contains only a couple of gates, which results in a relatively short oscillation period (or high frequency). Such an oscillation frequency might be too high to drive the on-chip measurement logic. By appending extra segments, we increase the delay and thus reduce the oscillation frequency, relaxing the speed requirement on the measurement circuitry. In addition, all TSVs in the same group can share the same counter without extra decode logic, since all of them are in the oscillator loop. This reduces wiring and the amount of DfT logic. The signal TE (test enable) controls the multiplexers selecting between the functional outputs coming from the internal logic and the oscillator loop. If TE = 0, the multiplexers 1 select the functional outputs coming from the internal logic, enabling a functional mode. If TE = 1, the circuit is configured in an oscillator loop and the functional driver output enable signal OE is overridden, activating a TSV test mode. The signals BY[1] . . . BY[N] (Bypass) control the multiplexers that include or exclude a TSV from the oscillator loop. OE (output enable) controls the tri-state drivers of the I/O cells. In functional mode, this signal is set by the internal logic. In test mode, OE is set to 1 to enable the drivers.

The approach was verified by simulations with HSPICE. For the simulations, the TSV models as described in Fig. 1 and the 45 nm Predictive Technology Model (PTM) low-power CMOS models [15] were used. BUF_X4 buffers from the Nangate 45 nm

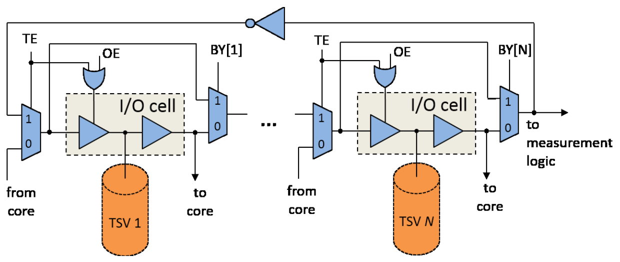


**Fig. 7** Ring oscillator with $N$ TSVs [4].

Open Cell Library [16] served as TSV drivers. For other gates, X1 versions were used.

First, a resistive-open fault in TSV at the location $x = 0.5$ is simulated using transient analysis to record the oscillation period of the ring oscillator $T$. In the first run, the faulty TSV is enabled (BY[1] = 0) and all other TSVs are bypassed (BY[2...N] = 1). In the second run, all TSVs are disabled. Subsequently, the oscillation period of the second run $T_2$ is subtracted from that of the first run $T_1$ for each value of $R_O$: $\Delta T = T_1 - T_2$. During an actual test, the values $T_1$ and $T_2$ will be measured by the on-chip DfT and the results sent to the test equipment and post-processed there. This subtraction step removes the propagation delay of the path through I/O cells $2 \ldots N$ and the inverter. The remainder is virtually the propagation delay due to the I/O cell and the TSV under test. This approach greatly reduces the effect of delay variations in gates and interconnects due to random process variations. The results of this simulation are shown in **Fig. 8**. As expected, an increase in the resistance $R_O$ leads to a reduction of the oscillation period. This indicates that we can detect resistive opens of a sufficient size by measuring the oscillation period. For instance, $\Delta T$ of a resistive defect of size 1 kΩ at $x = 0.5$ is reduced by 10% compared to the fault-free case that can be identified.

To see the effect of process variations and the effect of applying different supply voltages, a number of Monte Carlo (MC) simulations were run using the model described above extended by the following process-variation model: $3\sigma_{V_{th}} = 50\,\text{mV}$, $3\sigma_{L_{eff}} = 0.1L_{eff}$, where $\sigma_{V_{th}}$ is the variation in threshold voltage and $\sigma_{L_{eff}}$ is the variation in gate length. These data are consistent with the those reported by industry for recent technology nodes [17].

**Figure 9** shows the results of MC simulation for a fault-free TSV and a TSV with a resistive open defect of size 1 kΩ. The spread in the fault-free and faulty cases was analyzed at different voltage levels. At lower supply voltage levels, a part of data points from both fault-free and faulty cases overlap and thus become indistinguishable. Increasing the voltage reduces this overlap to a minimum until there is no aliasing. Even in presence of process variations, this TSV test method allows for detection of resistive-open defects that have a sufficiently large size and are located in the upper part of the TSV. The test resolution depends on the process variation: the more variation, the harder it becomes to distinguish small resistive opens from the fault-free case. In addition, higher supply voltage results in a better resolution: aliasing is reduced, allowing for detection of smaller resistive-open faults.

Leakage faults exhibit a different behavior than resistive open faults. To show this, the same simulation as described above ($N = 5$) was performed for a TSV with a leakage fault. **Figure 10** shows the dependence of $\Delta T$ on the leakage $R_L$ for different voltage levels.

According to the simulation results, leakage faults increase the oscillation period, which makes them distinguishable from the fault-free case as well as resistive open faults. Moreover, strong leakage faults below a certain threshold, $R_L \approx 1\,\text{k}\Omega$, prevent the circuit from oscillating. In other words, the TSV exhibits stuck-at-0 behavior. This threshold depends on the supply voltage: it drops as the voltage is increased. Another observation is that in the regions slightly above each threshold, $\Delta T$ is extremely sensitive to small variation in leakage. This indicates that leakage of a wide range can be easily detected if tested at different voltage levels. Strong leakage ($R_L$ low) will show up at higher $V_{DD}$. Weak leakage will become detectable at lower $V_{DD}$. This observation is consistent with the results of prior work on very-low-voltage tests for bridging faults [18], [19].

Next, we perform MC simulations to verify the robustness of our test method. **Figure 11** shows the results of MC simulations for a 3 kΩ leakage fault and the fault-free case at different voltage levels. We see that in the sensitive region right above the threshold (≈ 0.75 V), the data points for the two cases do not overlap. As we increase $V_{DD}$, the gap between them becomes smaller such that we cannot distinguish between the faulty and fault-free cases.

The simulation results confirm that resistive opens and leakage faults significantly change the oscillation period $T$ and hence they can be detected using the proposed method. Since resistive opens reduce $T$ and leakage faults increase $T$, these fault types are distinguishable from each other. The results also show that TSVs should be tested at multiple voltage levels in order to in-

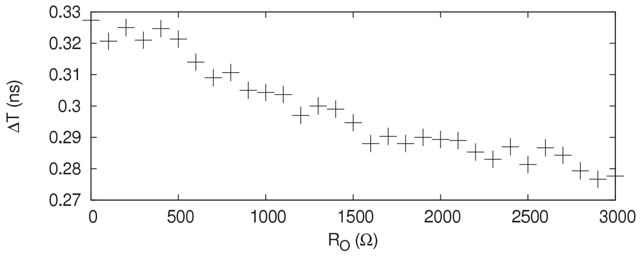


**Fig. 8** $\Delta T$ as a function of $R_O$ at location $x = 0.5$ and at 1.1 V supply voltage [4].

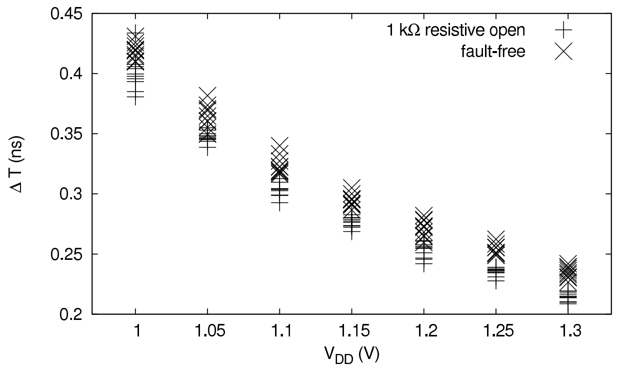


**Fig. 9** $\Delta T$ as a function of supply voltage in (a) fault-free case and (b) in case of 1 kΩ resistive open at $x = 0.5$ [4].

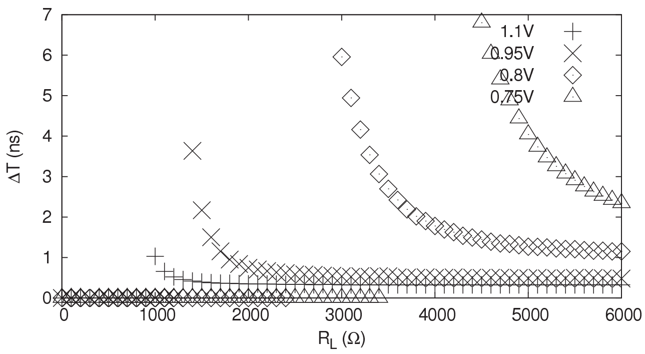


**Fig. 10** $\Delta T$ as a function of $R_L$ at different voltage levels [4].

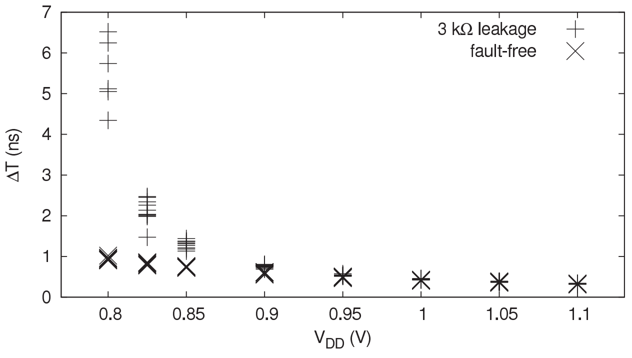


**Fig. 11** $\Delta T$ as a function of supply voltage in (a) fault-free case, and (b) in case of 3 kΩ leakage fault [4].

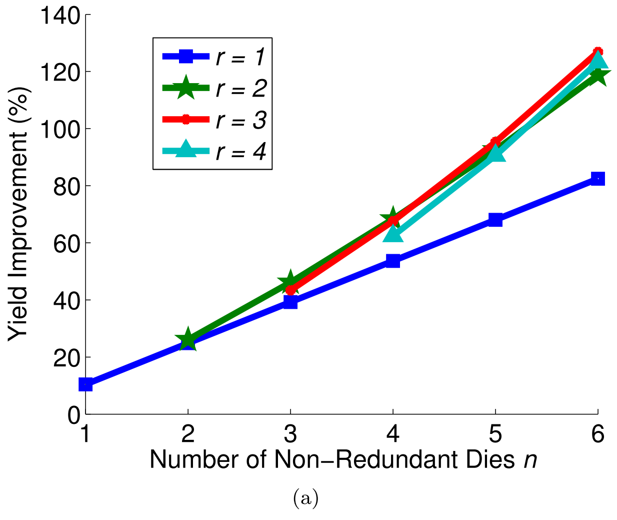
crease fault detectability.

## 2.4 Redundancy Optimization

Wafer matching and optimized placement and management of redundancy resources can significantly improve stack yields in 3D stacked memories. Differences between 2D and 3D memories with respect to repair are due to the types of 3D memory designs and the additional freedom in types of redundancy and placement of test circuits provided by 3D integration. Generally, there are three types of redundancy that are possible in a 3D memory architecture [20]:

- *Intra-Layer Redundancy* is most analogous to 2D redundancy architectures. In such a design, each memory die in the stack contains its own redundant resources that can be used to repair faulty memory cells on the die. Each memory die in the stack does not share its redundant resources with other dies in the stack.
- *Inter-Layer Redundancy* refers to a design where each die may or may not contain its own redundant resources. If a die does contain its own resources, then it shares these resources with other dies and has access to the resources on other dies. In this way, a die can utilize redundant resources on another die if it cannot repair all of its faulty memory cells with its own resources.
- *Layer Redundancy* refers to a repair architecture where redundant resources exist at the die or wafer level. In other words, any given 3D memory stack may have one or more dies devoted to redundant memory arrays that can be utilized for repair. If a memory die is unable to repair itself due to a large number of faulty memory cells, then the entire die must be replaced by a redundant memory die.

In Ref. [20], the authors calculate the yield and cost improvements of using layer redundancy in a stack of $n$ non-redundant layers and $r$ redundant layers, as shown in **Fig. 12**. What can be seen from Fig. 12 (a) is that layer redundancy improves stack yield for all scenarios with different values for $n$ and $r$. As the stack becomes larger, the yield improvement also increases significantly due to the increased likelihood of a faulty die existing in the stack. Examining Fig. 12 (b), it can be seen that layer redundancy is not cost effective for small stacks. For example, when $n = 1$ and $r = 1$, it is over twice the cost to produce a good stack with layer redundancy than it is to produce a good stack without layer redundancy. This is because adding one redundant layer



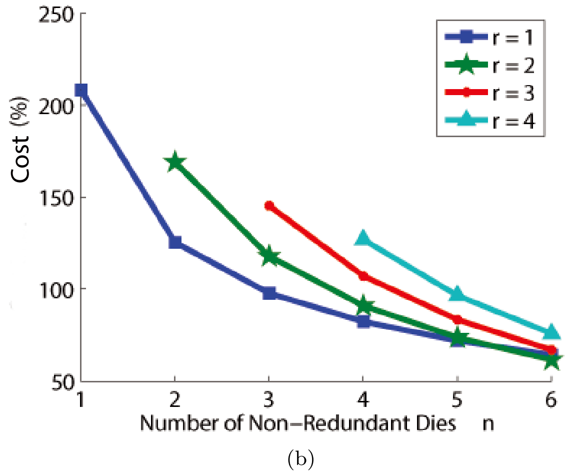


**Fig. 12** Yield improvement (a) and cost improvement (b) over the base case of no layer redundancy for stacks of varying $n$ non-redundant layers and $r$ redundant layers [21].

doubles the cost of manufacturing the stack but does not have a significant impact on stack yield.

In Ref. [22], a repair architecture was utilized such that adjacent memory dies can share their spare resources between them, creating inter-layer redundancy. If the resources available on one layer are not sufficient to repair that layer, then unused spares from adjacent layers can be used as needed. Utilizing wafer matching to place memory dies with insufficient resources to repair themselves adjacent to dies with excess resources in a stack, significant improvements were achieved in stack yield. The authors found yield improvements of nearly 30% for $8 \times 8$ spares on each layer when redundant resources are shared between dies, especially when there are fewer spares on each die. This is because in an environment with fewer redundant resources on each die, there is a greater likelihood that a given die will be unable to completely repair itself using only its own resources. As the number of spares increase, the likelihood of a die being capable of repairing all of its faulty rows and columns with its own resources increases, reducing the yield improvement to be gained by redundancy sharing.

## 3. Wafer Sort and Wafer Matching for Yield Improvement

In any 3D manufacturing and test flow, it is necessary to determine which tests should be performed and when, as well as as how stacking will take place in order to minimize cost. The stacking of dies to create a 3D IC can generally occur in one of three ways: wafer-to-wafer (W2W) stacking, die-to-wafer (D2W) stacking, or die-to-die (D2D) stacking. In W2W stacking, two wafers are bonded to one another, with dies in the same position on each wafer forming the stack. W2W bonding benefits from the highest manufacturing throughput of all of the stacking methods due to the need for only a single alignment step to bond two tiers of many stacks at the same time. Furthermore, it is easier for machines to handle and align large wafers than to do the same for dies when the dies are relatively small. For processes with particularly high wafer yields, pre-bond test may not need to be performed and unsorted wafers can be stacked to keep test costs low. The drawback of W2W bonding is that there is little flexibility as to which dies are bonded together. Due to this, W2W stacking is expected to have the worse compound stack yield out of the three methods as it may not be possible to prevent a bad die and a good die being bonded together.

In a W2W process, wafers are fabricated separately for each die in the stack, with Wafer 1 containing the dies for the lowest stack tier, Wafer 2 containing dies for the second stack tier, and so forth. These wafers then undergo pre-bond test to create the wafer map of good and bad dies. The wafers are placed in separate repositories, and wafer fabrication and test continues until the wafer repositories all contain an appropriate number of wafers. Once the repositories are filled, wafer matching and sorting can take place. The wafer maps for the wafers in each repository are sent to a computer and matched via a wafer matching algorithm in order to determine which wafers should be part of which stack.

It would be exceedingly costly to match all the wafers produced in a manufacturing run with all other wafers. By using limited-size repositories, only a small number of wafers must be matched to one another, thereby reducing the size of the matching problem to a manageable level. Two types of repositories – static or running – can be used for wafer matching.

Consider a repository that can hold $m$ wafers out of a production run of $e$ ($m \leq e$) wafers. In a *static* repository, the repository begins by being filled with $m$ wafers. The wafers in this repository are matched with one or more other repositories, and all repositories are then emptied of the matched wafers. Only after being emptied completely is the repository replenished, and this procedure is repeated $e/m$ times.

In a *running* repository, the repository begins by being filled and wafer matching takes place similarly to the static repository. However, after each matching step, only the best-matched wafer in the repository is removed. After the wafer is removed, it is immediately replaced by a new wafer and matching is performed again with a full repository. This process occurs $e$ times.

The wafer matching process that can be used is determined by the number of repositories that are used simultaneously during matching and the number of wafers within each repository that are considered during matching. There are two matching dimensions that depend on the number of repositories used:

- *Layer-by-Layer* (LbL) refers to the matching process whereby only two repositories are considered at any given time. For a stack of three dies, this means that repositories containing the wafers of the first two stack tiers are matched against one another and the wafers then bonded. To complete the stack, the repository of the partial stack is matched against the repository of the wafers containing Die 3, and then this wafer is bonded to the partial stack. This matching process is iterative for each bond in the stack.
- *All-Layers* (AL) refers to a matching process that considers all the repositories for each tier in the stack at the same time. This process is complete and may be capable of better results than an LbL process, but is also computationally more difficult.

There are also two wafer matching dimensions depending on the number of wafers considered in each repository simultaneously:

- *Wafer-by-Wafer* (WbW) is a greedy matching process whereby only the best-matched wafers between the repositories are considered at a time. These best-match wafers are then removed from the repository for purposes of matching, and the best-match is chosen from the remaining wafers. This process continues until the repositories are empty. Unlike LbL, where the repositories of wafers for two adjacent dies are completely matched and emptied before moving on to the next repository, WbW completes a wafer stack across all wafer repositories before matching wafers for the next stack.
- *All-Wafers* (AW) is an exhaustive matching process whereby every wafer in every repository visible to the process is considered simultaneously. In other words, all possible matching outcomes are checked for each of $m$ wafers across two or more repositories. Wafers are then matched to each other based on some criteria, for example maximizing the total expected compound stack yield across all wafers.

The dimensions from the above two lists can be matched to form five different wafer matching processes—LbL→WbW, WbW→LbL, LbL→AW, AL→WbW, and AL→AW. As an example, an LbL→WbW is iterative over two repositories at a time. In each iteration, the two best-matched wafers in each repository are chosen and removed from consideration. Each successive step matches the remaining wafers, which after the first step is either $m - 1$ for a static repository or $m$ again for a running repository. This continues until all wafers are matched, and then the process moves on to the repository for the next stack tier and so on.

Significant yield improvements can be achieved by using wafer matching in 3D integration. The authors of Ref. [23] generated a number of results using a benchmark process with 300 mm wafers. The stack utilized for results consisted of two dies, each of which was a square with an area (A) of 50 mm$^2$. This results in 1278 dies per wafer with a utilized wafer yield of 81.65%. **Figure 13** shows the increase in expected stack yield for multiple stack and repository sizes. The expected stack yield values are normalized for each stack size. As is demonstrated by the figure,

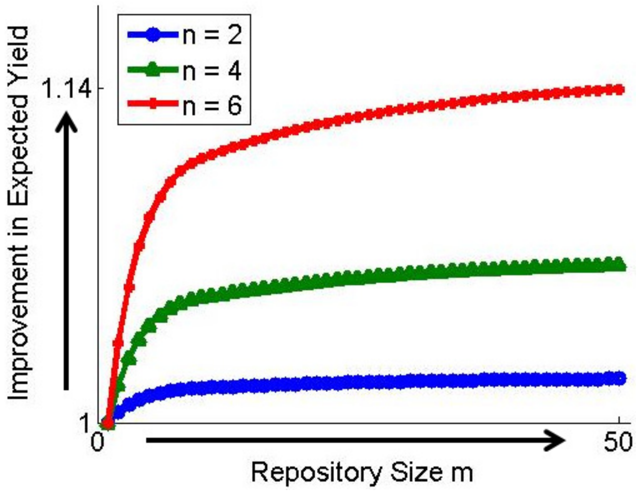


**Fig. 13** Increase in expected stack yield for various stack sizes $n$ and repository sizes $m$ [21].

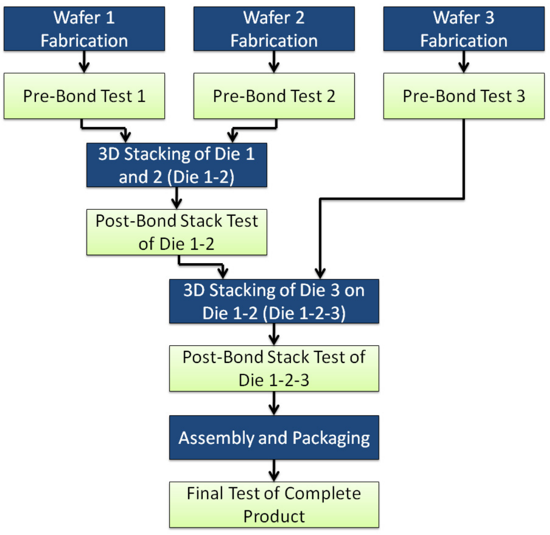

Wafer 1 Fabrication
Wafer 2 Fabrication
Wafer 3 Fabrication
Pre-Bond Test 1
Pre-Bond Test 2
Pre-Bond Test 3
3D Stacking of Die 1 and 2 (Die 1-2)
Post-Bond Stack Test of Die 1-2
3D Stacking of Die 3 on Die 1-2 (Die 1-2-3)
Post-Bond Stack Test of Die 1-2-3
Assembly and Packaging
Final Test of Complete Product

**Fig. 14** 3D IC manufacturing and test flow with multiple test insertions.

a significant yield increase is gained by using even small repositories, with most of the gain in yield seen at repository sizes under 10. As repository size increases further, the gain in yield begins to level off.

Similarly, the authors of Ref. [24] conducted experiments considering a variety of different matching processes with running repositories, using the same number of dies per wafer and wafer yield as in Ref. [23]. Due to the nature of running repositories, repository pollution must be controlled for. Repository pollution refers to the particularly faulty wafers in the repository rarely being matched with another wafer, effectively reducing the repository size over many match cycles. In Ref. [24], a first-in first-out (FIFO) method is devised to control pollution. In order to do this, the matching method changes which repository the first wafer is chosen from in a FIFO fashion. For example, for a process with $n = 3$ repositories labeled 1 through 3, for the first matching process a wafer is chosen from repository one via FIFO order and matched to wafers in the other repositories. For the second matching process, a wafer is first chosen from repository 2 in FIFO order and matched to the other repositories. After this, a wafer is chosen first from repository 3. After this has been done for all repositories, the sequence starts anew at repository 1. This process can control pollution by forcing the selection of the wafer that has been in a repository the longest when the repository is used to select the first wafer via FIFO order. Thus, a wafer will remain in a repository for at most $n \cdot m$ iterations. Results show that this method results in significant yield improvements above what is possible with just static repositories.

Wafer matching continues to be an important research problem, and recent work on this topic is reported in Refs. [25], [26].

## 4. Design-for-Test and Test Optimization

Due to the addition of new tests and test insertions for 3D ICs, test cost must be kept low to ensure that 3D IC production is economically feasible. In order to reduce test cost, optimizations need to be developed to design test architectures and create test schedules, to determine the placement of recovery architectures for faulty TSVs, and to reduce the impact of test circuitry on functional operation.

### 4.1 Test Scheduling for Partial and Complete Stacks

Post-bond test-architecture and test schedule optimization for 3D-stacked ICs can reduce test cost by minimizing test time. These optimizations must take into account 3D-specific test access, TSV count, and multiple test insertion constraints. Two different 3D integration cases are considered—(1) *hard dies*, in which a test architecture already exists, and (2) *soft dies*, for which the 2D (per die) and 3D (per stack) test architectures are co-optimized. For the sake of simplicity and ease of implementation, this section assumes session-based test scheduling [27], i.e., in which all tests which are executed simultaneously need to be completed before the next test session is started. While it is theoretically possible to have multiple dies on a given layer in a stack, for this section it is assumed that there is only one die per layer in a stack. Furthermore, a core is considered to be part of a single die only, i.e., "3D cores" are not considered. The optimization method presented in this section utilizes emerging standards, such as die-level wrapper described in Ref. [12].

Compared to two-dimensional ICs that typically require two test insertions, namely wafer test and package test, 3D stacking introduces a number of natural test insertions [2]. Because the die-stacking steps of thinning, alignment, and bonding can introduce defects, there may be a need to test multiple subsequent (partial) stacks during assembly. **Figure 14** shows an example manufacturing and test flow for a 3D stack. First, wafer test (i.e., pre-bond test) can be used to test die prior to stacking to ensure correct functionality, as well as to match die in a stack for power and performance. Next, Die 1 and Die 2 are stacked, and then tested again. If pre-bond TSV test is not performed, then this test insertion will be the first test of the TSVs between Die 1 and Die 2. This step also ensures that defects can be detected in the stack due to additional 3D manufacturing steps such as alignment and bonding. The third die is added to the stack and all dies in the stack, including all TSV connections, are retested. Finally, the "known good stack" is packaged and the final product is tested.

Optimization methods are needed to minimize test time not only for the final stack test, i.e., if the intermediate (partial) stacks are not tested, but also to minimize the total test time if the final stack and partial stacks are tested during bonding.

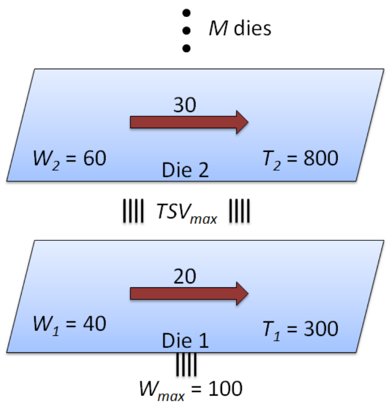
In a 3D IC, which currently consist of anywhere from two to eight dies [28], the lowest die is usually directly connected to chip I/O pins and therefore can be tested using test pins. To test the non-bottom dies in the stack, test data must enter through the test pins on the lowest die. Therefore, to test other dies in the stack, the test access mechanism (TAM) must be extended to all dies in the stack through the test pins at the lowest die. To transport test data up and down the stack, "TestElevators" [12] need to be included on each die except for the highest die in the stack [2]. The number of test pins and TestElevators as well as the number of TSVs used affect the total test length for the stack.

To illustrate test architecture and schedule optimization, consider an example 3D IC with three hard dies with given test access architectures as shown in **Fig. 15**. For a hard die, the 2D test architecture on the die is fixed. The only structure over which the designer has control is the 3D TAM. Hard dies offer less flexibility for optimization in the sense that each die must have exactly the pre-defined number of input and output TAM wires appropriated to it in the design of the 3D TAM. Therefore, the only decisions that can be made in designing the 3D TAM is which (if any) dies can be tested in parallel with one another given the limitations on test pins and test TSVs. Hard dies may be present in TAM design problems if vendors sell fabricated dies to a 3D integrator.

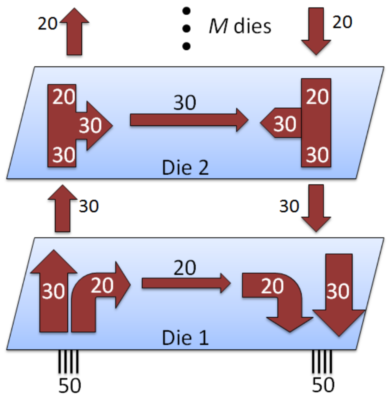
Figure 15 (a) illustrates the variables that arise for the hard-die problem. As can be seen, a fixed 2D TAM width (the horizontal arrow) is given along with the known test time $T$ for each die. The given constraints are the number of test pins $W_{\max}$ and the number of test TSVs $TSV_{\max}$ available. A solution, therefore, can be given as in Fig. 15 (b). Here, each die receives the required and pre-defined test bandwidth, but Die 1 and Die 2 are tested in parallel through the 3D TAM.

Optimizations published in Refs. [29], [30] utilized integer-linear programming to create an optimization model for the 3D test-access mechanism and test schedule considering any or all possible test insertions. The number of available test pins $W_{\max}$ were all situated on the bottom die. A limit $TSV_{\max}$ was placed on the number of test elevators that could be added between any two dies. A five-die stack was created from the ITC'02 test benchmarks. Test time results for a hard-die stack are shown in **Fig. 16** (a), and for soft dies in Fig. 16 (b) with respect to varying $TSV_{\max}$ and $W_{\max}$.

From the figure, it can be seen that both $TSV_{\max}$ and $W_{\max}$ determine which dies should be tested in parallel and thus the total

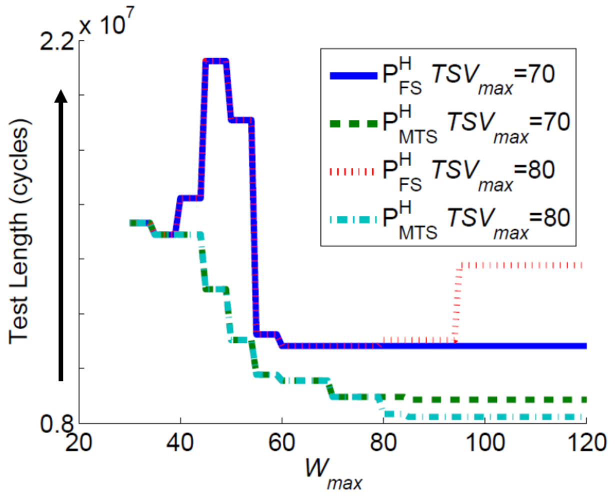


(a) Problem instance

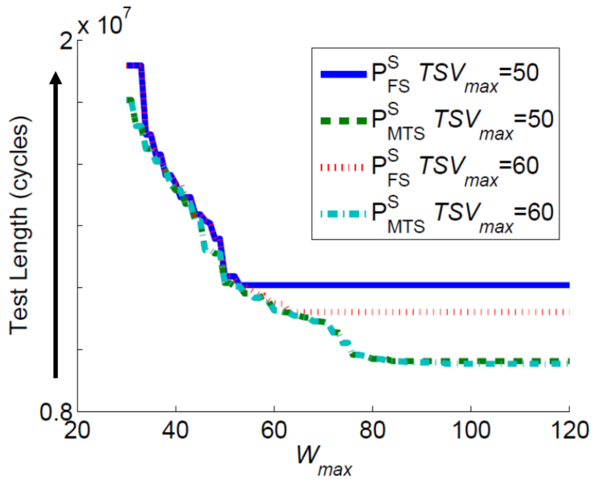


(b) Architecture solution

**Fig. 15** Illustration of 3D TAM optimization with hard dies: (a) a problem instance; (b) an optimized architecture.



(a)



(b)

**Fig. 16** The test time with respect to $TSV_{\max}$ and $W_{\max}$ considering all possible test insertions for (a) hard dies and (b) soft dies [30].

test time for the stack. For a given value of $TSV_{\max}$, increasing $W_{\max}$ does not always decrease the test time for hard dies, although test time never increases. Similarly, increasing $TSV_{\max}$ for a given $W_{\max}$ does not always decrease the test time. These Pareto-optimal points provide insight in how many test resources are needed for a stack. Optimizing only for the final stack test does not always reduce test time when multiple test insertions are considered. In fact, it is often the case that the test time is higher when increasing the values of $TSV_{\max}$ and $W_{\max}$.

For optimization of 3D ICs with soft dies, Pareto-optimality is almost non-existent when $W_{\max}$ is varied. This is due to the fact that when dies in the stack are soft, it is almost always possible to find one die for which adding an extra test pin reduces the overall test time. When test time stops decreasing, the TSV limits for the problem instance have been reached and no further optimization is possible.

## 4.2 Retiming for Delay Recovery

In order to enable both pre- and post-bond testing, die level wrappers have been proposed in the literature [12], [13] and have been briefly discussed earlier in this paper. These die wrappers include boundary scan cells at the interface between die logic and TSVs to add controllability and observability at the TSV, and are utilized as gated scan-flops in pre-bond TSV probing. Adding boundary flops to TSVs between two layers adds two additional clocked stages to a functional path, which would otherwise not exist in a two-dimensional (2D) design. Bypass paths can be added to the boundary scan cells to multiplex the functional input between being latched in the flop or being output directly to/from the TSV. During functional mode, the bypass path is active and a signal traveling across dies is never latched in the boundary register; however, the bypass path still introduces additional path latency. It is important to note that additional latency is added by test architectures not just to logic-on-logic stacks, but also to memory-on-logic and memory-on-memory stacks that contain dies with die wrapper boundary registers.

Retiming is an algorithmic approach to improving multiple aspects of circuit design post-synthesis by moving the positions of registers with respect to combinational logic while preserving circuit functionality [31]. The concept of retiming can be extended by utilizing retiming algorithms to recover the latency added by boundary scan cell bypass paths in a 3D IC [32]. Retiming in this case is performed after synthesis and 3D boundary cell insertion. Two-dimensional retiming methods can be reused after test-insertion by fixing the location of die wrapper boundary flops so that they remain at the logic/TSV interface during retiming. This requirement ensures that die logic is not moved across dies when retiming is performed on a complete stack. A bypass path is added to each register so that in functional mode, data does not need to be latched, thereby replacing extra clock stages with added latency. Retiming is then performed to recover the added latency of the bypass along the TSV path.

Retiming can be performed either at the die- or stack-level. Die-level retiming allows greater control over the redistribution of slack throughout the stack. For example, a designer may not want to move any registers on a particular die $D$ but would still like to recover the additional latency of adding a wrapper to that die. In this case, additional (dummy) delay can be added to the TSV paths on the dies adjacent to $D$. Retiming can then be performed on the adjacent dies in an attempt to recover the additional path delay due to the wrapper cells in the adjacent dies and the wrapper cells of $D$. While die-level retiming does not consider the total path delay for paths that cross die boundaries, stack-level retiming can exploit this added degree of freedom. In stack-level retiming, the complete stack is retimed as a monolithic entity. The boundary GSFs are once again fixed during retiming to prevent the movement of logic from one die to another. During stack retiming, the intended clock frequency for the stack can be used as a timing target because all circuit paths are known. While die-level retiming provides more control over slack redistribution, stack-level retiming provides greater leeway to the retiming algorithm.

A flowchart of the steps required for post-wrapper insertion die-level retiming is shown in **Fig. 17**. First, the design is synthesized into a structural circuit definition. In die-level retiming, paths crossing the TSV interface are incomplete and the total delay across these paths cannot be considered. Because paths crossing die boundaries are likely to be the least-slack paths in a 3D stack, the clock period for the stack may be too large to provide a tight timing constraint when considering a single die. In order to determine an appropriate timing target, timing analysis is performed to identify the amount of slack on the least-slack path of the die. The target clock period for retiming is incrementally reduced until the least-slack path has no positive slack. Wrapper insertion is then performed, adding delay to the TSV paths equal to the bypass path through a boundary GSF. During retiming, boundary GSFs are fixed so that the retiming algorithm does not consider them as movable registers nor does it attempt to move logic or other registers past the GSFs. After the retiming algorithm has executed, timing analysis is again performed to deter-

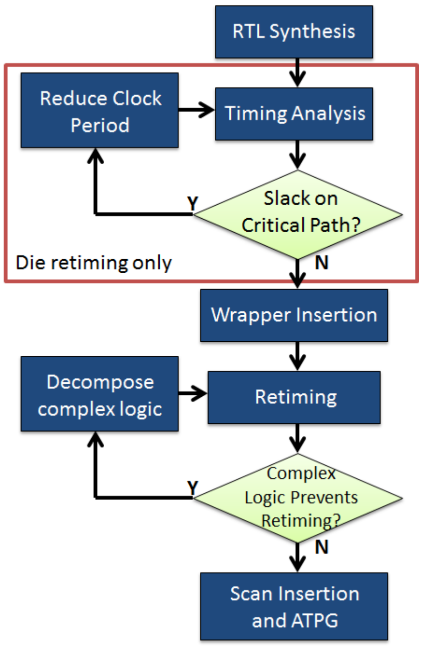


**Fig. 17** Flowchart for retiming of a 3D stack at either die- or stack-level [32].

mine if all paths in the die satisfy the target timing constraint. If they do not, the path that has the most negative slack is examined to determine if complex logic gates on the path may be preventing retiming. If complex logic gates are preventing retiming, they are decomposed into simple logic cells from the cell library and retiming is performed again. This process continues until all paths meet the timing target or no complex logic gates are restricting retiming. Finally, scan insertion and ATPG are performed for the die.

The authors of Ref. [32] showed that retiming can be effective in reducing the timing overhead of wrapper boundary cell insertion on critical inter-die paths. They partitioned a DES core from the OpenCore benchmark circuits across four dies. **Table 1** shows retiming results in terms of percentage of the delay of the wrapper cells that can be recovered and the change in stuck-at pattern count between the original and retimed circuit. As can be seen from the results, retiming recovered 100% of the additional GSF bypass mode delay in most cases. The effect of DfT-insertion and retiming on stuck-at fault pattern count was found to be negligible. In some cases, this resulted in fewer test patterns after retiming. In other cases, retiming incurred a small increase in pattern count. Since registers are moved or added throughout the circuit, controllability and observability changes variably.

## 4.3 Robust Optimization of 3D Test-access Architectures

Due to the potential of 3D stacks to integrate a large number of large SoCs, the complexity and the cost of test is increased. Therefore, 3D test requires careful attention in order to optimize test cost. Recent work on 3D test strategies have addressed this issue and presented several methods for test architecture optimization and test scheduling. These methods are based on exact optimization techniques such as integer linear programming (ILP) and heuristics such as rectangle packing [33], [34]. These conventional optimization methods assume that all input parameters, such as core test times and test power limits are known constants. However, the values of some of these parameters are not known precisely at the design stage, for instance, due to inaccuracies in simulations. Moreover, in a 3D scenario, a die can be designed to be used in multiple stacks with different properties, such as power limits and number of test pins.

Optimization methods targeting a single point in the input parameter space may result in non-optimal solutions in practice. Therefore, uncertainties in input parameters should be taken into account at the design optimization stage in order to provide "robust solutions." The goal of robust optimization is to find a solution that remains close to the optimum in the presence of parameter variations.

Table 1 A comparison of delay, area, and pattern count results for die- and stack-level retiming for the DES logic circuit partitioned across four dies.

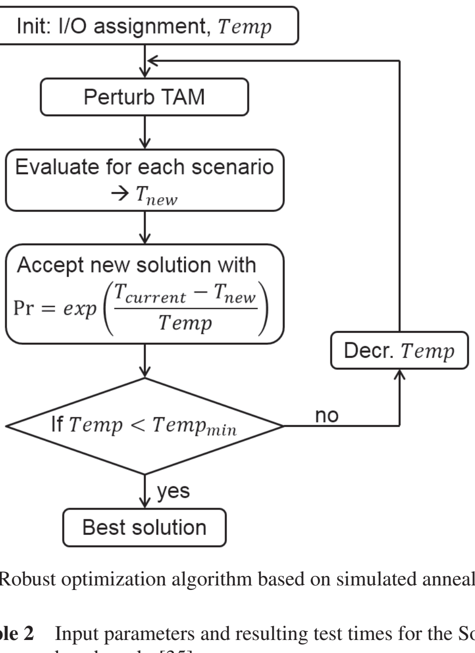
| | Die 0 | Die 1 | Die 2 | Die 3 | Complete Stack |
|---|---|---|---|---|---|
| % Delay Recovered | 100 | 100 | 60 | 100 | 100 |
| % Change in Pattern Count | −2.5 | −4.2 | 0.8 | 5.3 | 8.1 |

A robust optimization approach has been proposed in Ref. [35]. This work formulates an integer linear programming (ILP) model for robust optimization of test architecture design and test scheduling, which allows for optimization over a range of input parameters and minimizes the expectation of test time. The proposed ILP model is practical for systems, such as SoCs or 3D-stacked ICs, that contain a relatively small number of cores. For larger systems, for which the exact solution of the ILP model becomes intractable, a heuristic algorithm based on simulated annealing has been proposed. This algorithm uses the exact ILP method to solve small sub-problems.

The optimization algorithm is outlined in **Fig. 18**. This heuristic starts with an initial solution for the test architecture, which is iteratively perturbed in order to optimize the test time. After each perturbation, the new solution is evaluated using the ILP model for a range of input parameters. A better solution is always accepted and a worse solution is only accepted with a certain probability based on the difference between the solutions and an artificial variable "temperature" which is decreased at every iteration step. After a certain number of iterations, the algorithm stops and the best solution is kept.

The proposed test-optimization method has been verified on ITC'02 benchmarks that were used as 3D dies. Even though the ILP model for the complete problem requires long CPU times for realistic dies with many cores, this model can be used in a heuristic to solve small sub-problems, i.e., for evaluation of the solution for each scenario.

**Table 2** shows the results of this experiment. The uncertain parameters are the nominal maximum TAM width of the SOC $W_{m,n}$ and the nominal maximum test power $P_{m,n}$. Nine discrete points



Init: I/O assignment, $Temp$
Perturb TAM
Evaluate for each scenario $\rightarrow T_{new}$
Accept new solution with $\Pr = exp\left(\frac{T_{current} - T_{new}}{Temp}\right)$
If $Temp < Temp_{min}$
no
Decr. $Temp$
yes
Best solution

**Fig. 18** Robust optimization algorithm based on simulated annealing [35].

**Table 2** Input parameters and resulting test times for the SoC benchmarks [35].

| Benchmark | $W_{m,n}$ | $P_{m,n}$ | $T_{nom}$ | $T_{rob}$ | impr |
|---|---|---|---|---|---|
| h953 | 10 | 60 | 156.7 | 145.6 | 7% |
| d695 | 32 | 50 | 144.0 | 131.0 | 9% |
| p93791 | 100 | 45 | 223.5 | 207.9 | 7% |
| d695_h953 | 32 | 60 | 191.7 | 161.0 | 16% |
| p22810 | 50 | 60 | 1844.0 | 1634.0 | 11% |

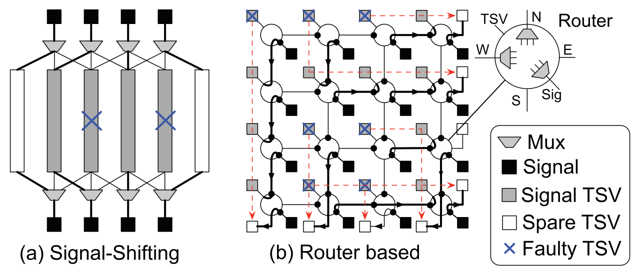


**Fig. 19** Illustration of TSV repair solutions [22], [36].

for maximum test power $P_m$ and maximum TAM width $W_m$ are assumed, such that the nominal values occur with the probability of 0.6 and the other two values that are offset from the nominal value by ±20%, with the probability of 0.2 each. The results show that the test time using robust optimization $T_{rob}$ is measurably reduced compared to the nominal test time $T_{nom}$. Therefore, neglecting variations in input parameters will result in inefficient single-point solutions with increased test time.

## 4.4 TSV Repair

One promising approach to ensure high yield of 3D-stacked ICs is to add spare TSVs for built-in self-repair (BISR). Various TSV repair strategies based on TSV redundancy have been recently reported in the literature for manufacturing yield enhancement [22], [36], [37], [38], [39].

The repair capability of these solutions vary according to their different redundancy architectures and the corresponding repair algorithms. In Refs. [36], [37], one or more redundant TSVs are added to a group of TSVs to achieve different grouping ratios and a defective TSV is swapped with a fault-free one via signal shifting, shown in **Fig. 19** (a). The hardware overhead includes a set of $(1 \times m)$ multiplexers added before and after the TSVs. Once a faulty TSV is identified during pre- or post-bond test, TSV shifting is enabled by using a fuse on the multiplexer to select a redundant TSV in order to repair the faulty one. In Ref. [40], spare TSV rows are added to a TSV array for repair to reduce storage requirements for the reconfiguration data. Figure 19 (b) shows a router-based TSV repair architecture which is based on graph model [22].

In practice, however, if a TSV has a manufacturing defect, it is more likely that neighboring TSVs are also defective due to clustering [41]. This issue has been considered in Refs. [38], [42]. In Ref. [42], a probabilistic model of defect clustering is considered to trade off TSV grouping strategies, thus optimizing the grouping ratio under different defect clustering distributions. In Ref. [38], TSV redundancy is allocated and optimized through integer linear programming, considering the constraints of delay introduced by TSV rerouting and the efficiency of grouping ratios.

In Ref. [39], in-field TSV repair was proposed to target TSV failures in-field (at time $t > 0$). Electromigration-induced aging effects increase delay in TSVs at different rates under various workloads. Regular and redundant TSV grids are modeled using edge-disjoint repair paths. **Figure 20** shows implementation of the in-field TSV repair framework based on routers.

In Ref. [43], the authors proposed a repair strategy that is significantly different from TSV redundancy. Resistive-short faults

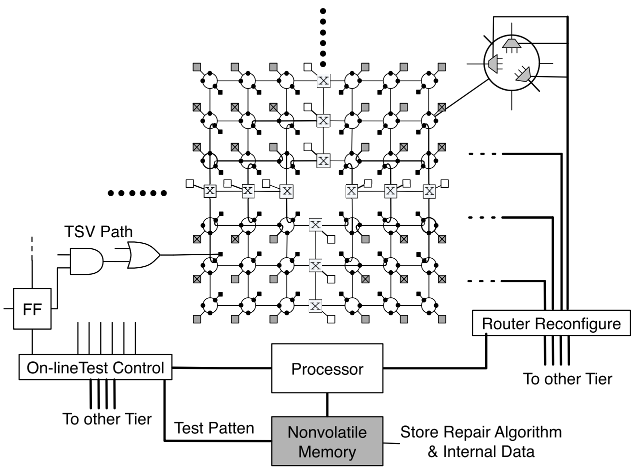


**Fig. 20** Illustration of route-based TSV repair architecture [39].

are targeted, where signals in TSVs tend to be weakened due to voltage degradation. A signal-restoration scheme was proposed to strengthen the signal based on a custom analog comparator, which is specifically designed to reduce the overhead.

## 4.5 Interconnect Testing for 2.5D ICs

Interposer-based 2.5D ICs are being advocated as an alternative choice for IC design and they are emerging as precursors to 3D integration. In 2.5D ICs, dies are placed next to each other on a silicon interposer. An interposer is a passive device providing interconnects between active dies stacked on top of it and connections to the package I/Os [44], [45]. Interposer testing is usually done at the post-bond stage in order to detect defects in the interposer (e.g., shorts and opens), as well as defects due to missing or deformed micro-bumps faults that cause misalignment between dies, micro-bumps, and the interposer.

Due to probe technology limitations, these horizontal interconnections cannot be accessed from the front side via probing after micro-bumps are mounted on the silicon interposer [46]. Therefore, the interposer can only be tested from one side. Recent efforts have identified some potential solutions to post-bond interposer testing [47], [48]. They are based on extensions of the IEEE 1149.1 standard test access port (TAP) and the associated boundary-scan architecture (JTAG) [49].

In this subsection, a new boundary-scan test method is presented for interposer interconnect testing [50]. This method uses a scan chain to load stimuli at one end of an interposer wire and capture response at the other end. Two types of boundary-scan cells are needed to support interconnect testing, namely the scan-in boundary-scan cell (SIBC) and the scan-out boundary-scan cell (SOBC). The SIBCs are connected to the primary outputs of each die. The SOBCs are connected to the primary inputs of each die. The structures of SIBCs and SOBCs, as shown in **Fig. 21**, are different from each other since they serve different purposes during test. Their control signals and corresponding modes are listed in **Table 3**.

In order to implement at-speed interconnect testing, an RC delay block and a multiplexer are included in SOBC. The RC delay block is implemented using a low-pass second-order Bessel filter in order to generate a functional clock period delay. A multiplexer

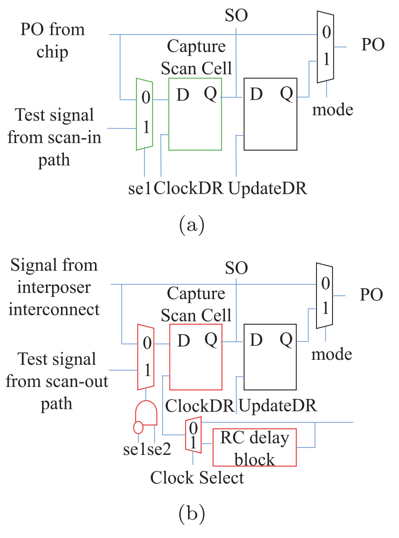


**Fig. 21** Designs of the (a) scan-in boundary-scan cell (SIBC) and the (b) scan-out boundary-scan cell (SOBC), RC delay block is used to generate a functional clock period delay for at-speed test [50].

**Table 3** Values of control signals for the boundary-scan cells and corresponding operational modes.

| Control signals | SIBC mode | SOBC mode |
|---|---|---|
| se1 = 1, se2 = 0 | Test mode | Functional mode |
| se1 = 0, se2 = 1 | Functional mode | Test mode |
| se1 = 0, se2 = 0 | Functional mode | Functional mode |

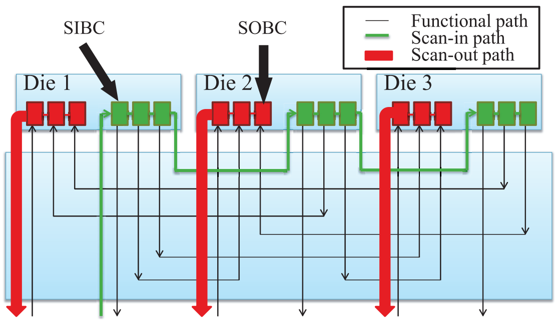


**Fig. 22** Generic test architecture for silicon interposer [50].

determines whether the original clock or the delayed clock serves as the clock signal for SOBCs. During open/short testing, since a common test clock is used, Clock_select is set to 0. During delay test, after the last shift-in clock cycle, Clock_select is toggled. Next, an additional clock cycle is applied to launch the test patterns out of SIBCs. SOBCs will receive a clock that is delayed by one functional cycle in order to capture test responses. In this way, delay testing can be efficiently implemented.

Since the SIBCs and SOBCs are functionally independent, boundary-scan chain is divided into two separated chains in the design, namely the scan-in path and the scan-out path. In the scan-in path, all the SIBCs are grouped and connected together. Similarly, all the SOBCs are grouped and connected together in the scan-out path. In this way, the scan-in of test stimuli and the scan-out of test responses can be carried out in parallel, which leads to a reduction in interconnect testing time. The scan-in path and scan-out path are highlighted in **Fig. 22**. With the proposed on-chip test architecture, probe needles can be employed to shift test patterns and capture responses. The test procedures for detecting opens, shorts, and delay defects are described next.

Testing for opens and shorts involves the following steps:

(1) SIBCs are set to operate in test mode and SOBCs are set to operate in functional mode by setting se1 to 1 and se2 to 0. Test patterns are shifted into the SIBCs through the scan-in path.

(2) Test patterns are applied to the interconnects. Note that some SOBCs are not connected to the SIBCs through interconnects, but are connected to the C4 bumps (solder bumps on the bottom of an interposer in order to mount the interposer onto the package). Test patterns are applied to these SOBCs directly from probe needles through C4 bumps and TSVs. This procedure is used for testing vertical interconnects; the test path consists of C4 bumps, TSVs, and micro-bumps connected to SOBCs.

(3) All test responses are captured by SOBCs. Note that some SIBCs are not connected to SOBCs through interconnects, but are connected to the C4 bumps. Thus, the test patterns launched by these SIBCs cannot be captured by the SOBCs, but are captured by probe needles through TSVs and C4 bumps. This procedure is used for testing vertical interconnects; the test path consists of C4 bumps, TSVs, and micro-bumps connected to SIBCs

(4) SIBCs are set to operate in functional mode and SOBCs are set to operate in test mode by setting se1 to 0 and se2 to 1. Test responses are shifted out through scan-out path. Meanwhile, new test patterns are shifted into the SIBCs through the scan-in path.

Testing for delay faults involves the following steps:

(1) Test patterns are shifted into the launch cells through the scan-in path.

(2) After all test patterns are stored in SIBCs, "Clock_select" is set to 1. The clock of SOBCs will come from an RC delay block. This guarantees the test responses can be captured at-speed.

(3) The test patterns are launched to the interconnects when one additional clock cycle is applied to SIBCs. Meanwhile, this clock goes through the RC delay block and generates a delayed capture clock. Test responses are captured at-speed on the rising edge of the delayed clock.

(4) Test responses are shifted out through the scan-out path. Meanwhile, new test patterns are shifted into the SIBCs through the scan-in path.

Results obtained using this approach can be found in Ref. [50].

## 5. Cost Modeling and Test-Flow Selection

Test cost has emerged as a potential showstopper in the adoption of 3D integration. The choice of test flow, i.e., what tests are used and when they are applied during 3D integration ("what to test", "when to test") affects test cost. 3D stacking involves many possible test insertions. Due to multiple yield and test cost parameters corresponding to different dies and tests, such as for pre-bond, post-bond, and partial stack, an exponentially large number of test flows must be evaluated. Therefore, analysis methods and tools are needed for test-cost optimization and automated test-flow selection.

Several papers have been published recently on various aspects of test-cost modeling and optimization for 3D-stacked ICs [51], [52], [53]. These papers include attempts to hand-pick a test flow or select a test flow based on explicit enumeration of a few candidate test flows. However, prior work does not provide any means for systematically exploring (e.g., through implicit enumeration) the solution space of all possible test flows and reporting the test flow that minimizes test cost.

### 5.1 Cost Model

Test-cost optimization for 3D-stacked ICs was considered in Ref. [54] using a cost model that takes into account various test costs at each step of the stacking process. The model is generic and flexible in that it provides placeholders for different test costs that are typically incurred during 3D integration. The proposed model can be adapted for wafer-to-wafer (W2W), die-to-wafer (D2W), and die-to-die (D2D) stacking. A heuristic procedure is described that is guided by a matrix-partitioning problem. The results highlighted the impact of various parameters on test cost and test-flow selection.

Figure 14 sketches a typical integration of three dies to form a stack. This process involves incremental stacking of one die at a time. During testing of a stack, tests targeting faults in either of the dies present in the stack can be applied. Hence, in addition to pre-bond testing, a die can be tested at multiple stages of stacking as well. Moreover, there can be multiple types of tests, having different fault coverage and test application costs. If a test with lower cost is selected, that can save expenses upfront, but the lower fault coverage can result in the stacking of defective dies, thereby lowering the overall stack yield, and consequently increasing the overall cost of stack creation. Therefore, test-flow selection involves selection of the test insertions (also referred to as test moments in Ref. [51]), and tests that provide the best trade-off between cost and quality. For this simple example, there exist $2^{3+3+2} = 256$ different combination of test insertions: tests for Die 1 and Die 2 can be applied at all the three stages and there are two stages in which tests for Die 3 can be applied. Moreover, for each selection of insertions, there are different ways in which tests can be selected out of a given test set. It can be easily shown that the test insertions can be selected in $O(2^{l^2})$ possible ways, where $l$ is the total number of dies in a stack. The optimization problem rapidly becomes intractable with the addition of dies, and because of the added complexity associated with selection of tests. If we consider inter-die interconnect tests, even more test flows are possible. An important goal is to minimize the total cost that includes test cost and the part of manufacturing cost that is affected by yield and test escapes.

A subset of the notation used in Ref. [54] is shown in **Table 4** and **Table 5** for the given parameters and the unknown variables, respectively, that are used in the model described in Ref. [54]. Constrained by the availability of space, we illustrate the model using a simple example of a stack consisting of two dies. Suppose that the fault coverage of the pre-bond test chosen for Die 1 (Die 2) is $fb_1$ ($fb_2$) and that for the post-bond test is $f_1$ ($f_2$). For simplicity, we do not account for interconnect testing in this example. Depending on whether pre-bond tests are applied, **Table 6**

**Table 4** Table describing the given parameters used in the cost model [54].

| Symbol | Meaning |
|---|---|
| | Parameters related to pre-bond testing |
| $l$ | Total number of dies in the stack |
| $\mathcal{D}_i$ | $i^{th}$ die in the stack |
| $DC_i$ | Manufacturing cost of each instance of $\mathcal{D}_i$ |
| $n_i$ | Number of instances of $\mathcal{D}_i$ manufactured. |
| $\lambda_i$ | Yield for die $\mathcal{D}_i$. |
| | Parameters related to stack testing |
| $\mathcal{S}_k$ | A stack of dies $\mathcal{D}_1, \mathcal{D}_2, \ldots \mathcal{D}_k$, $2 \leq k \leq l$. The stack $\mathcal{S}_k$ is partial for $k < l$. |
| $SC_k$ | Manufacturing cost of each instance of $\mathcal{S}_k$ |
| $\omega_{ik}$ | The component of bond yield of stack $\mathcal{S}_k$ related to defects induced in die $\mathcal{D}_i$ during the stacking of $\mathcal{S}_k$. The overall bond yield for stack $\mathcal{S}_k$ is given by $\prod_{i=1}^{k} \omega_{ik}$. |
| $PC$ | Cost of packaging and testing a full stack |

**Table 5** Table describing the unknown variables used in the cost model [54].

| Symbol | Meaning |
|---|---|
| $n'_i$ | Number of instances of $\mathcal{D}_i$ that are available for creating $\mathcal{S}_i$. |
| $m_k$ | Number of instances of $\mathcal{S}_k$ manufactured |
| $m'_k$ | Number of instances of $\mathcal{S}_k$ that have been detected to be fault-free after stack testing. |

**Table 6** Table showing the number of instances available dies after pre-bond testing.

| Case | Pre-bond test | | $n'_i$ | |
|---|---|---|---|---|
| | Die 1 | Die 2 | Die 1 | Die 2 |
| Case 1 | X | X | $n_1$ | $n_2$ |
| Case 2 | ✓ | X | $n_1 \cdot \lambda_1^{fb_1}$ | $n_2$ |
| Case 3 | X | ✓ | $n_1$ | $n_2 \cdot \lambda_2^{fb_2}$ |
| Case 4 | ✓ | ✓ | $n_1 \cdot \lambda_1^{fb_1}$ | $n_2 \cdot \lambda_2^{fb_2}$ |

**Table 7** Table showing the number of instances of fault-free stack remaining after stack testing for Case 3 in Table 6.

| Case | Post-bond test | | $m'$ |
|---|---|---|---|
| | Die 1 | Die 2 | |
| Case 1 | X | X | $m_2 = \min\{n_1, n_2 \cdot \lambda_2^{fb_2}\}$ |
| Case 2 | ✓ | X | $m_2 \cdot (\omega_{12} \cdot \lambda_1)^{f_1}$ |
| Case 3 | X | ✓ | $m_2 \cdot (\omega_{22})^{f_2}$ |
| Case 4 | ✓ | ✓ | $m_2 \cdot (\omega_{12} \cdot \lambda_1)^{f_1} \cdot (\omega_{22})^{f_2}$ |

shows values of $n'_i$ for both the dies. The number of instances of stack $\mathcal{S}_2$ created after stacking is $m_2 = \min\{n'_1, n'_2\}$.

For each of the cases listed in Table 6, there exist four different cases based on the decisions taken for applying post-bond tests. For Case 3 in Table 6, we list these four cases in **Table 7** and show the number of fault-free stacks remaining after applying stack test for each case. Based on the cases chosen for pre-bond test and post-bond test, the final cost of manufacturing and testing IC differs. For example, on choosing the Case 3 from Table 6 and the Case 2 from Table 7, the final cost can be computed as $n_1 \cdot DC_1 + n_2 \cdot (DC_2 + b_2) + m_2 \cdot SC_2 + m_2 \cdot t_1 + m'_2 \cdot PC$, where $b_2$ and $t_1$ are the costs associated with the selected pre-bond test for $\mathcal{D}_2$ and post-bond test for $\mathcal{D}_1$, respectively. The objective of the optimization problem is to find a test flow with minimum cost of manufacturing and testing the IC.

### 5.2 Experimental Results

Experiments were run on stacks consisting of more than three dies. Different test costs and corresponding fault-coverage val-

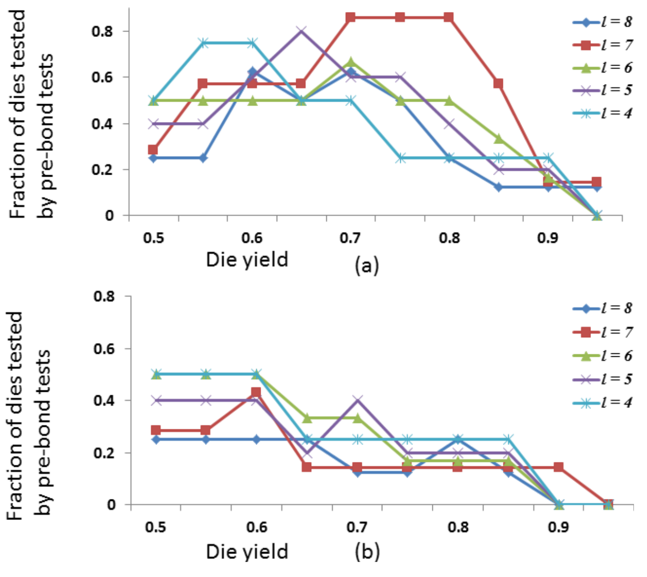


**Fig. 23** Effect of varying die yield on pre-bond test selection under (a) low pre-bond test cost, and (b) high pre-bond test cost [54].

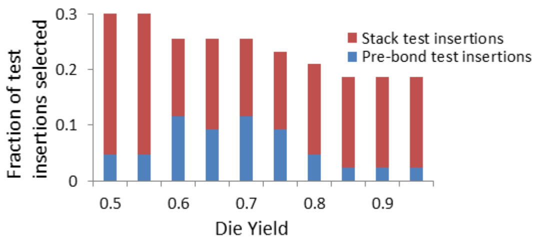


**Fig. 24** Fraction of all possible test insertions selected as a function of die yield for a stack with eight dies [54].

ues were arbitrarily chosen for these experiments. **Figure 23** (a) depicts the effect of varying die yields on the fraction of dies to which pre-bond tests were applied. The die yields for all dies are swept from 0.5 to 0.95. The bond yields are fixed in a range of 0.9 to 0.95. For low values of die yield, the number of dies tested pre-bond is usually higher. Note that a cost-effective solution to a test-flow selection problem depends on the problem instance; therefore, a generic rule cannot be established that all dies with low yield must always be tested before stacking. This is also highlighted by Fig. 23 (a). Those dies that are not tested before stacking at low values of die yield, are tested later after stacking. The optimization tool decided to skip pre-bond tests for dies at low die yield values because it was able to find cost-effective tests downstream in the stacking process instead of investing heavily on pre-bond tests for faulty dies upfront. Experiments were repeated by doubling the cost of pre-bond tests. The results are shown in Fig. 23 (b). The fraction of dies chosen is less in this case, further highlighting the importance of a cost-analysis tool. **Figure 24** shows that, as expected, the fraction of test insertions chosen decreases with increasing die yield.

In an another experiment, bond yields are varied from 0.5 to 0.95 after fixing the die yields to high values (in a range of 0.9 to 0.95). On plotting the fraction of selected stack test insertions against bond yield values (**Fig. 25**), it was found that as the bond yield increases, the number of selected stack test insertions decreases.

The heuristic method is also compared with an exhaustive enu-

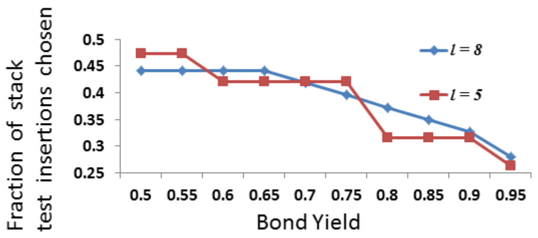


**Fig. 25** Effect of varying bond yield on stack test selection [54].

meration procedure for stacks with up to 10 dies. According to the results reported in Ref. [54], exhaustive enumeration is not feasible for these large designs, and the heuristic method reported far better results when the former was restricted to run for 3 hours of CPU time.

## 6. Conclusions

Despite the benefits of 3D ICs, their manufacturing cost is a major showstopper preventing high-volume manufacturing of 3D ICs. Test is among the most important important factors affecting manufacturing cost. Therefore, traditional test methods must be evaluated and adapted, and new test techniques developed, for 3D ICs in order to reduce cost and increase product quality.

This survey paper covered a variety of different 3D test techniques. We presented methods for pre-bond TSV testing an repair, including BIST-like methods and methods based on TSV probing. Since there are multiple ways of stacking dies, such as wafer-to-wafer, die-to-wafer, and die-to-die stacking, it is necessary to determine the stacking strategy in order to optimize the stack yield. Another aspect of 3D test we focused on is optimization of design-for-test and test scheduling. This includes (a) design of stack-level test architectures, (b) retiming for recovering the latency added by wrapper cells, (c) robust optimization of die-level test-access architectures, (d) TSV repair through redundant TSVs, and (c) interposer testing for 2.5D ICs. Test cost is also affected by the choice of the test flow during 3D integration. Therefore, we have presented cost modeling for a 3D test flow and an efficient heuristic that that solve the test-flow-selection optimization problem for a large number of dies in a 3D stack.

**Acknowledgments** The authors acknowledge research support from the National Science Foundation under grant No.CCF-1017391, the Semiconductor Research Corporation (SRC) under contract No.2118.001, Intel Corporation, and Cisco Systems.

## References

[1] Lee, H.-H.S. and Chakrabarty, K.: Test Challenges for 3D Integrated Circuits, *IEEE Design & Test of Computers*, Vol.26, No.5, pp.26–35 (2009).
[2] Marinissen, E.J. and Zorian, Y.: Testing 3D Chips Containing Through-Silicon Vias, *Proc. IEEE International Test Conference* (*ITC*), pp.1–11 (2009).
[3] Chen, H., Shih, J., Li, S., Lin, H., Wang, M. and Peng, C.: Electrical Tests for Three-Dimensional ICs (3DICs) with TSVs, *Informal Proc. International Test Conference 3D-Test Workshop* (2010).
[4] Deutsch, S. and Chakrabarty, K.: Non-Invasive Pre-Bond TSV Test Using Ring Oscillators and Multiple Voltage Levels, *Proc. Design, Automation, and Test in Europe* (*DATE*) *Conference*, pp.1065–1070 (2013).
[5] Lee, K.: Trends in Test, keynote talk presented at *IEEE Asian Test Symposium* (*ATS*) (Dec. 2010).
[6] Lewis, D.L. and Lee, H.-H.S.: A Scan-Island Based Design Enabling Prebond Testability in Die-Stacked Microprocessors, *Proc. IEEE In-*

*ternational Test Conference* (*ITC*), pp.1–8 (2007).
[7] Cho, M., Liu, C., Kim, D., Lim, S. and Mukhopadhyay, S.: Design Method and Test Structure to Characterize and Repair TSV Defect Induced Signal Degradation in 3D System, *Proc. International Conference on Computer-Aided Design* (*ICCAD*), pp.694–697 (2010).
[8] Chen, P.-Y., Wu, C.-W. and Kwai, D.-M.: On-Chip Testing of Blind and Open-Sleeve TSVs for 3D IC before Bonding, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.263–268 (2010).
[9] Smith, K., Hanaway, P., Jolley, M., Gleason, R., Strid, E., Daenen, T., Dupas, L., Knuts, B., Marinissen, E.J. and Van Dievel, M.: Evaluation of TSV and Micro-bump Probing for Wide I/O Testing, *Proc. IEEE International Test Conference* (*ITC*), pp.1–10 (2011).
[10] Yaglioglu, O. and Eldridge, B.: Direct Connection and Testing of TSV and Microbump Devices Using NanoPierce$^{TM}$ Contactor for 3D-IC Integration, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.96–101 (2012).
[11] Ohara, Y., Noriki, A., Sakuma, K., Lee, K.-W., Murugesan, M., Bea, J., Yamada, F., Fukushima, T., Tanaka, T. and Koyanagi, M.: 10 $\mu$m Fine Pitch Cu/Sn Micro-Bumps for 3-D Super-Chip Stack, *Proc. IEEE International Conference on 3D System Integration* (*3DIC*), pp.1–6 (2009).
[12] Marinissen, E.J., Verbree, J. and Konijnenburg, M.: A Structured and Scalable Test Access Architecture for TSV-Based 3D Stacked ICs, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.269–274 (2010).
[13] Noia, B. and Chakrabarty, K.: Pre-Bond Probing of TSVs in 3D Stacked ICs, *Proc. IEEE International Test Conference* (*ITC*), pp.1–10 (2011).
[14] Huang, L.-R., Huang, S.-Y., Sunter, S., Tsai, K.-H. and Cheng, W.-T.: Oscillation-Based Pre-Bond TSV Test, *IEEE Trans. Computer-Aided Design*, Vol.32, No.9, pp.1440–1444 (2012).
[15] 45nm Predictive Technology Model, available from ⟨http://ptm.asu.edu/⟩.
[16] Nangate 45nm Open Cell Library, available from ⟨http://www.nangate.com/openlibrary⟩.
[17] Bernstein, K., Frank, D.J., Gattiker, A.E., Haensch, W., Ji, B.L., Nassif, S.R., Nowak, E.J., Pearson, D.J. and Rohrer, N.J.: High-performance CMOS variability in the 65-nm regime and beyond, *IBM Journal of Research and Development*, Vol.50, No.4.5, pp.433–449 (2006).
[18] Liao, Y. and Walker, D.: Fault coverage analysis for physically-based CMOS bridging faults at different power supply voltages, *Proc. IEEE International Test Conference* (*ITC*), pp.767–775 (1996).
[19] Renovell, M., Huc, P. and Bertrand, Y.: Bridging Fault Coverage Improvement by Power Supply Control, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.338–343 (1996).
[20] Taouil, M. and Hamdioui, S.: Layer Redundancy Based Yield Improvement for 3D Wafer-to-Wafer Stacked Memories, *Proc. IEEE European Test Symposium* (*ETS*), pp.45–50 (2011).
[21] Noia, B. and Chakrabarty, K.: *Design-for-Test and Test Optimization Techniques for TSV-based 3D Stacked ICs*, Springer (2013).
[22] Jiang, L., Ye, R. and Xu, Q.: Yield Enhancement for 3D-Stacked Memory by Redundancy Sharing Across Die, *Proc. International Conference on Computer-Aided Design* (*ICCAD*), pp.230–234 (2010).
[23] Verbree, J., Marinissen, E.J., Roussel, P. and Velenis, D.: On the Cost-Effectiveness of Matching Repositories of Pre-Tested Wafers for Wafer-to-Wafer 3D Chip Stacking, *Proc. IEEE European Test Symposium* (*ETS*), pp.36–41 (2010).
[24] Taouil, M., Hamdioui, S., Verbree, J. and Marinissen, E.J.: On Maximizing the Compound Yield for 3D Wafer-to-Wafer Stacked ICs, *Proc. IEEE International Test Conference* (*ITC*), pp.1–10 (2010).
[25] Singh, E.: Exploiting Rotational Symmetries for Improved Stacked Yields in W2W 3D-SICs, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.32–37 (2011).
[26] Singh, E.: Impact of Radial Defect Clustering on 3D Stacked IC Yield from Wafer to Wafer Stacking, *Proc. IEEE International Test Conference* (*ITC*), pp.1–7 (2012).
[27] Flottes, M., Pouget, J. and Rouzeyre, B.: Sessionless Test Scheme: Power-Constrained Test Scheduling for System-on-a-Chip, *Proc. IFIP International Conference on Very Large Scale Integration* (*VLSI-SOC*), pp.105–110 (2001).
[28] Phys.org, available from ⟨http://pda.physorg.com/_news170952515.html⟩.
[29] Noia, B., Chakrabarty, K. and Xie, Y.: Test-Wrapper Optimization for Embedded Cores in TSV-Based Three-Dimensional SOC, *Proc. International Conference on Computer Design* (*ICCD*), pp.70–77 (2009).
[30] Noia, B., Chakrabarty, K. and Marinissen, E.J.: Optimization Methods for Post-Bond Die-Internal/External Testing in 3D Stacked ICs, *Proc. IEEE International Test Conference* (*ITC*), pp.1–9 (2010).
[31] Leiserson, C.E. and Saxe, J.B.: Optimizing Synchronous Systems, *Journal of VLSI Computing Systems*, pp.41–67 (1983).
[32] Noia, B. and Chakrabarty, K.: Retiming for Delay Recovery after DfT Insertion on Inter-Die Paths in 3D ICs, *IEEE Trans. Computer-Aided Design* (2013).
[33] Jiang, L., Xu, Q., Chakrabarty, K. and Mak, T.: Layout-Driven Test-Architecture Design and Optimization for 3D SoCs under Pre-Bond Test-Pin-Count Constraint, *Proc. International Conference on Computer-Aided Design* (*ICCAD*), pp.191–196 (2009).
[34] Noia, B., Goel, S.K., Chakrabarty, K., Marinissen, E.J. and Verbree, J.: Test-architecture Optimization for TSV-based 3D Stacked ICs, *Proc. IEEE European Test Symposium* (*ETS*), pp.24–29 (2010).
[35] Deutsch, S., Chakrabarty, K. and Marinissen, E.J.: Uncertainty-Aware Robust Optimization of Test-Access Architectures for 3D Stacked ICs, *Proc. IEEE International Test Conference* (*ITC*), pp.1–10 (2013).
[36] Nicolaidis, M., Pasca, V. and Anghel, L.: Through-Silicon-Via Built-In Self-Repair for Aggressive 3D Integration, *Proc. IEEE International On-Line Testing Workshop* (*IOLTW*), pp.91–96 (2012).
[37] Loi, I., Mitra, S., Lee, T.H., Fujita, S. and Benini, L.: A Low-Overhead Fault Tolerance Scheme for TSV-Based 3D Network on Chip Links, *Proc. International Conference on Computer-Aided Design* (*ICCAD*), pp.598–602 (2008).
[38] Ye, F. and Chakrabarty, K.: TSV Open Defects in 3D Integrated Circuits: Characterization, Test, and Optimal Apare Allocation, *Proc. ACM/IEEE Design Automation Conference* (*DAC*), pp.1024–1030 (2012).
[39] Jiang, L., Ye, F., Xu, Q., Chakrabarty, K. and Eklow, B.: On Effective and Efficient in-Field TSV Repair for Stacked 3D ICs, *Proc. ACM/IEEE Design Automation Conference* (*DAC*), pp.74:1–74:6 (2013).
[40] Huang, Y.-J. and Li, J.-F.: Built-In Self-Repair Scheme for the TSVs in 3-D ICs, *IEEE Trans. Computer-Aided Design*, Vol.31, No.10, pp.1600–1613 (2012).
[41] Van der Plas, G., Limaye, P., Loi, I., Mercha, A., Oprins, H., Torregiani, C., Thijs, S., Linten, D., Stucchi, M., Katti, G., Velenis, D., Cherman, V., Vandevelde, B., Simons, V., Wolf, I.D., Labie, R., Perry, D., Bronckers, S., Minas, N., Cupac, M., Ruythooren, W., Olmen, J.V., Phommahaxay, A., de Potter de ten Broeck, M., Opdebeeck, A., Rakowski, M., Wachter, B.D., Dehan, M., Nelis, M., Agarwal, R., Pullini, A., Angiolini, F., Benini, L., Dehaene, W., Travaly, Y., Beyne, E. and Marchal, P.: Design Issues and Considerations for Low-Cost 3-D TSV IC Technology, *IEEE Journal of Solid-State Circuits*, Vol.46, No.1, pp.293–307 (2011).
[42] Zhao, Y., Khursheed, S. and Al-Hashimi, B.M.: Cost-Effective TSV Grouping for Yield Improvement of 3D-ICs, *Proc. IEEE Asian Test Symposium* (*ATS*), pp.201–206 (2011).
[43] Cho, M., Liu, C., Kim, D.H., Lim, S.K. and Mukhopadhyay, S.: Pre-Bond and Post-Bond Test and Signal Recovery Structure to Characterize and Repair TSV Defect Induced Signal Degradation in 3-D System, *IEEE Trans. Components, Packaging, and Manufacturing Technology*, Vol.1, No.11, pp.1718–1727 (2011).
[44] Sunohara, M., Tokunaga, T., Kurihara, T. and Higashi, M.: Silicon Interposer with TSVs (Through Silicon Vias) and Fine Multilayer Wiring, *Electronic Components and Technology Conference* (*ECTC*), pp.847–852 (2008).
[45] Kumagai, K., Yoneda, Y., Izumino, H., Shimojo, H., Sunohara, M., Kurihara, T., Higashi, M. and Mabuchi, Y.: A Silicon Interposer BGA Package with Cu-filled TSV and Multi-layer Cu-plating Interconnect, *Electronic Components and Technology Conference* (*ECTC*), pp.571–576 (2008).
[46] Goel, S.K., Adham, S., Wang, M.-J., Chen, J.-J., Huang, T.-C., Mehta, A., Lee, F., Chickermane, V., Keller, B., Valind, T. et al.: Test and Debug Strategy for TSMC CoWoS$^{TM}$ Stacking Process based Heterogeneous 3D IC: A Silicon Case Study, *Proc. IEEE International Test Conference* (*ITC*), pp.1–10 (2013).
[47] Chi, C.-C., Wu, C.-W., Wang, M.-J. and Lin, H.-C.: 3D-IC Interconnect Test, Diagnosis, and Repair, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.1–6 (2013).
[48] Huang, S.-Y., Lee, J.-Y., Tsai, K.-H. and Cheng, W.-T.: At-Speed BIST for Interposer Wires Supporting On-the-Spot Diagnosis, *Proc. IEEE International On-Line Testing Symposium* (*IOLTS*), pp.67–72 (2013).
[49] IEEE Standard Test Access Port and Boundary Scan Architecture, *IEEE Std 1149.1-2001* (2001).
[50] Wang, R., Chakrabarty, K. and Eklow, B.: Post-bond Testing of the Silicon Interposer and Micro-bumps in 2.5D ICs, *Proc. IEEE Asian Test Symposium* (*ATS*), pp.147–152 (2013).
[51] Taouil, M., Hamdioui, S., Beenakker, K. and Marinissen, E.J.: Test Cost Analysis for 3D Die-to-Wafer Stacking, *Proc. IEEE Asian Test Symposium* (*ATS*), pp.435–441 (2010).
[52] Chen, Y., Niu, D., Xie, Y. and Chakrabarty, K.: Cost-Effective Integration of Three-dimensional (3D) ICs Emphasizing Testing Cost Analysis, *Proc. International Conference on Computer-Aided Design* (*ICCAD*), pp.471–476 (2010).
[53] Chou, Y.-W., Chen, P.-Y., Lee, M. and Wu, C.-W.: Cost Modeling

and Analysis for Interposer-Based Three-Dimensional IC, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.108–113 (2012).

[54] Agrawal, M. and Chakrabarty, K.: Test-Cost Optimization and Test-Flow Selection for 3D-Stacked ICs, *Proc. IEEE VLSI Test Symposium* (*VTS*), pp.276–281 (2013).

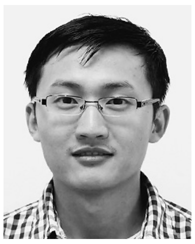
**Krishnendu Chakrabarty** received his B. Tech. degree from the Indian Institute of Technology, Kharagpur, in 1990, and M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively. He is now the William H. Younger Distinguished Professor of Engineering in the Department of Electrical and Computer Engineering and Professor of Computer Science at Duke University. In addition, he serves as the Executive Director of Graduate Studies in Electrical and Computer Engineering. Professor Chakrabarty is a recipient of the National Science Foundation Early Faculty (CAREER) award, the Office of Naval Research Young Investigator award, the Humboldt Research Award from the Alexander von Humboldt Foundation, Germany, and 10 papers awards at major IEEE conferences. Professor Chakrabarty's current research projects include: testing and design-for-testability of integrated circuits; digital microfluidics, biochips, and cyberphysical systems; optimization of digital print and enterprise systems. He has authored 15 books on these topics, published over 480 papers in journals and refereed conference proceedings, and given over 210 invited, keynote, and plenary talks. He has also presented 30 tutorials at major international conferences. Prof. Chakrabarty is a Fellow of ACM, a Fellow of IEEE, and a Golden Core Member of the IEEE Computer Society. He holds two US patents and he has several pending patents. He was a 2009 Invitational Fellow of the Japan Society for the Promotion of Science (JSPS). He is a recipient of the 2008 Duke University Graduate School Dean's Award for excellence in mentoring, and the 2010 Capers and Marion McDonald Award for Excellence in Mentoring and Advising, Pratt School of Engineering, Duke University. He served as a Distinguished Visitor of the IEEE Computer Society during 2005–2007 and 2010–2012, and as a Distinguished Lecturer of the IEEE Circuits and Systems Society during 2006–2007 and 2012–2013. Currently he serves as an ACM Distinguished Speaker. Professor Chakrabarty served as the Editor-in-Chief of *IEEE Design & Test of Computers* during 2010–2012. Currently he serves as the Editor-in-Chief of *ACM Journal on Emerging Technologies in Computing Systems*. He is also an Associate Editor of *IEEE Transactions on Computers* and *IEEE Transactions on Biomedical Circuits and Systems*. He serves on the Steering Committee of *IEEE Transactions on VLSI Systems* and *IEEE Journal of Exploratory Solid-State Computational Devices*, and as an Editor of the *Journal of Electronic Testing: Theory and Applications* (JETTA).

**Mukesh Agrawal** received his B. Tech and M. Tech degrees in Computer Science and Engineering from the Indian Institute of Technology, Kharagpur, in 2008. He worked for Microsoft Corporation as a Software Design Engineer for two years. He is now a Ph.D. candidate in the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA. His research interests include computer architecture, design-for-test, and test-optimization techniques for many-core designs.

**Sergej Deutsch** received his Diplom Degree in Electrical Engineering (eq. to M.S.) from the University of Technology Braunschweig (TU-BS), Germany, in 2011. As a TU-BS student, he was a recipient of a scholarship from The German National Academic Foundation. Deutsch is currently pursuing a Ph.D. degree in Electrical Engineering at Duke University in Durham, NC, USA. His research focuses on testing and design-for-testability of 3D-stacked ICs. Deutsch has published several conference papers and was a recipient of best paper award at the IEEE Asian Test Symposium in 2012.

**Brandon Noia** received his B.S.E. degree in biomedical engineering and electrical and computer engineering from Duke University in 2008. He is currently a Ph.D. candidate in electrical and computer engineering from Duke University. Brandon earned an SRC/Global Research Collaboration Master's Scholarship in 2008 to work in the area of 3D test. In 2010, he was awarded an SRC Graduate Research Fellowship to continue his work. Brandon earned 2nd place in the ACM DAC Student Research Competition in 2012. He won the Best Oral Presentation at the Duke ECE Graduate Research Workshop in 2012, as well as the Best in Session Award at TECHCON in 2012 for his work on pre-bond TSV probing. He also received the ACM SIGDA Turing Celebration Travel Scholarship in 2012 and was a Design Automation Conference Young Student Support recipient in 2008. Brandon has 13 conference and 5 journal publications, including publications in IET Computers and Digital Techniques, IEEE Transactions on Computer-Aided Design, and the Journal of Electronic Testing: Theory and Applications. He has presented his work at conferences across the world, including the North Atlantic Test Workshop, the IEEE International Conference on Computer Design, the IEEE European Test Symposium, the IEEE Asian Test Symposium, the IEEE International Test Conference, the IEEE VLSI Test Symposium, and the IEEE International 3D System Integration Conference. Brandon has three patents pending, covering research in 3D test areas such as pre-bond known-good-die test and 3D retiming flows. He is co-author of a book through Springer titled *Design-for-Test and Test Optimization Techniques for TSV-based 3D Stacked ICs*.

**Ran Wang** received his B.Sci. degree from Zhejiang University, Hangzhou, China, in 2012. He is currently pursuing the Ph.D. degree in Electrical Engineering at Duke University in Durham, NC, USA. His research focuses on testing and design-for-testability of 2.5D ICs.



**Fangming Ye** received his B.Eng. degree from Fudan University, Shanghai, China, in 2009, and the M.S.E. degree from the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA, in 2011. He is currently pursuing the Ph.D. degree at Duke University. He was an intern with Cadence Design Systems, Endicott, NY and Huawei Technologies, Santa Clara, CA, USA. His current research interests include fault modeling, optimization algorithms, machine learning, 3D ICs, board-level diagnosis, and fault-tolerant and reconfigurable design for digital system.

(Invited by Editor-in-Chief: *Hiroyuki Tomiyama*)