Architecture of Ring-based Redundant TSV for Clustered Faults

Wei-Hen Lo, Kang Chi, TingTing Hwang
Department of Computer Science, National Tsing Hua University, R.O.C
turtleevil_1@hotmail.com, tingting@cs.nthu.edu.tw

Abstract

Three-dimensional Integrated Circuits (3D-ICs) that employ the Through-Silicon Vias (TSVs) vertically stacking multiple dies provide many benefits, such as high density, high bandwidth, low-power. However, the fabrication and bonding of TSVs may fail because of many factors, such as the winding level of the thinned wafers, the surface roughness and cleaness of silicon dies, and bonding technology. To improve the yield of 3D-ICs, many redundant TSV architectures were proposed to repair 3D-ICs with faulty TSVs. These methods reroute siganls of faulty TSVs to other regular or redundant TSVs. In practice, the faulty TSVs may cluster because of imperfect bonding technology. To resolve the problem of clustered TSV faults, router-based [1] redundant TSV architecture was the first paper proposed to pay attention to this clustering problem. Their method enables faulty TSVs to be repaired by redundant TSVs that are farther apart. However, for some rarely occurring defective patterns, their method consumes too much area. In this paper, we propose a ring-based redundant TSV architecture to utilize the area more efficiently as well as to maintain high yield. Simulation results show that for a given number of TSVs (8×8) and TSV failure rate (1%), our design achieves 54% *area reduction of MUXes per signal, while the yield of our ring-based redundant TSV architectures can still maintain* 98.47% *to* 99.00% *as compared with router-based desgin [1]. Furthermore, the minimum shifting length of our ring-based redundant TSV architecture is at most* 1 *which guarantees the minimum timing overhead of each signal.*

I. Introduction

Three-dimensional integrated circuits (*3D IC*s) have been proposed as an effective solution to overcome scaling bottleneck. 3D-ICs stack multiple dies and link them together with Through-Silicon Vias (TSVs). Due to shorter vertical interconnect path by TSVs, it results in lower parasitic losses, reduced power consumption, higher I/O density and improved system performance. In spite of above benefits, TSVs may fail during the assembly process and cause the circuits failure [7]–[9]. The types of TSV defects include the misalignment of TSVs and bond pads [14], [15], random open defects [19], impurity of TSVs [20], bond pads short, leak and delaminating [21].

To avoid chip failure caused by defects of TSVs, several redundant TSV designs have been proposed. The basic idea of these methods is to add some redundant TSVs. Thus, the faulty TSVs can be replaced by these redundant TSVs. In [10], the idea has been realized in 3D DRAM memory where every four signal TSVs and two redundant TSVs are allocated as a group. Each signal TSV is connected to a MUX to reroute the signal to neighboring signal TSV or the redundant TSV. In [5], Hsieh et al. proposed a redundant TSV design based on the concept of TSV-chain. Their method adds only one redundant TSV in each TSV block and uses *2-to-1* MUXes to chain the signal TSVs and the redundant TSV together. The signals can be shifted to their neighboring TSVs following the direction towards the redundant TSV. In [11], a redundant row or column of TSVs is added for fault tolerance. However, the previous work is all based on the assumption that TSV faults are independent to each other and follow the uniform distribution. When there are multiple TSV faults clustering together, these previous redundant TSV architecture may not be able to repair TSV faults. In practice, the main causes of clustered TSV defects include the winding level of thinned wafer, the surface roughness and cleaness of silicaon dies and bonding technology.

Li Jiang et al. [1] addressed this fault clustering problem and proposed a router-based TSV redundant architecture for fault tolerance. Their design can not only repair the faulty TSVs by their neighboring signal TSVs but also by other distant signal TSVs. To achieve this goal, their design needs a lot of routers each of which consists of three *3-to-1* MUXes in the TSV block. These routers allow the signals to bypass the routers and to be rerouted to distant TSVs. Therefore, their method can repair almost all combination of clustered TSV faults. However, in order to repair all possible TSV defective patterns, each signal TSV is placed with a router. Therefore, the routers cause a lot of area overhead in the TSV block. Moreover, the space between TSVs may be widened by these routers, hence the probability of clustered TSV faults is less than expected.

In this paper, we propose a new redundant TSV architecture to repair clustered TSV faults. One feature of our architecture is to maintain high yield rate with much less hardware overhead. In addition, the minimum shifting length in our design is always *1* which guarantees the minimum timing overhead of each signal. Our design mainly divides the TSVs in the TSV block to multiple rings. The signals located in these rings can be shifted in the direction of their own rings and their outer rings. The redundant TSVs are placed in four corners of the TSV block or any other locations of the outermost ring. Moreover, the number of redundant TSVs can be adjusted depending on the number of faulty TSVs to be repaired.

The rest of this paper is organized as follows. In Section II, we will analyze the fault rate of clustered TSV faults and show that some complicated clustered TSV defects rarely happen. Next, in Section III, our ring-based architecture for TSV redundancy is introduced. Section IV presents the experimental results for various hypothetical 3D-ICs. Finally, the conclusions are given in Section V.

II. Motivation

In view of the fact that the main bonding steps including the winding level of thinned wafers, the surface roughness and cleanness of silicon dies, cause faulty TSVs to be close to each other, Li Jiang et al [1]. have proposed a router-based redundant TSV repairing framework to solve this clustered TSV faults problem. In Li's work, each signal TSV is placed near a router. If one signal is disconnected due to a TSV fault, the router can reroute the signal to a neighboring or a distant fault-free TSV. Redundant TSVs are placed near the right and bottom boundary of the TSV block. Their method is able to repair multiple heavy-clustered faulty TSVs. Assume there are 4×4 signal TSVs and *8* redundant TSVs in the TSV block. Their router-based design is shown in Figure 1, where white circles , grey circles, black squares, octagons represent signal TSVs, redundant TSVs, signals that are to pass to different tiers and routers (switches), respectively. The routers consist of *3 3-to-1* MUXes. The signals can be shifted to their neighboring TSVs or bypass the routers. Figure 1 shows one of the most difficult defective patterns where five clustered faulty TSVs are denoted as *X*. The arrows represents the repair paths from signals to the TSVs. A repair path means that the signal is shifted to a target TSV. For example, signal *a* is rerouted to TSV *b* by passing two routers *r1*, *r2*.

This router-based redundant architecture is effective in recovering multiple heavily clustered faulty TSVs. However, the probability of occurrence of such defective pattern is negligible. Our analysis is described as follows. Let the yield model *Compound Poisson Distribution* [2] which is commonly used in IC manufacture [22] be adopted to model the clustered defects. Figure 2 shows the probability of occurrence of different number of faulty

978-3-9815370-4-8/DATE15/©2015 EDAA

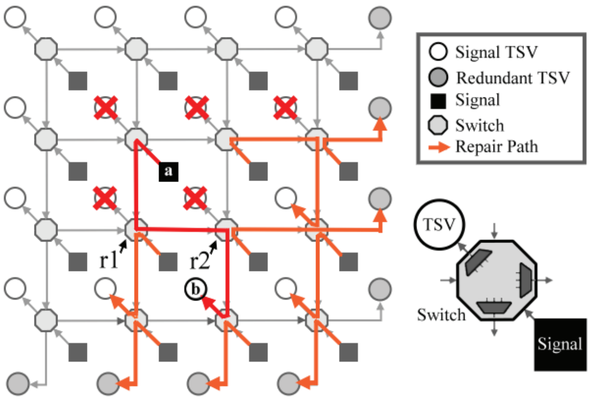


Fig. 1. Example of router-based TSV repair framework repairing a 5-faulty-TSVs case

TSVs where the parameter controlling clustering effect (denoted as α) varies from *0.5* to *5*. Note that smaller values of α indicate increased fault clustering. Based on our analysis, the probability of cases that have less than *4* faulty TSVs reaches *92.65%* to *96.41%* for all α values based on *Compound Poisson Distribution*. In other words, the probability is only about *5%* for the cases that have more than *3* faulty TSVs.

Moreover, based on the probability of faults generating by *Compound Poisson Distribution*, the fault distribution on the plane quantitated by Pareto Distribution [17] shows that the difficult defective patterns are very rare to happen. The analysis is shown as follows.

Let the defective probability is inverse proportional to the area. That is, our assumption is that the bounding box area consisting of faulty TSVs follows *Pareto Distribution* [17]. Hence, different bounding boxes consisting of faulty TSVs represent different defective patterns. Mathematically, a *Pareto Distribution* of area region consisting of *N* clustering faulty TSVs with parameter α is defined by:

$$Probability\{X > B\} = \begin{cases} (B_m/B)^{\alpha}, & x \geq B_m, \\ 1, & x < B_m, \end{cases} \quad (1)$$

where B_m is the (necessarily positive) minimum possible value of *X*, *B* represents the bounding box consisting of *N* faulty TSVs and α is constant. Equation (1) means that the bounding box consisting of *N* faulty TSVs must be larger than a minimum area B_m and the probability of a bounding box larger than *B* is inversely proportional to *B*. The parameter α controls the degree of the *heavy-tailed* effect, where large α means that the bounding box consisting of *N* faulty TSVs are more likely to be close to *B*. This model has also been used in modeling the memory traffic [23], [24].

To understand the occurrence of the most difficult patterns, we randomly generated *100000* bounding boxes consisting of *5* faulty TSVs whose area following *Pareto Distribution* with $\alpha = 1$ and $B_m = 3$. We set $B_m = 3$ since the minimum bounding box area consisting of *5* faulty TSVs equals to *3* (2×3). Figure 3 shows the statistics result where there are *32%* of bounding boxes equal to *3*. That is, if we assume the probability of occurrence of *5* faulty TSVs by *Compound Poisson Distribution* is shown as the first row in Figure 2, the probability of defective pattern, i.e. the most difficult patterns for repairing (the most compact patterns) shown in Figure 1 is approximately $0.91\% \cdot 32\% = 0.291\%$. Therefore, spending a lot of hardware resources such as routers and redundant TSVs to repair this type of defective pattern is inefficient. In addition, the space between TSVs may be widened by those routers. This may alleviate the clustering effect caused by the imperfect bonding process.

Due to the above mentioned reason, it is less efficient to use a lot of hardware to simply repair a defective pattern occuring less than *0.291%* as shown in Figure 1. To solve this problem, we design a new TSV repair architecture that can repair most of TSVs defective patterns considering clustering effect with both less the area overhead and the number of redundant TSVs.

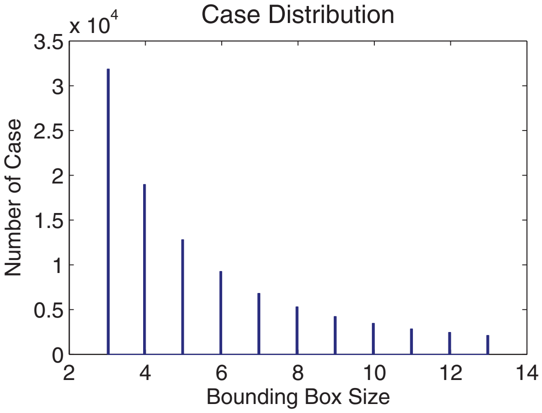


Fig. 3. 100000 Bounding boxes consisted of 5 faulty TSVs following pareto distribution

III. Proposed TSV Redundancy Architecture

Since some patterns of clustered TSV defects as shown in Figure 1 rarely happen, it is inefficient to spend a lot of hardware resources to fix those faults. In order to reduce hardware usage and still maintain high faulty TSV repairing rate considering clustering effect, we propose a new ring-based TSV redundancy architecture that considers the most common TSVs defects patterns.

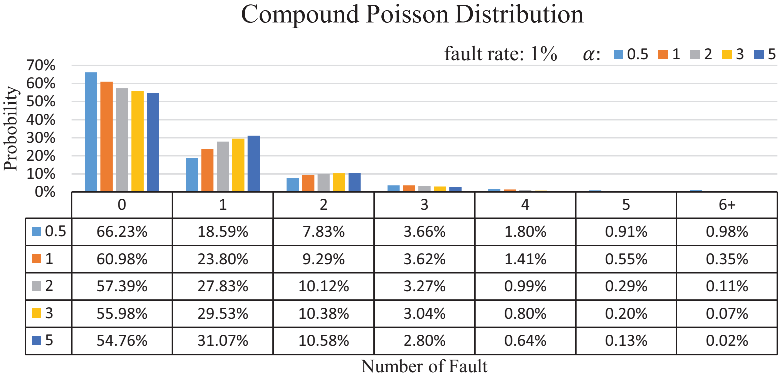
A. Ring-based TSV Redundancy Architecture Design

The overall structure of our ring-based TSV redundancy architecture is illustrated in Figure 4, where there has 8×8 TSVs. Note that the size of TSV block is not restricted to 8×8. Our design can be extended to arbitrary sizes of TSV blocks. In Figure 4, assume the probability of *5* or more than *5* TSVs defects is low enough, our design places *4* redundant TSVs in *4* corners to recover at most *4* faulty TSVs. The TSVs in the TSV block are divided into multiple rings in our design. Take Figure 4 as an example, where 8×8 TSVs are divided into *3* rings. They are the *first ring*, the *second ring*, and the *third ring*. Other TSVs are categorized as the *innermost TSVs*. The *innermost TSVs* will form either a ring or a straight line depending on the size of TSVs in a block. For example, in a 3×8 TSV block, the innermost TSVs will form a line.

In Figure 4, first, the signals of TSVs in the *first ring* can be shifted to their neighboring TSVs in any direction. In other words, these signals can be shifted to neighboring TSVs in the *first ring* clockwise and counterclockwise, and to neighboring TSVs in the *second ring*. Next, the signals of the TSVs in the *second ring* can be shifted to the neighboring TSVs in the *first ring* and the neighboring TSVs in the same ring **clockwise** while the signals of TSVs in the *third ring* can be shifted to the neighboring TSVs in the *second ring* and the neighboring TSVs in the same ring **counterclockwise**. Finally, the signals of the *innermost TSVs* can be shifted to neighboring TSVs in *innermost TSVs* and its neighboring TSVs in the *third ring*.

The detailed interconnections of the outermost signals i.e. the *first ring* in this example and their TSVs are shown in the lower right corner of Figure 5. Every TSV in the outermost ring is connected to a *4-to-1* MUX whose inputs are from its own signal and other three neighboring signals. These MUXes are connected to an e-fuse array which can be programmed by a scan-chain to define a connection. By default, all signals are set to the value which represents the connection to its own TSV. After the testing for TSVs is done, the MUXes will receive appropriate controlling values to reroute the signals. This allows signals to be shifted to their neighboring TSVs or their own TSV. There are three directions for signal rerouting. One is to reroute the signal to the TSVs in the inner rings, and the other two are to reroute the signals to their neighboring TSVs clockwise or counterclockwise.

On the other hand, the detailed interconnection of the inner signals and their TSVs is illustrated in Figure 6. We divide the inner TSVs into



	0	1	2	3	4	5	6+
0.5	66.23%	18.59%	7.83%	3.66%	1.80%	0.91%	0.98%
1	60.98%	23.80%	9.29%	3.62%	1.41%	0.55%	0.35%
2	57.39%	27.83%	10.12%	3.27%	0.99%	0.29%	0.11%
3	55.98%	29.53%	10.38%	3.04%	0.80%	0.20%	0.07%
5	54.76%	31.07%	10.58%	2.80%	0.64%	0.13%	0.02%

Fig. 2. Probability of occurrence of different number of defective TSVs

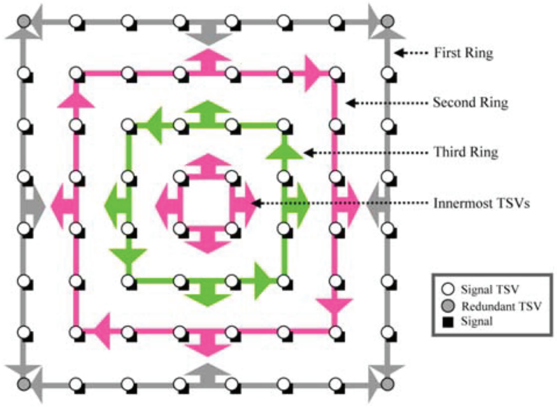


Fig. 4. Ring-based TSV redundancy design

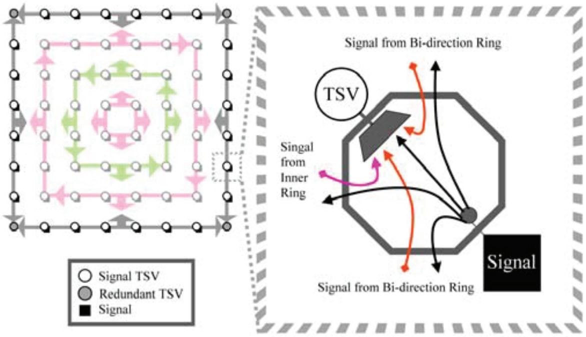


Fig. 5. Detailed interconnection of outermost TSVs

several layers of rings. The directions for signal shifting of each layer is either clockwise or counterclockwise alternatively. In Figure 6, the shifting direction of the outer ring (*second ring*) is clockwise while the shifting direction of the next layer of the ring (*third ring*) is counterclockwise. The signals can also be shifted to their neighboring TSVs which are located in the next layer of the ring. First, we consider the second outermost ring. The structure of this ring is different from that of other inner rings. Take two signals and their TSVs in the dashed rectangle in Figure 6 as an example. The TSV which is located at the corner is connected a *3-to-1* MUX whose inputs are its own signal, signal from the same ring and signal from the outer ring. On the other hand, the TSV which is not located at the corner is connected to a *4-to-1* MUX whose inputs are its own signal, the signal from the same ring, the signal from inner ring and the signal from outer ring. As Figure 6 shows, the signal can be shifted to the neighboring TSVs in two different directions. One is to reroute the signal to the neighboring TSV in the next layer of the ring and the other is to reroute the signal to the neighboring TSV in the same ring clockwise. Second, we consider all other inner rings. In Figure 6, only the *third ring* is in this category. In this category, a TSV is connected to a *3-to-1* MUX whose inputs are its own signal, signal from the same ring, and signal from inner ring. Note that only the TSVs in the *first ring* and *second ring* except for its *4* corners are connected to *4-to-1* MUXes. All other TSVs in the inner rings are connected to *3-to-1* MUXes.

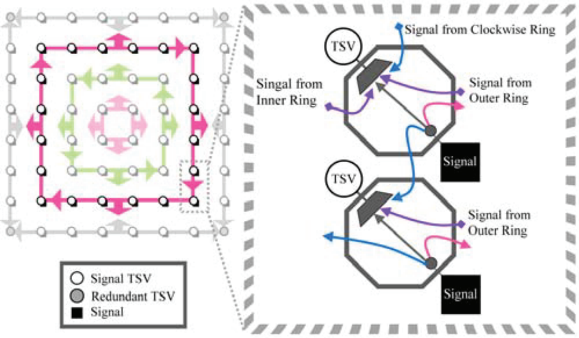


Fig. 6. Detailed interconnection of TSVs in inner rings

As for the most inner signals and TSVs, we simply let the signals reroute to the next layer of the ring and the neighboring TSVs which are also *innermost TSV*. Once there is a faulty TSV inside the *innermost TSVs*, the signal of that faulty TSV can be shifted to the next layer or the other *innermost TSV*. It follows that the signals in the next layer are also shifted to the next next layer and finally the shifting process propagates the signal to one of the redundant TSVs in the corners.

Figure 7 shows an example where white circles represent regular signal TSVs, grey circles redundant TSVs, black squares signals, the crosses on regular signal TSVs faulty TSVs, the small arrows the directions for signal rerouting, and the large blue arrows the repair sequence. We assume that the size of TSV block is 8×8 and there are *4* clustered faulty TSVs forming an *L* shape. The signals of those faulty TSVs are shifted to their neighbors TSVs following the direction of the arrows. This causes all signals on the paths of arrows to be shifted. Shifting continues till the last signals are shifted to redundant TSVs in *4* corners.

Our ring-based design can be applied to TSV block whose size is large enough to form a ring. Figure 8 shows another example of our ring-based architecture for an 8×9 TSV block and *8* redundant TSVs evenly distributed in the outermost ring. In this example, we also show how *8* clustered TSV faults are repaired. The repairing sequence is denoted as those large blue arrows.

As to the hardware overhead, our analysis is as follows. Let the size of TSV block be $N \times M$ and the number of redundant TSVs in the outermost ring be *R*. Our design needs $2 \times (N + M - 2) - R$ *4-to-1* MUXes in the

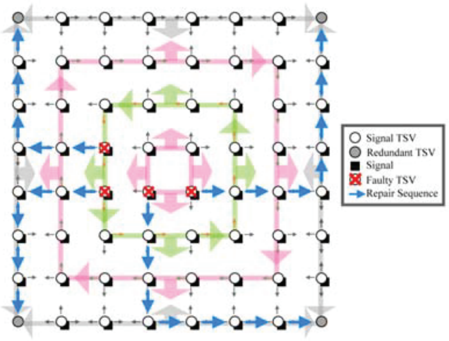


Fig. 7. Example of our ring-based design

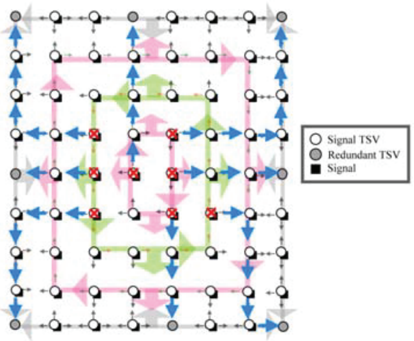


Fig. 8. Example of our ring-based design with *8* redundant TSVs when the size of TSV block is 8×9

outermost ring, $2\times((N-2)+(M-2)-2)-4$ *4-to-1* MUXes in the first layer of the inner rings and $(N-2)\times(M-2)-(2\times((N-2)+(M-2)-2)-4)$ *3-to-1* MUXes in the rest of inner rings. The redundant TSVs in the outermost ring require *4 2-to-1* MUXes and *R - 4 3-to-1* MUXes.

For example, let the size of TSV block be 8×8 and the number of redundant TSV *4*. In this case, the TSVs in the first ring and in the second ring except its *4* corners need *40 4-to-1* MUXes and the other TSVs need *20 3-to-1* MUXes. The redundant TSVs need *4 2-to-1* MUXes.

B. Analysis of Nonrepairable Defect Patterns and Recovery Rate

In this section, we will show that our ring-based design can repair most defective patterns. Assume the number of TSV faults follows *Compound Poisson Distribution* and let the size of TSV block be 8×8, α *0.5* and the fault rate of TSV *1%*. Then, the probabilities of *0* faulty TSV, *1* faulty TSV, and *2* faulty TSVs are *66.23%*, *18.59%*, and *7.83%*, respectively. First, if the number of faulty TSVs is less than *3*, the signals can be rerouted to their neighboring TSVs since there are at least *3* directions for signal rerouting in our design. Therefore, our ring-based design guarantees to repair all defective patterns with less than *3* faulty TSVs. Now for the case of *3* faulty TSVs, the probability is *3.66%*. Let us only consider the most compact cases, i.e. the most difficult cases for repairing as shown in Figure 9. If the bounding boxes of 2×2 consisting of *3* faulty TSVs follow *Pareto Distribution*, the probability of the most compact cases is *39%*. Figure 9(a)-(d) shows all *4* cases. Among these cases with the most compact faulty TSVs, cases demonstrated in Figure 9(a) (*25%*) can not be repaired by our design because all three possible routing directions of a signal are all blocked by faulty TSVs, while cases demonstrated in Figure 9(b)-(d) are repairable (*75%*). Therefore, the probability of nonrepairable defective patterns with *3* TSV faults and bounding box of 2×2 is $3.66\%\times 39\%\times25\% = 0.356\%$. If the bounding boxes consisting of *3* faulty TSVs are larger than 2×2, our ring-based architecture guarantees to repair all these defective patterns. It is because the three routing directions of a signal will not be all blocked in these cases.

Next, for the case of *4* faulty TSVs, its probability is *1.8%*. Then, the probability of most compact and nonrepairable *4* faulty TSVs is $1.8\%\times39\% = 0.702\%$. These defective patterns are nonrepairable because the three routing directions of every signal inside the bounding box are blocked. If the bounding box consisting of *4* faulty TSVs is larger than 2×2, we pessimistically assume that these defective patterns consist of *1* single TSV located at one corner and the other three TSVs located at the diagonal corner whose bounding box is 2×2. Note that this is a pessimistic assumption, because many of defective patterns consist of loosely distributed faulty TSVs. Those loosely distributed faulty TSVs can be repaired by our design. By this pessimistic assumption and the previous deduction of *3* most compact faulty TSV, at most *25%* of the most compact cases of *3* faulty TSVs are nonrepairable in our ring-based design. Hence, the probability of *4* nonrepairable faulty TSVs whose bounding box larger than 2×2 is $1.8\% \times (100 - 39)\% \times 25\% = 0.275\%$.

Finally, assume our ring-based design only use *4* redundant TSVs. Then, the defective patterns with more than *4* faulty TSVs are all nonrepairable. The probability of *5* or more than *5* faulty TSVs is *1.89%*. In summary, the upper bound of the overall failure rate is $0.356\% + 0.702\% + 0.275\% + 1.89\% = 3.223\%$. Moreover, this failure rate is pessimistic since we pick the parameter α *0.5*. This small α value results in high probability of large number of faulty TSVs and more defective patterns of heavily clustered multiple faulty TSVs. An analysis on tighter bound will be found in our journal version.

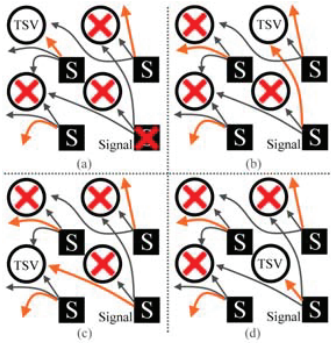


Fig. 9. All 4 combination of most compact cases of 3 faulty TSVs

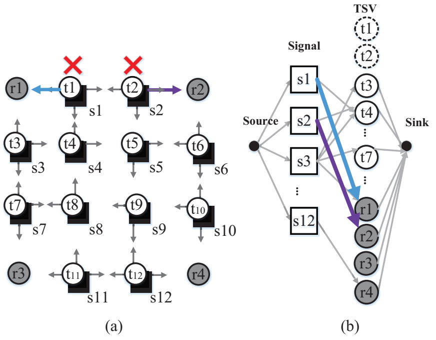
C. Repairing Algorithm

After the fault map is derived from TSV testing, we need to analyze whether the TSV block is repairable, and generate the repair sequences if possible. In [1], repairing problem was transformed into a maximum length bounded flow problem and a diagonal-grouping heuristic was proposed to derive their repair paths to meet length constraint. In our design, since the signal can only be shifted to its neighboring TSVs, the problem is simpler than the previous work. We employ the Maximum Flow method [6] to find our solutions. Our problem formulation is as follows:

Let the flow framework denoted as *G(V,E)*, where node set $V = \{Source\} \cup \{Signal\} \cup \{TSV\} \cup \{Sink\}$ and edge set $E = \{IE\} \cup \{AE\} \cup \{OE\}$. Node *Source* and *Sink* represent the source and sink node of the flow network, respectively. Nodes in *Signal* represent the set of signals, and nodes in *TSV* includes two sets, *STSV* and *RTSV* where *STSV* and *RTSV* are the sets of regular signal TSVs and redundant TSVs. *IE*, *AE*, and *OE* represent sets of incoming edges, assignment edge, outgoing edges. Incoming edge are constructed from *Source* to *Signal*, and outgoing edge are constructed from *TSV* to node *Sink*.

An assignment edge $e_{ij} \in AE$ is constructed if a node $v_i \in S$ connects a node $v_j \in STSV \cup RTSV$ that is not faulty. The capacity of incoming edges, assignment edges and outgoing edges are all set to *1*. After the graph is constructed, the max flow is computed. If the maximum flow equals to the number of signals, then this TSV block is repairable.

Figure 10 shows an example where Figure 10(a) is a TSV block and Figure 10(b) its flow graph. Let the size of TSV block be 4×4 with *4* redundant TSVs at *4* corners. In Figure 10(b), nodes *s1* to *s12* represent signals, nodes *t1* to *t12* regular signal TSVs, and nodes *r1* to *r4* redundant TSVs. Assume *t1* and *t2* are faulty TSVs as shown in Figure 10(a). The edges *(s1,t1)*, *(s1,t2)*, *(s2,t1)*, *(s2,t2)*, *(s4,t1)*, *(t1, Sink)* and *(t2, Sink)* are all removed from *G*. Edges are only constructed between signals and non-faulty TSVs. For example, *s3* can use fault-free TSVs *r1*, *t3*, *t4* and *t7*. Hence, four assignment edges, *(s3,r1)*, *(s3,t3)*, *(s3,t4)* and *(s3,t7)* are constructed. On the other hand, since *t1* and *t2* are faulty, signal *s1* can pass through only fault-free TSVs *t4* and *r1*. Hence, only edges from node *s1* to nodes *t4* and *r1* are constructed. After performing Maximum Flow algorithm, the edges which connect *s1* and *r1*, *s2* and *r2* have flow *1*. This means that signal *s1* is shifted to *r1* and signal *s2* is shifted to *r2*. Since the total capacity in *G* equals to *12*, this defective pattern is repairable.

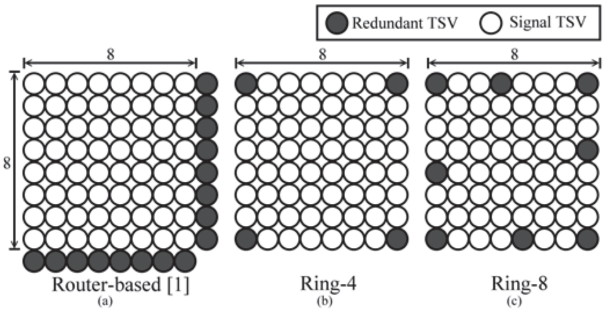


(a) (b)

Fig. 10. Flow Graph of the TSV Block

IV. Experimental Results

We compare our ring-based architecture and router-based redundant TSV architecture [1]. Assume there are 8×8 TSVs. Router-based redundant TSV structure uses *16* redundant TSVs located at the right column and bottom row as shown in Figure 11(a). Redundant/Signal TSVs ratio (R/S ratio) of router-based structure is *1:4*. In our design, we consider two kinds of ring-based redundant TSV structures: *Ring-4* and *Ring-8*. *Ring-4* stands for our ring-based redundant TSV architecture using *4* redundant TSVs at *4* corners as shown in Figure 11(b). *Ring-8* adopts the same basic structure of *Ring-4* but replaces a signal TSV by redundant TSV at each row and column as shown in Figure 11(c). The R/S ratio of *Ring-4* and *Ring-8* are *1:15* and *1:7*.



(a) (b) (c)

Fig. 11. Overview of TSV block

The number of signal TSVs in router-based design is *64*, while the number of signal TSVs in *Ring-4* and *Ring-8* are *60* and *56*, respectively. The number of redundant TSVs is *16* in router-based architecture while our ring-based architecture only uses *4* and *8* redundant TSVs, respectively. Note that it is flexible to set any number of redundant TSVs in our ring-based design. We can simply put those redundant TSVs in any position of the outermost ring of a TSV block.

A. *Comparison of Hardware Cost*

In our experiment, the technology parameters are based on Nangate *45*nm Open Cell Library [18]. The area of *2-to-1* MUX is $1.33 \times 1.4 = 1.862\mu m^2$ in Nangate *45*nm Open Cell Library. The *3-to-1* MUX and *4-to-1* MUX consist of *2* and *3* *2-to-1* MUXes, respectively. Therefore, the area of *3-to-1* MUX and *4-to-1* MUX is set to $2.66 \times 1.4 = 3.724\mu m^2$ and $3.99 \times 1.4 = 5.586\mu m^2$.

Table I shows the number of TSVs, the number and the area of MUXes of the router-based redundant TSV architecture and our ring-based redundant TSV architecture. The column denoted as *Router-based* [1] represents the results of router-based architecture. The columns denoted as *Ring-4* and *Ring-8* represent the results of our ring-based architectures. The rows labeled *# Signal TSV*, *# Redundant TSV*, *R/S ratio*, *2-to-1 MUXes*, *3-to-1 MUXes*, *4-to-1 MUXes*, *Area of 2-to-1 MUXes*, *Area of 3-to-1 MUXes*, *Area of 4-to-1 MUXes*, *Total Area of MUXes* and *Total Area of MUXes / # Signals* are the number of regular signal TSVs, the number of redundant TSVs, the ratio of redundant TSVs to signal TSVs, the total number of *2-to-1* MUXes, the total number of *3-to-1* MUXes, the total number of *4-to-1* MUXes, total area of *2-to-1* MUXes, total area of *3-to-1* MUXes, total area of *4-to-1* MUXes, total area of all MUXes, and the ratio of total area of MUXes to the number of signals, respectively. In our experiments, a TSV block contains at most *64* signal TSVs. For router-based architecture, the number of signal TSVs is set to *64*. For our ring-based architecture *Ring-4* and *Ring-8*, the number of signal TSVs equals to *60* and *56*, and the redundant TSVs are placed in *4* corners and borders of TSV block. Since the numbers of signal TSVs with different architecture are not the same, we compare the *R/S Ratio* (or named *redundancy ratio*) of different architectures. The *R/S Ratio* of router-based architecture is *25%* while *R/S Ratio* of *Ring-4* and *Ring-8* are *6.67%* and *14.29%*. This means the number of required redundant TSVs in our ring-based design is much less than the router-based design. Next, we will compare the number of multiplexer needed. There are *192 3-to-1 MUXes* and no other types of MUXes used in router-based architecture. In our ring-based architecture, *Ring-4* uses *4 2-to-1* MUXes, *20 3-to-1* MUXes and *40 4-to-1* MUXes while *Ring-8* uses *4 2-to-1* MUXes, *24 3-to-1* MUXes and *36 4-to-1* MUXes. It is clear that our designs significantly reduce the usage of *3-to-1* MUXes with some usage of *2-to-1* MUXes and *4-to-1* MUXes. The total area of MUXes used in router-based architecture is $715\mu m^2$ while the total area of MUXes used in *Ring-4* and *Ring-8* is $305.3\mu m^2$ and $297.9\mu m^2$. The ratio of total area of MUXes to the number of signals in router-based architecture is *11.17* while that in *Ring-4* and *Ring-8* is *5.08* and *5.32*, which is only *45.2%* compared to router-based architecture.

TABLE I
Number of TSVs and Number of MUXes of Router-based Architecture and Our Ring-based Architecture

	Router-based[1]	*Ring-4*	*Ring-8*
# Signal TSV	64	60	56
# Redundant TSV	16	4	8
# R/S ratio	1:4 (25%)	1:15 (6.67%)	1:7 (14.29%)
# *2-to-1* MUXes	0	4	4
# *3-to-1* MUXes	192	20	24
# *4-to-1* MUXes	0	40	36
Area of *2-to-1* MUXes	0	7.4	7.4
Area of *3-to-1* MUXes	715	74.5	89.4
Area of *4-to-1* MUXes	0	223.4	201.1
Total Area of MUXes	715	305.3	297.9
Total Area of MUXes / # Signal	11.17	5.08	5.32

B. *Recovery Rate Analysis*

In this Section, the relation between the number of TSVs in each TSV block and recovery rate is analyzed based on the probabilistic models. We first define the fault rate of TSVs as the expected value of the faulty TSVs. That is, if the fault rate of TSVs is *1%* and the number of TSVs is *64*, the expected value of the faulty TSVs will become $64 \times 1\% = 0.64$. Then, we make two assumptions:

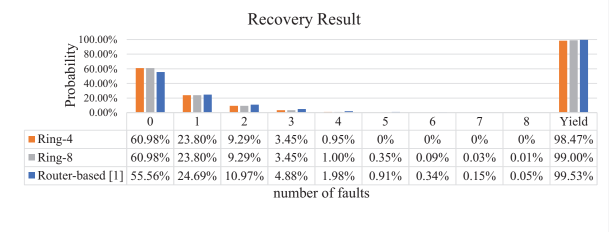
1) The number of faulty TSVs follows Compound Poisson Distribution,
2) The bounding box consisting of *N* faulty TSVs follows *Pareto Distribution*.

Our experiment is developed on 3.0 GHz Linux environment with 64 GB memory. We simulated the proposed architecture using C++/STL programming language with LEDA library [13]. Two sets of experiments are evaluated for router-based architecture and our ring-based architecture.

The parameter α of *Compound Poisson Distribution* is set to *1*. The fault rate of TSVs is set to *1%* and the parameter α of *Perato Distribution* is set to *1*. The simulation flow is as follows :

First, *1,000,000* random cases were generated under *Compound Poisson Distribution*. After the number of faulty TSVs of a random case is decided, we randomly generated the locations of those faulty TSVs whose bounding box follows *Pareto Distribution*. For example, if the probability of *3* faulty TSVs is *4.88%*, there should be near $1000000 \times 4.88\% = 48800$ random cases with *3* faulty TSVs. Then, the bounding box of these faulty TSVs follows *Pareto Distribution* (very likely to be 2×2). In the end, the locations of these faulty TSVs are randomly generated within that bounding box. After the faulty cases are prepared, they will be simulated by LEDA *Maximum Flow* function to test if they can be repaired or not.

For router-based architecture, 8×8 singal TSVs and *16* redundant TSVs are set. As for our ring-based architectures, *Ring-4* and *Ring-8*, the numbers of signal TSVs are *60* and *56* while the numbers of redundant TSVs are *4* and *8*. Figure 12 shows the yields for three different settings. The row denoted as *Router-based* [1] represents the results of router-based architecture and rows denoted as *Ring-4* and *Ring-8* represent the results of our ring-based architectures. The columns labeled *0* to *8* represent the probabilities of different number of faulty TSVs. Figure 12 shows that the router-based architecture can repair almost all cases with faulty TSVs fewer than *5* and achieve overall yield rate *99.53%*. On the other hand, our ring-based architecture *Ring-4* can repair at most *4* faulty TSVs. However, since it is less likely to have more than *4* faulty TSVs and some defect patterns of *3* faulty TSVs and *4* faulty TSVs can still be repaired by our ring-based architecture, the overall yield rate still reaches *98.47%*. As for our ring-based architecture *Ring-8*, the overall yield rate improves to *99.00%* because it can repair up to *8* faulty TSVs. In summary, our ring-based architecture reduces a lot of hardware resources compared to router-based architecture but still maintains high yield rate.



	0	1	2	3	4	5	6	7	8	Yield
Ring-4	60.98%	23.80%	9.29%	3.45%	0.95%	0%	0%	0%	0%	98.47%
Ring-8	60.98%	23.80%	9.29%	3.45%	1.00%	0.35%	0.09%	0.03%	0.01%	99.00%
Router-based [1]	55.56%	24.69%	10.97%	4.88%	1.98%	0.91%	0.34%	0.15%	0.05%	99.53%

Fig. 12. Recovery Rate Table

C. *Shifting Length Analysis*

In our architecture, a signal is shifted at most one position while in router-based architecture, a signal may have a long repairing path. The length of the repairing path changes the length of path delay. In this section, we compare the shifting length of signals with different types of redundant TSV architectures. Table II shows the comparison between our ring-based architectures and the router-based architecture. The column denoted as *#Faults* represents the number of TSV faults in our test cases. The columns denoted as *Max. shifting length* represents the maximum shifting length of signals. The maximum shifting length in our ring-based designs is always *1* because our designs only allow the signals being shifted to their neighboring TSVs. This guarantees the minimum timing overhead of each shifted signal. On the contrary, the router-based architecture can not guarantee the timing overhead of a certain signal since it allows the signals bypassing the neighboring TSVs to remote TSVs. As a result, when a signal is on a timing critical path and is shifted to a remote TSV, it may lead to timing violations. Table II also shows when the number of TSV faults grows, the maximum shifting length also grows in router-based architecture while it remains *1* in our ring-based architectures.

V. Conclusion

In this paper, we design a ring-based redundant TSV architecture with efficient area cost to repair clustered faulty TSVs. The defective patterns of faulty TSVs are modeled as bounding boxes consisting of the faulty TSVs. Assume that the size of these bounding boxes follows *Pareto Distribution*. The probabilities of different defective patterns are derived. Based on the defect model, our ring-based redundant TSV architecture only recovers the most likely defective patterns. Simulation results show that for a given number of TSVs (8×8), TSV failure rate (1%), careful selection of grouping ratios, our design achieves *54%* area reduction of MUXes per signal, while the yield of our ring-based redundant TSV architectures can still maintain *98.47%* to *99.00%*. The minimum shifting length of our ring-based redundant TSV architecture is at most *1* which guarantees the minimum timing overhead of each signal.

TABLE II
Comparison Results of Total Shifting Length and Maximum Shifting Length

#Faults	*Max. shifting length*	
	Ring-4 and Ring-8	*Router-based*[1]
2	1	1
3	1	2
4	1	2
5	1	2
6	1	3
7	1	3
8	1	3

References

[1] L. Jiang, Q. Xu, B. Eklow, "On effective TSV repair for 3D-stacked ICs," *DATE'12*, pp. 793-798, March 2012.

[2] I. Koren and Z. Koren, "Defect tolerance in VLSI circuits: techniques and yield analysis," Proc. of the IEEE, 86(9):18191838, 1998.

[3] Y. Zhao, S. Khursheed, B. M. Al-Hashimi, "Cost-Effective TSV Grouping for Yield Improvement of 3D-ICs," *ATS '11*, pp. 201-206, November 2011.

[4] U. Kang, et al. "8 Gb 3-D DDR3 DRAM using through-silicon-via technology," *IEEE Journal of Solid-State Circuits*, 45(1):111119, January. 2010.

[5] A. Hsieh, T. Hwang, M. Chan, M. Tsai, C. Tseng, H. Li, "TSV Redundancy: Architecture and Design Issues in 3D IC," *DATE'10*, pp. 166-171, March 2012.

[6] A.V. Goldberg and S. Rao. "Beyond the flow decomposition barrier," *Journal of the ACM*, 45(5):783797, 1998.

[7] R. Agarwal, W. Zhang, P. Limaye, R. Labie, B. Dimcic, A. Phommahaxay, and P. Soussan, "Cu/Sn Microbumps Interconnect for 3D TSV Chip Stacking", *Proceedings of Electronic Components and Technology Conference (ECTC'10)*, pp. 858-863, 2010.

[8] N. Lin, J. Miao, P. Dixit, "Void formation over limiting current density and impurity analysis of TSV fabricated by constant-current pulse-reverse modulation," *In Microelectronics Reliability*, 2013.

[9] J.U. Knickerbocker, et al. "Three-dimensional silicon integration. IBM Journal of Research and Development", 52(6):553569, November 2008.

[10] U. Kang, et al. "8 Gb 3-D DDR3 DRAM using through-silicon-via technology. IEEE Journal of Solid-State Circuits", 45(1):111119, Jan. 2010.

[11] I. Loi, et al. "A low-overhead fault tolerance scheme for TSV-based 3D network on chip links", In Proc. Intl Conf. on Computer-Aided Design, pp. 598602, 2008.

[12] D. H. Kim, S. Kim, S. K. Lin, "Impact of nano-scale through-silicon vias on the quality of today and future 3D IC designs", In Proc. SLiP, pp. 1-8, June 2011.

[13] "http://www.algorithmic-solutions.com", *LEDA Library*

[14] R. Patti, "Three-Dimensional Integrated Circuits and the Future of System-on-Chip Designs," Proc. of the IEEE, vol. 84, no. 6, June 2006.

[15] A. W. Topol, J. D. C. La Tulipe, L. Shi, et al., "Three Dimensional Integrated Circuits," IBM Journal of Research and Development, vol. 50, no. 4/5, pp. 491-506, July/Sepetember 2006.

[16] C. H. Stapper, F. M. Armstrong, and K. Saji, "Integrated circuit yield statistics," Proc. IEEE, vol. 71, pp.453 - 470, 1983.

[17] B. C. Arnold, "Pareto Distributions," International Co-operative Publishing House, 1983

[18] Nangate, "The Nangate 45nm Open Cell Library," http://www.nangate.com.

[19] L. Jiang, Y. Liu, L. Duan, Y. Xie, and Q. Xu, "Modeling TSV open defects in 3D-stacked DRAM," *ITC'10*, pp. 1-9, November 2010.

[20] N. Lin, J. Miao, P. Dixit, "Void formation over limiting current density and impurity analysis of TSV fabricated by constant-current pulse-reverse modulation," Microelectronics Reliability, vol. 53, pp. 1943-1953, 2013.

[21] K. H. Lu, S. Ryu, Q. Zhao, X. Zhang, J. Im, R. Huang, and P. S. Ho, "Thermal Stress Induced Delamination of Through Silicon Vias in 3-D Interconnects," *ECTC'10*, pp. 40-45, June 2010.

[22] Murphy, B.T., "Cost-Size Optima of Monolithic Integrated Circuits," *Proc. IEEE no. 12 vol. 52*, pp. 1537–1545, 1964

[23] B. J. Ho, B. Nader, "A Generic Traffic Model for On-Chip Interconnection Networks," *The First International Workshop on Networks-on-Chip Architectures*, 2009

[24] Y. Kim, D. Han, O. Mutlu, M. Harchol-Balter, "ATLAS: A scalable and high-performance scheduling algorithm for multiple memory controllers," *HPCA'10*, pp. 1-12, January 2010.