Volume 8, No. 5, May-June 2017
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

Recovery of TSV Based 3D IC

Sudeep Ghosh
Department of Information Technology
Gurunanak Institute of Technology
Kolkata, India

Mandira Banik
Department Of Computer Science & Engineering
Gurunanak Institute of Technology
Kolkata, India

Tridib Chakraborty
Department of Information Technology
Gurunanak Institute of Technology
Kolkata,India

Abstract: In recent years Through-silicon-via (TSV) based 3D integrated circuit (3D IC) has emerged as an important research area. But due to some manufacturing defects there may have some defected TSVs. To solve this problem it is needed to replace faulty TSVs by redundant TSVs. Allocation of redundant TSVs to a group of functional TSVs is an attracting solution to recover TSVs faults. In this paper we have addressed an approach to allocate redundant TSVs to a group of functional TSVs and to replace defected functional TSVs by the redundant TSV

Key words: Recovery, Redundant TSVs, Multiple dependencies.

1. INTRODUCTION

Three dimension integrated circuit (3D IC) is gaining significance attention in semiconductor industry. A 3D IC is formed by different active device layers stacked one above another and these different layers are connected through a vertical connector known as through silicon via (TSV). TSVs are used as vertical inter connector so that they can act as a single device to achieve improved performance like reducing power consumption, smaller footprint, shorter wire length and heterogeneous integration than two dimensional integrated circuit (2D IC) [1]. During manufacturing of 3D IC different types of errors can occur in TSVs [2]. TSVs defect may reduce the yield because a single TSV defect may paralyze the whole chip, so a recovery mechanism is very much essential to overcome TSV faults and to make the chip functional.

To increase the yield of 3D IC, use of redundant TSV is considered as an attractive solution. A redundant TSV is used to replace a faulty functional TSV so that signal can be rerouted through redundant TSV from lower die to upper die. One of the methods to implement the redundant TSV is to form a group of functional and redundant TSVs. The idea of grouping of functional and redundant TSVs is presented in [3, 4]. In [5], it has been shown that the best group ratio of functional and redundant TSV is created depending on available numbers of multiplexers (MUXs). But in [5] the wire length required to reroute the signal has not been considered. Hence, we focus on providing higher repair capability. The work is also based on utilizing redundant TSVs. Regular and redundant TSVs are partitioned into groups using a specified group ratio (regular-to-redundant), where each group can have multiple spare TSVs and multiplexers are used to reroute the signals through a redundant TSV in case defective TSVs exist in that group. Due to the allocation of multiple TSVs in a group, the repair capability is significantly increased. Also in our work, a group may have more than one redundant TSV; so we use inner spiral search algorithm to find out the nearest redundant TSV for replacing faulty functional TSV. In this manner we can reduce the required wire length.

The rest of the paper is organized as follows. Section 2 describes the previous work related to use of redundant TSV. Section 3 describes problem we have considered. Section 4 describes proposed method and an illustrative example is presented in section 5. Section 6 describes experimental results and comparison with others' works and finally section 7 concludes this paper.

2. PRIOR WORK

As of now there is no public data of TSV failure rate. This failure rate varies due to various parameters like different foundries, maturity of TSV technology, height/width, pitch size etc. But TSV process technology has advanced significantly in recent years.

To improve 3D memory product, Samsung presented [6] a TSV redundancy strategy. In [6] two redundant TSVs and four functional TSVs are clubbed together to form a group. In this method only two faulty TSVs can be replaced as there are only two redundant TSV present in this group.

Hsieh et al. [7] proposed an architecture of TSVs that contains one spare TSV to form a TSV-chain. If there is 'n' number of functional TSVs and one spare TSV, then the redundancy ratio would be 1: n and it can tolerate only one TSV fault. In [8], it is shown that an architecture for uniformly distributed TSV architecture where a grid of all TSVs is formed and if any fault occurs then the signal will be rerouted through nearest TSV of that TSV grid. In [9] NoC link is used for $n^{\times}n$ TSVs grid. Here, to overcome the fault row or column is added. Suppose a redundant column is added, so each spare TSV is added to its corresponding row so that it can repair any one of faulty TSVs of that row.

If there are M numbers of rows or columns added and the number of functional TSVs is N then ratio will be M: N i.e. it can repair M number of TSVs failures in each row or column grid.

In this paper we proposed a uniformly distributed TSV architecture and an algorithm to find out nearest redundant TSV to recover faults.

3. PROBLEM DEFINITION

Consider the TSV grid as a 2D matrix, where each element represents a TSV. Our problem is to find an architecture and corresponding algorithm to find out the faulty TSV and replace it with a redundant unused TSV.

Our proposed algorithm does a linear search through the matrix to find those elements which are faulty TSVs. Then the algorithm does a reverse spiral search with respect to each element found to be faulty. Then the reverse spiral search finds a redundant unused TSV nearest to the faulty one and replaces the faulty one with redundant one.

4. PROPOSED TSV REDUNDANCY ARCHITECTURE

In order to handle independent TSV fault, our solution is to offer more repair options for each defective TSV. In other words we try to increase repair flexibility so that a defective TSV can be replaced by a nearest spare TSV.

4.1. Overall Structure

The proposed architecture is depicted in Fig1.This architecture is linked with TSV pads by switches and wires to construct the TSV grid and redundant TSVs are also linked with functional TSVs. We have taken an example where there is total 192 numbers of regular TSVs and 64 numbers of redundant TSVs. So total 16 groups have been formed and each group contains 12 functional TSVs and 4 redundant TSVs. In this architecture each of functional TSVs is linked with all redundant TSVs. If any fault occurred due to TSV fault, then the signal reroutes through a nearest redundant TSV. To reroute the signal through a redundant TSV, the switch links two pads of faulty TSV to a nearest redundant TSV. This process repeats until all redundant TSVs are used. Like many 3D-SIC designs [13, 14] TSVs are fabricated in a regular manner and grouped as bundles. These uniformly placed TSVs are linked together to construct the proposed TSV redundancy architecture.

4.2. Repair Path Routing

The design of switch depends on the placement of redundant TSVs. Consider Fig. 2 as an example; here redundant TSVs are placed on the middle of the TSV grid. As a result, signals can route to any direction (North, South, East and West). We use Fig. 2 as an example to introduce the concept of repair path and to show its routing capability. If there is any faulty TSV (see the crossed circle) detected, the signal reroutes through a redundant TSV. The connection from a faulty TSV to a redundant TSV is called repair path. (see the dotted arrow).

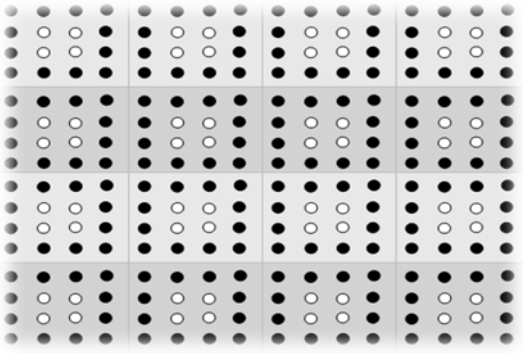

Fig1: Proposed TSV redundancy architecture.

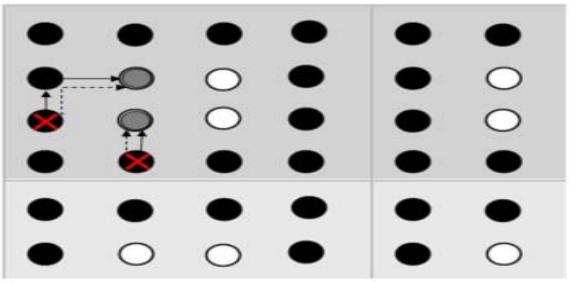

Fig 2: Repairing faulty TSV.

4.3. Formation of Groups

In our proposed algorithm each group will be having maximum 12 functional TSVs and maximum redundant TSVs will be distributed depending upon the total number of redundant TSV available. Suppose there are 35 functional TSVs. So there will be 2 groups having 12 functional TSVs each and another group having 11 TSVs.

4.4. Distribution of Redundant TSV

Distribution of redundant TSV to each group as follows: there are groups containing X, Y and Z numbers of functional TSVs, and M is the total number of redundant TSVs. So these M number of redundant TSVs will be divided into these 3 groups. Total numbers of functional TSVs = X + Y + Z. So Group 1 will get x numbers of redundant TSVs, where x = $(X/S)^{\times}$ S. Where 'S' is total number of functional TSV. Similarly group 2 will get y numbers of redundant TSVs where y = $(Y/S)^{\times}$M. Similarly group 3 will get z numbers redundant TSVs where z = $(Z/S)^{\times}$M.

Suppose there are 30 functional TSVs and 8 numbers of redundant TSVs. So groups will contain 12, 12, and 6 numbers of functional TSVs. Group 1 and group 2 have 3 TSVs each and group 3 will get 2 redundant TSVs.

4.5 Algorithm 1: SearchRedundent(x, y)

Input: T = Top row, L= Left column, R = Right column, B = Bottom row, N= Total number of functional TSVs .x=row number, y=column number, i=0, j=0;
Output: Will find out nearest redundant TSV to replace faulty TSV.
Begin
Initialize variable
T ← x, L ← y, R ← y+1, B ← x+1
Search from west to north-west
i ← x

```
 if (L >= 0 && L < N)
   for (i=x, i>=T && i >=0;i--)
      if  TSV[i][L] == 1
         replacement found at position TSV[i][L]
           TSV[i][L] will be marked as 0
END
Search for North-West to North-East
j ← L+1
  if (T >= 0 && T < N && L > 0)
    for ( j<=R && j<N)
       if (TSV[T][j] == 1)
            replacement found at position TSV[T][j]
              TSV [T][j] will be marked as 0
END
Search for North-east to South-east
if(R < N)
     for (i=T+1; i<=B && i<N; i++)
         if  (TSV [i][R] == 1)
             replacement found at position TSV [i][R]
               TSV[i][R] will be marked as 0
END
Search for South-east to South-west
if (B < N)
    for (j=R-1; j>=L &&  j>=0; j--)
        if (TSV[B][j] == 1)
            replacement found at TSV[B][j]
              TSV[B][j] will be marked as 0
END
Search for South-west to West
 if (L >= 0 && L < N)
     for (i=B -1;i>=x+1 && i<N; i--)
         if (TSV[i][L] == 1)
             replacement found at position TSV[i][L]
               TSV[i][L] will be marked as 0
END
 T ← T+ 1, L ← L +1, R ← R + 1, B ← B + 1 // Increasing
search area by 1 both side  Until ( flag=1).
```

4.6 Calculation of Multiplexer and dependency

Let there are 'X' numbers of redundant TSVs and 'Y' numbers of functional TSVs. So Y-to-1 MUXs are needed for each redundant TSV. Hence X numbers of Y-to-1 MUXs are needed to connect all redundant TSVs to functional redundant TSVs. Each Y-to-1 MUXs can be implemented by (Y-1) numbers of 2-to-1 MUXs. So the total numbers of 2-to-1 MUXs needed at input side is $X^{*}(Y-1)$.

At the output end, for each functional TSV each MUX has X+1 input line .For Y number of functional TSVs ,Y numbers of (X+1)-to-1 MUXs are required. Now the number of equivalent 2-to-1 MUXs is $Y^{*}((X+1)-1) = XY$.

So the total Number MUX needed = $X^{*}(Y-1) + XY = X(2Y-1)$

Table 1: MUX and dependency calculation

Functional TSV (Y)	Redundant TSV (X)	Total MUX need X(2Y-1)	Group ratio Y:X	Dependency
12	4	92	One 12:4	48
20	8	152	One 12:4,one 8:4	80
24	8	184	Two 12:4	96
48	16	368	Four 12:4	192
54	18	384	Four 12:4,one 6:2	204
56	20	444	Four 12:4,one 8:4	224
60	20	460	Five 12:4	240
100	34	570	Six 12:4,One 4:2	296
120	40	920	Ten 12:4	480
130	44	996	Ten 12:4,one 10:4	520
150	50	1126	Twelve 12:4,One 6:2	588
200	68	1532	Sixteen 12:4,One 8:4	800

5. ILLUSTRATIVE EXAMPLE

Our proposed algorithm is illustrated with an example presented in Fig. 4. Here a 2D matrix representing the TSV structure. For each iteration one faulty TSV is detected and SearchRedundant(x,y) function is called. x and y are the row number and column number of the faulty TSVs respectively . The function does a reverse spiral search corresponding to the faulty TSV, finds a redundant TSV which is nearest and makes a connection through switches, we first assigned some elements with a value 0 (which represents functional TSVs), some elements with a value 1(which represents TSVs those are redundant) and some elements with a value 2(that represents faulty TSVs). Fig 4 shows the initial step.

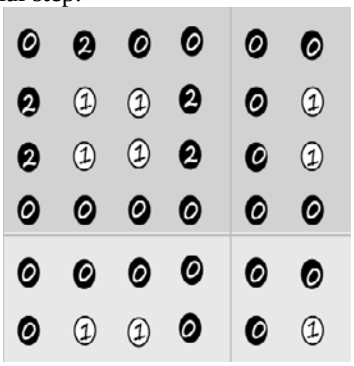

Fig 4:Initial step.

In first iteration faulty TSV (mark as 2) will be replaced by a redundant TSV. As first faulty TSV (mark as 2) present at top row then it will search its right side where it will find a functional TSV and then it will search south side where it will find a redundant TSV and then this redundant TSV will be marked as 0 as showing in Fig 5. The signal of faulty TSV will be rerouted through this redundant TSV. Fig 6 shows a faulty TSV present at left column of TSV grid to find nearest redundant TSV it will search it's right side and it will find a redundant TSV and this redundant will be marked as 0 i.e. replacement found. Fig 7 describes the

replacement of right column's faulty TSVs, to find out redundant TSV it will first search its right which is outside of TSV grid, so it will stop searching to its right side and it will start searching again. It will search its south side and then left sight of its present position and it will find a redundant TSV which will be marked as 0.

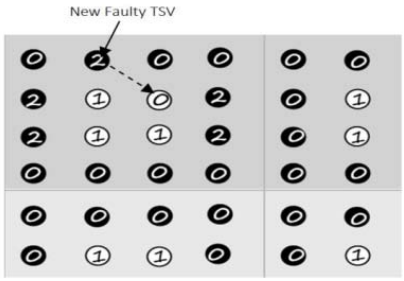

Fig 5: First Iteration

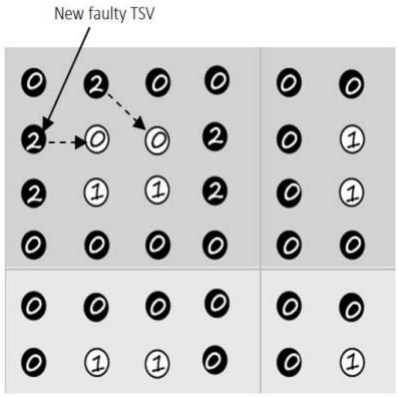

Fig 6: Second Iteration

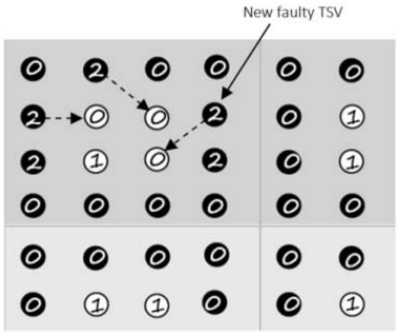

Fig 7: Third Iteration.

6. RESULTS

We have done the experiment by considering a constant grouping ratio (regular to spare TSV ratio) of 12:4 as suggested by our architecture. It is clearly seen from the experiment that with the increase of the number of regular TSVs, the yield percentage reduces gradually. We fix the TSV failure rate in this experiment and it varies 0.001% to 0.005%. The yield can be obtained based on probabilistic model to show yield comparison under Poisson distribution.
Table 2: Yield analysis with varying number of TSV and varying TSV fault.

Regular TSV	1000			100000			1000000			10000000		
TSV Failure rate	.001	.005	.01	.001	.005	.01	.001	.005	.01	.001	.005	.01
Yield (%)	99.99	99.99	99.99	99.99	99.98	98.66	99.99	99.89	96.73	99.999	98.91	71.74
Spare TSV	333			33333			333333			33333333		

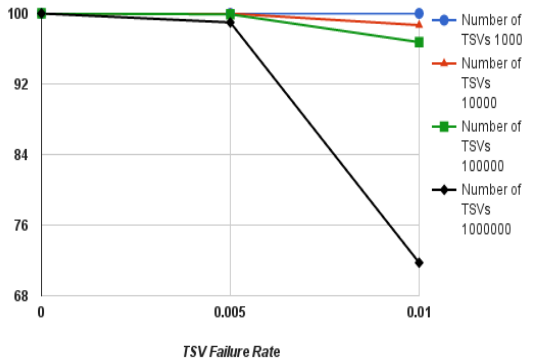

Fig 8 shows that if we increase the number of TSVs then TSV failuar rate decreases.

We have compared our proposed work [5]. In [5] groups are created dynamically on the basis of functional TSVs, redundant TSVs and available MUXs. But in our work we first made the group and then distributed the Redundant TSV according to our proposed architecture. Table 3 shows the required MUXs as per the proposed architecture for different functional and redundant TSVs,

If there are more than two redundant TSVs in a group, in [10] it is not discussed how to replace faulty TSV by a redundant TSV. But in our work we used inner spiral search algorithm to replace faulty TSV by a redundant TSV and this algorithm always replaces the nearest redundant TSV so that the wire length is minimum.

Table 3: Comparison of required MUXs with previous work.

Functional TSV		Redundant TSV		Total MUX need	
[5]	Proposed work	[5]	Our	[5]	Proposed work
20	20	4	4	148	98
100	100	50	50	1450	1118

Wire length calculation:

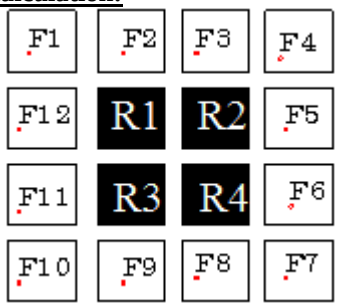

Fig 8: Grid view of TSVs.

This is the grid view of our proposed architecture. The maximum wire length is between F1 to R4 or F4 to R3. If we assume that distance between each block is 50 µm (as taken in the work [10]), then maximum wire length is 141.42 µm i.e. wire length required to connect F1 and R4 is141.42 µm .

Table 4: Comparison of wire length with previous work

Number of Functional TSV	Number of redundant TSV	Total wire length of [10] (µm)	Our method (µm)
10	3	6323	5687
30	10	13970	9712

Table 4 shows total wire length and it is seen that required wire length is less in our work compared to [10,11,12].

7. CONCLUSION

In this paper we have proposed a TSV repair mechanism by using redundant TSVs. Also we have proposed a searching algorithm to find out nearest redundant TSV. In this architecture the number of MUXs has been decreased from existing architecture and hence the cost will be reduce as the number of MUXs is less. As we are replacing the nearest redundant TSV to reroute the signal of faulty functional TSV, so wire-length will be minimum. Also we have increased total dependency of a group. Though we have considered only uniformly distributed TSVs, it will be our future work to implement for non-uniformly distribution of TSVs.

REFERENCES

[1]. R. Weerasekera et al, "Extending Systems-on chip to the Third Dimension: Performance,Cost and Technological Tradeoffs," in Int. Conference on Computer-Aided design, pp.212-219, 2007.

[2]. H. Chen, J.-Y. Shih, S.-W. Li, H.-C. Lin, M.-J. Wang, and C.-N.Peng., "Electrical tests for three-dimensional ics (3dics) with tsvs," in Proc. of 3D Test Workshop Informal Digest, 2010.

[3]. Yi Zhao, Saqib Khursheed, Bashir M. Al-Hashimi, "Cost-Effective TSV Grouping for Yield mprovement of 3D-ICs", in Pro**c. A**sian Test Symposium, pp: 201-206, 2011.

[4]. Ang-Chih Hsieh, TingTing Hwang, Ming-Tung Chang, Chih-Mou Tseng and Hung-Chun Li, "TSV Redundancy: Architecture and Design Issues in 3D IC", in Proc. DATE, pp: 166-171, 2010.

[5]. Surajit Kumar Roy, Sobitri Chatterjee, Chandan Giri and Hafizur Rahaman,"Repairing of Faulty TSVs using Available Number of Multiplexers in 3D ICs", in Proc. Of ASQED, pp 155-160, 2013.

[6]. U. Kang, et al. 8 Gb 3-D DDR3 DRAM using through-silicon-via technology. IEEE Journal of Solid-State Circuits, 45(1):111–119, Jan.2010.A.C.

[7]. Hsieh, et al. TSV redundancy: Architecture and design issues in 3D IC. In Proc. Design, Automation, and Test in Europe Conf. Exhibition,pp. 166–171, 2010.

[8]. Li Jiang, Qiang Xu, and Bill Eklow : On Effective TSV Repair for 3D-Stacked ICs. Design and Automated test in Europe ,2012.

[9]. I. Loi, et al. A low-overhead fault tolerance scheme for TSV-based 3D network on chip links. In Proc. Int'l Conf. on Computer-Aided Design,pp. 598–602, 2008.

[10]. Surajit Kumar Roy, Kaustav Roy, Chandan Giri and Hafizur Rahaman : Recovery of Faulty TSVs in 3D ICs, 16th Int'l Symposium on Quality Electronic Design, pp 533-536,2015.F.

[11]. Ye and Krishnendu Chakrabarty "TSV Open Defects in 3D Integrated Circuits: Characterization, Test and OptimalSpare Allcation", in Proc. DAC, pp: 1024-1030,2012.

[12]. Jing Xie, Yu Wang and Yuan Xie, "Yield-Aware Time-Efficient Testing and Self-fixing Design For TSV-Based 3D ICs" in Proc.Asia and South Pacific Design Automation Conference (ASPDAC),pp: 738 - 743, 2012.

[13]. M. Kawano, et al. A 3D packaging technology for 4 Gbit stacked DRAM with 3 Gbps data transfer. In IEEE Int'l Electron Devices Meeting, pp.1–4, 2006.

[14]. T. Zhang, et al. A customized design of DRAM controller for on-chip 3D DRAM stacking. In Proc. IEEE Conf. on Custom Integrated Circuits, pp. 1–4.

