Using Asymmetric Layer Repair Capability to Reduce the Cost of Yield Enhancement in 3D Stacked Memories

M. Tauseef Rab, Asad A. Bawa, and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78712-1084
Email: {tauseefrab, bawa}@utexas.edu, touba@ece.utexas.edu

Abstract

Three-dimensional (3D) technology makes it possible to organize memories as cell arrays stacked on logic where upper die layers contain the cell arrays and the bottom layer implements the peripheral logic. This creates new degrees of freedom that can be exploited to optimize the use of spare rows/columns to maximize yield. This paper proposes a new idea that exploits an additional degree of freedom that has not previously been utilized which is that the order of the die in the stack can be selected. The cell array dies can be ordered with the one with the most defective cells at the lowest layer, followed by next most defective, and so forth finishing with the die with the fewest defective cells on the top layer. All the cell array dies have identical designs and are manufactured identically. However, the peripheral logic die is designed in a way where it costs less to provide repair on the lower layers than it does on the higher layers of the cell arrays. This is done by limiting the domain over which some spares can be used thereby reducing the number of fuses needed for configuring the spare. Results in the paper show that the ability to skew repair capability across the different layers in a 3DIC allows greater yield enhancement at lower cost both in terms of number of spares and number of fuses.

1. Introduction

Three-dimensional (3D) technology using through silicon vias (TSVs) provides new ways to organize and construct memories. One approach is to use *stacked banks* where each stacked die contains a different bank of memory. This significantly reduces wire-length routing in comparison to a corresponding multi-bank 2D memory. Another exciting new approach that becomes possible is to have *cell arrays stacked on logic* (using the term from [Taouil 11]). In this configuration, the upper die layers contain the cell arrays while the bottom layer implements the peripheral logic (i.e., row decoders, column select logic, sense amplifiers, row buffers, output drivers, etc.). The advantage of isolating the peripheral logic on a separate layer is that different process technologies can be used. For example, cell arrays can be implemented with process technology optimized for density (e.g., NMOS), whereas the peripheral logic can be implemented with process technology optimized for speed (e.g., CMOS). This approach was first commercially used by Tezzaron Semiconductors.

Given the high defect rates in memories, spare rows and columns are typically used to allow for post-manufacturing repair in order to enhance yield [Schuster 78], [Zorian 03]. The memory is tested, and a defect map is generated indicating which cells in the memory are defective. Based on the defect map, the memory is reconfigured to use the spare rows and columns to bypass defective cells [Kuo 87], [Wey 87], [Hemmady 89]. The memory reconfiguration can be done either at manufacture time with fuses, or it can be done with a built-in self-repair (BISR) scheme [Kim 98].

In a conventional single die implementation of a memory, if it is not possible to repair all the defective cells with the available spare rows and columns, then the die is discarded as worthless. In a 3D memory where multiple die are stacked together, the idea of using unused spares in one die to help in repairing another die has been proposed in [Chou 09, 10, 11] and [Jiang 10]. In these approaches, if there are too many defective cells to repair using a die's own intra-die resources, it can borrow unused spares from other die. This is applicable for any of the integration methods, i.e., wafer-to-wafer (W2W), die-to-wafer (D2W), or die-to-die (D2D). However, it is especially powerful for D2W and D2D where the specific die to be stacked together can be selected to optimize overall yield. For example, consider the case where each die contains one spare row and one spare column. With the ability to share spares across die, then dies with 4 defects could be stacked together with dies containing 0 defects, and dies with 3 defects could be stacked with dies containing 1 defect, and so forth. So by categorizing every die in a lot and carefully distributing them among the various 3D stacks, the number of unusable die can be minimized. The schemes in [Chou 09, 10, 11] and [Jiang 10] were conceived for a *stacked banks* type configuration for 3D memory using additional TSVs to share spares across layers, but the same concept could be applied for a

978-1-4673-2658-2/12/$31.00 ©2012 IEEE

cell arrays stacked on logic type configuration also where the row decoders, column select logic, and reconfiguration logic is all located on the bottom layer.

In this paper, a new idea is proposed for further improving yield in 3D memories with reduced cost by using an additional degree of freedom that is not exploited in earlier work which is that the order of the die in the stack can also be selected and optimized. This degree of freedom can be exploited in D2W or D2D *cell arrays stacked on logic type* configurations in the following way. The cell array dies can be ordered with the one with the most defective cells at the lowest layer, followed by next most defective, and so forth finishing with the die with the fewest defective cells on the top layer. All the cell array dies have identical designs and are manufactured identically. However, the peripheral logic die is designed in a way where it costs less to provide repair on the lower layers than it does on the higher layers of the cell arrays. One simple example of this concept is the following. Suppose there are 4 layers of cell arrays, and each cell array die contains one spare column, and for simplicity, there are no spare rows. So there are a total of 4 spare columns. The concept of sharing unused spares among stacked dies used in [Chou 09, 10, 11] and [Jiang 10] could be applied to share the 4 spare columns among all die in the stack which would ensure that any four die with a cumulative total of 4 defects or less could be stacked together and be repaired. However, an alternative with asymmetric layer repair capability would be to dedicate two spares to only be used for the lowest layer (where the die with the most defects can always be placed) while the other two spares could be used for any of the four layers. For each spare column that is dedicated to only be used in one layer (the lowest layer in this example), the space of possible columns that it can be configured to replace is reduced by a factor of 4 (since there are 4 layers in this example) meaning that $\log_2(4)=2$ less fuses would be required to implement the reconfiguration logic for each of those spares to select which column it will replace. Since there are two such dedicated spares in this example, the total number of fuses for asymmetric repair is reduced by 4 compared to symmetric repair. In terms of overall repair capability, the asymmetric repair approach could always handle any stack of four die with a cumulative total of 4 defects or less provided at least one of those die has 0 defects. So if the overall yield of cell array die with 0 defects is expected to be greater than 25%, then there will be at least one die with 0 defects that could be allocated to each stack. Results will be shown later in the paper that the overall yield using asymmetric repair can be effectively equivalent to that of symmetric repair while using fewer fuses, or alternatively for the same number of fuses, the yield can be improved.

While the simple example above illustrated the concept of asymmetric layer repair capability using spares dedicated to certain layers, another efficient way that this concept can be used is with the *selective row partitioning (SRP)* scheme described in [Rab 09]. SRP provides a way to logically segment a single spare column and use it to repair multiple defective cells in multiple other columns. This capability comes at the cost of additional fuses, but if it can be utilized to repair multiple defective cells then the fuse cost per defective cell repaired can be minimized. Using SRP symmetrically for all layers would tend to be inefficient because some layers may have 1 or 0 defective cells thereby completely wasting the extra fuses used to implement SRP. However, the proposed idea here is to use SRP for only one or a few layers and then match up the cell array die having the most defective cells that can most efficiently benefit with the SRP layers to efficiently utilize the SRP capability. This would reduce the number of spares that need to be incorporated in the cell array die thereby reducing the required redundancy to achieve a given defect tolerance. For example, instead of requiring 2 spare columns per cell array die, SRP could be used to achieve the same yield using only 1 spare column per cell array die with little or no increase in the number of fuses. Reducing the cell array size helps reduce area, delay, and power for the overall memory.

2. Asymmetric Spares

Figure 1 shows a block diagram of a 3D multi-layer memory organized as *cell arrays stacked on logic*. All the cell array dies are identical. The spare columns and rows are evenly distributed among the layers. The row decoder, column select logic, and other peripheral logic are located in a separate logic layer. The fuses for configuring the spares to perform repair are also located on the logic layer. To maximize the repair capability, the spares could be used for global inter-die repair. In other words, each spare column (row) could be used to replace any column (row) on any layer. The main cost for this would be the number of fuses needed to configure each column (row) globally across all layers. If there are n layers and c columns (r rows), the number of fuses needed to make each spare column (row) a global spare that can be used to replace any bit line (word line) in any layer would be $log_2[n]$ to select the layer and $log_2[c]$ ($log_2[r]$) to select the bit line (word line). So the cost of a global spare versus the cost of a spare local to one die is the following:

Fuses for global spare = $log_2[c]$ (or $log_2[r]$) + $log_2[n]$

Fuses for local spare = $log_2[c]$ or $log_2[r]$

The proposed idea is to consider allowing some spares to be global and some to be local rather than the conventional symmetric design having all local spares or all global spares. The optimal number of local and global spares and their distribution across the layers depends on the expected distribution of defects. Without loss of generality, consider the case where defects are equally

likely in each cell of each die (i.e., they are not clustered in certain die). Suppose 1000 die are manufactured each containing 2^k bits and the bit error rate is 2^{-k}. Then the expected distribution of defects/die is shown in Table 1. If there are 4 layers and 4 global spares, then the 1000 die can be combined together to construct 250 3D-ICs with the configuration shown in Table 2. As can be seen from Table 2, 15 3DICs would combine one die with 4 defects together with three die having 0 defects. 61 3D-ICs would combine one die with 3 defects, one die with 1 defect, and two die with 0 defects, and so forth.

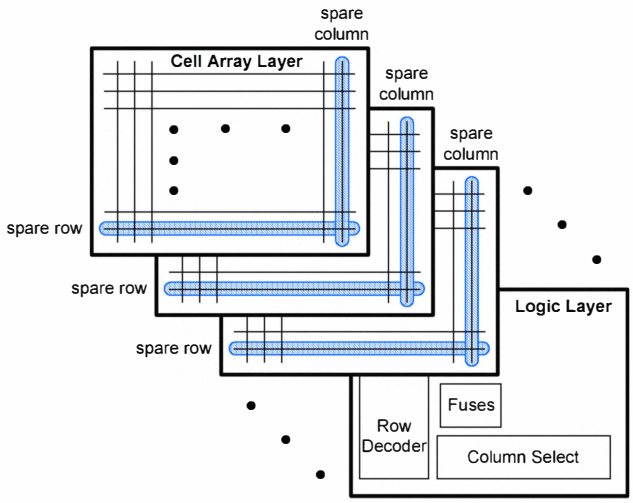

Figure 1. 3D Multi-layer Memory Organized as Cell Arrays Stacked on Logic

Using the proposed approach, we can exploit the degree of freedom of which order the die are stacked. The peripheral logic built on the logic array can be designed so that it always allocates two local spares to the lowest layer and then two global spares that can be used in any layer. The die with the most defects is then always placed in the lowest layer. As can be seen in Table 2, all the configurations that are used have at least one die with 2 or more defects. Thus, the two local spares are fully utilized. Since all configurations have no more than 4 defects in total, the combination of the 2 fully utilized local spares plus the 2 global can repair all the defects in all the configurations. Thus the yield in this case would be identical to using 4 global spares, but the advantage is that the number of fuses is reduced by 4 because each local spare needs $log_2(4\ layers)$ fewer fuses.

Table 1. Expected distribution of defects/die for 1000 die with 2^k bits and bit error rate of 2^{-k}.

Defects/Die	Number of Die
0	370
1	370
2	184
3	61
4	15

Table 2. Way to construct 250 4-layer 3DICs using the 1000 die shown in Table 1.

Num. 3D-ICs	Config.	Defects/Die				
		4	3	2	1	0
15	4-0-0-0	15	0	0	0	45
61	3-1-0-0	0	61	0	61	122
10	2-2-0-0	0	0	20	0	20
145	2-1-1-0	0	0	145	290	145
19	2-1-0-0	0	0	19	19	38
250	**Total**	**15**	**61**	**184**	**370**	**370**

Now consider a second example where 996 die are manufactured each containing 2^k bits and the bit error rate is $1.5x2^{-k}$. The expected distribution of defects/die is shown in Table 3. If there are 6 layers and 2 global spares per die, then 166 3DICs can be constructed using the configurations shown in Table 4.

Using the proposed method, 3 local spares could be allocated for the lowest layer, 2 local spares could be allocated for the second lowest layer, 1 local spare could be allowed for the third lowest layer, and 3 global spares could be used to cover the rest of the defects not covered by the local spares. Thus a total of 9 spares is required with 6 of those being local and only 3 of them being global. This helps to significantly reduce the number of fuses required.

A procedure for allocating local and global spares to minimize the number of spares and number of fuses is given in the next section.

Table 3. Expected distribution of defects/die for 996 die with 2^k bits and bit error rate of $1.5x2^{-k}$.

Defects/Die	Number of Die
0	222
1	334
2	251
3	125
4	47
5	14
6	3

Table 4. Way to construct 166 6-layer 3DICs using the 996 die shown in Table 3.

Num. 3D-ICs	Config.	Defect/Die						
		6	5	4	3	2	1	0
3	6-2-1-0-0-0	3	0	0	0	3	3	9
14	5-2-1-1-0-0	0	14	0	0	14	28	28
47	4-2-2-1-0-0	0	0	47	0	94	47	94
23	3-3-1-1-1-0	0	0	0	46	0	69	23
61	3-2-2-1-1-0	0	0	0	61	122	122	61
11	3-2-1-1-1-1	0	0	0	11	11	44	0
7	3-2-1-1-1-0	0	0	0	7	7	21	7
166	**Total**	**3**	**14**	**47**	**125**	**251**	**305**	**215**

3. Procedure for Allocating Spares

Given an expected defect distribution (i.e., the information shown in Tables 1 and 3), the following procedure can be used to configure the die in the layers to construct 3D-ICs to minimize the total number of spares required and maximize the number of local spares (i.e., to obtain the information shown in Tables 2 and 4).

Step 1: Set the *total-defect-limit* per 3D-IC equal to the number of defects in the die with the most defects.

Step 2: Fill the lowest layer of each of the *n* 3D-ICs with the *n* die having the most defects.

Step 3: For the next lowest layer, fill each of the *n* 3D-ICs from the population of remaining die placing the most defective die in the 3D-IC with the fewest total defects under the constraint that the total defects in any 3D-IC does not exceed the *total-defect-limit*.

Step 4: Step 3 is repeated until all layers are filled at which point the procedure completes. However, if a point is reached where it is impossible to fill a 3D-IC from the population of remaining die without violating the *total-defect-limit* constraint, then the *total-defect-limit* is incremented by 1 and the procedure starts over from Step 2.

Once the procedure above is completed, then the set of possible distributions of defects per layer is known. The final *total-defect-limit* is the total number of spares required. The minimum number of defects in a particular layer is the number of local spares that can be fully used for that layer. If some distributions have 0 defects in some layer, then no local spares can be fully used for that layer. Once the local spares are determined, then the number of global spares is simply the total number of spares minus the number of local spares.

The above procedure gives the minimum total number of spares. However, in some cases some global spares can be converted to local spares (i.e., with no net increase in the total number of spares) while still satisfying the constraints. This arises because the number of spares has to be a whole number, so there can be some slack. So a final post-processing step would be to iterate through each layer and try to convert a global spare to become a local spare for that layer while still satisfying the constraints.

This design procedure described in this section uses enough spares to ensure that all die in the considered distribution can be repaired and utilized (i.e., 100% utilization). If it is acceptable to allow some of the die with the most defects to be discarded (i.e., have less than 100% utilization), then the same procedure can still be used. The input distribution would simply be adjusted based on the desired utilization. For example, if it is acceptable to discard all die with 4 or more defects, then the input distribution for the procedure would only contain the population of die with fewer than 4 defects. Under this condition the procedure would minimize the number of total spares and maximize the number of local spares.

4. Combining with Selective Row Partitioning

Another very efficient way that the proposed concept can be used is with the SRP scheme described in [Rab 09]. The SRP method selectively chooses one or more row address bits to decode thereby partitioning the row address space. The column that is replaced by a particular spare column can be different for each partition of the row address space. This allows a single spare column to repair multiple defects provided each defect exists in a different partition of the row address space. The cost of SRP is that the number of fuses required for configuring the spare column is now multiplied by the number of row address partitions since it can be different in each row addresses partition. For example, suppose the number of columns is c. If SRP was used to decode two row address bits and create 4 partitions of the row address space, then the number of fuses required would be $4\ log_2(c)$. However, if 4 defects are repaired, then the fuse cost is the same as if four separate spare columns were used to repair the four defects each requiring $log_2(c)$ fuses. On the other hand, if only 3 defects are repaired, then the number of fuses is higher for SRP with one spare column compared to using 3 spare columns (i.e., $4\ log_2(c)$ vs. $3\ log_2(c)$). So the efficiency of SRP depends on how much of the maximum repair capability of SRP can be utilized.

With the proposed idea of asymmetric repair, it is possible to use SRP for one layer and then select a die whose defect profile can be most efficiently repaired with SRP. By selective matching up die with layers implementing SRP, the repair capability of the SRP can be efficiently used. When it is possible to always maximally use the SRP repair capability, then there is no additional fuse cost for using SRP. Thus it can effectively

either reduce the total number of spares required for a given yield or enhance the yield for a given number of spares without requiring additional fuses. If SRP's repair capability cannot always be maximally used, then there is some fuse overhead for using SRP compared with using more spares, but it still may be worthwhile. For example, it may be possible to reduce the number of spares included in each cell array die by using SRP at the cost of more fuses. So the tradeoff would be the cost of the additional fuses versus the area, delay, and power reduction resulting from reducing the number of spares on each cell array die.

SRP can be used by first selecting spares using the procedure in Sec. 3. A subset of the local spares in some layer can be replaced by SRP. The smaller the number of partitions that are used by SRP, the higher the utilization of repair capability will be. In Sec. 5, results are shown for two cases. One is where SRP is used only when it will not increase the number of fuses (i.e., its capacity is fully utilized), and the other is where SRP is used more aggressively so that the number of spares per cell array die is reduced at some cost in terms of additional fuses.

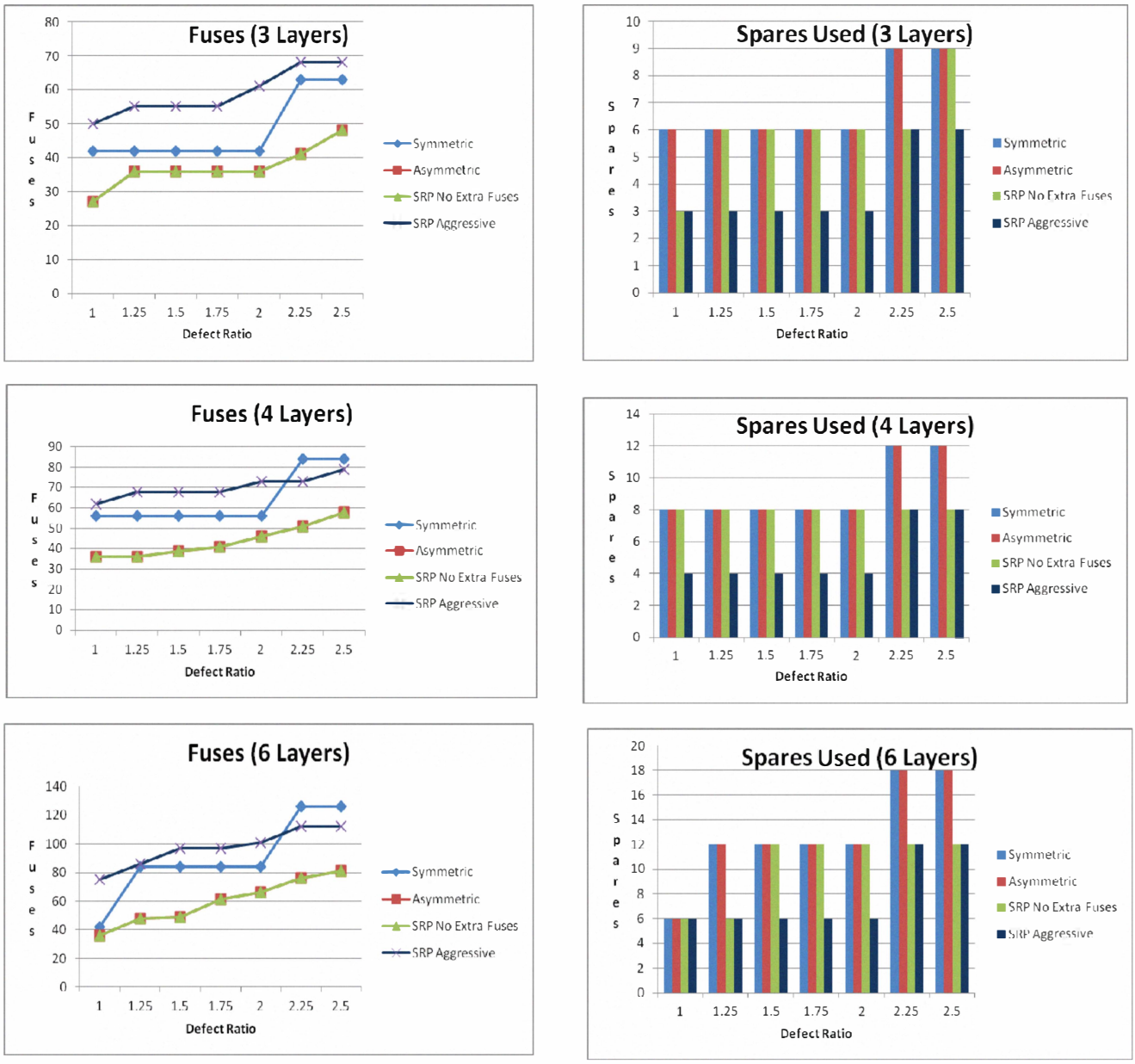

Figure 2. Comparison of Fuse Cost and Number of Spares using Different Methods Targeting Different Defect Ratios

5. Experimental Results

Experiments were performed to evaluate the effectiveness of the proposed asymmetric layer repair approach. Defects were modeled as being uniformly distributed across memory cells and dies with no clustering factor used. This is a conservative model for the proposed method. Greater improvement would be obtained the more clustered the defects are. Simulations were performed targeting different defect ratios where the defect ratio is defined as the average number of defective memory cells per cell array die. Note that the proposed method does not require knowing the exact defect ratio that will exist when the cell array die are manufactured, it simply targets some maximum defect ratio for which it is desired that the repair capability be able to handle.

The results are shown in Fig. 2. Results are plotted for four different methods providing repair capability sufficient for the targeted defect ratio. The first is the conventional symmetric repair where all spares were considered to be global. The second is the proposed asymmetric repair where the procedure in Sec. 3 is used to select local and global spares. The logic layer is designed to implement these spares using the minimum number of fuses. The third method is using the proposed asymmetric repair with SRP only when it does not increase the number of fuses. The fourth method is using the proposed asymmetric repair with aggressive use of SRP sufficient to reduce the number of spares required per cell array die. The results were generated assuming 3 layers, 4 layers, and 6 layers of cell array dies. For each number of layers, there is one graph for the number of fuses per targeted defect ratio and another graph for the total number of spares per targeted defect ratio. Note that the total number of spares is a multiple of the number of layers since each cell array die is identical and contains the same number of spares.

From the results, it can be seen that the number of fuses can be significantly reduced using the proposed asymmetric repair approach. If SRP is used, in some cases the number of spares per cell array die can also be reduced without any increase in the number of fuses (e.g., when the defect ratio is 2.25 or 2.5, one spare per die can be reduced with no increase in the number of fuses). Finally, if SRP is used more aggressively, it is possible to reduce the number of spares per cell array die at the cost of more fuses. Note that the total number of fuses required to reduce the number of spares/die is generally less than the number of fuses required for conventional symmetric SRP.

6. Conclusions

The new concept of asymmetric repair described here is made possible by the degree of freedom in multi-layer 3D-IC memories that the order of the die in the stack can be selected. By matching up die with more defects to layers that have greater repair capability, the number of fuses required to handle a particular defect ratio is significantly reduced. Moreover, the SRP technique from [Rab 09] can be efficiently utilized to also reduce the number of spares per cell array

References

[Chou 09] Y.-F. Chou, D.-M. Kwai, and C.-W. Wu, "Memory Repair by Die Stacking with through Silicon Vias", *Int. Workshop on Memory Technology, Design, and Testing*, pp. 53-58, 2009.

[Chou 10] C.-W. Chou, Y.-J. Huang, and J.-F. Li, "Yield-Enhancement Techniques for 3D Random Access Memories", *Int. Symp. on VLSI Design Automation and Test (VLSI-DAT)*, pp. 104-107, 2010.

[Chou 11] Y.-F. Chou, D.-M. Kwai, and C.-W. Wu, "Yield Enhancement by Bad-Die Recycling and Stacking with Through-Silicon Vias", *IEEE Trans. on VLSI Systems*, Vol. 19, Issue 8, pp. 1346-1356, Aug. 2011.

[Jiang 10] L. Jiang, R. Ye, and Q. Xu, "Yield Enhancement for 3D-Stacked Memory by Redundancy Sharing across Dies," *Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 230-234, 2010.

[Kim 98] I. Kim, Y. Zorian, G. Komoriya, H. Pham, F.P. Higgins, and J.L. Lewandowski, "Built In Self Repair for Embedded High Density SRAM," *Proc. of International Test Conference*, pp. 1112-1119, 1998.

[Kuo 87] S.-Y. Kuo and K. Fuchs, "Efficient Spare Allocation for Reconfigurable Arrays," IEEE Design & Test, Vol. 4, Iss. 1, pp. 24-31, Feb. 1987.

[Rab 09] M.T. Rab, A.A. Bawa, and N.A. Touba, "Improving Mmeory Repair by Selective Row Partitioning,", *Proc. of IEEE Symposium on Defect and Fault Tolerance,* pp. 211-219, 2009.

[Schuster 78] S.E. Shuster, "Multiple word/bit line redundancy for semiconductor memories," *IEEE Journal of Solid-State Circuits*, Vol. SC-13, pp. 698-703, 1978.

[Taouil 11] M. Taouil and S. Hamdioui, "Layer Redundancy Based Yield Improvement for 3D Wafer-to-Wafer Stacked Memories," *European Test Symposium*, pp.45-50, 2011.

[Wey 87] C.-L. Wey and F. Lombardi, "On the Repair of Redundant RAM's", *IEEE Trans. on Computer-Aided Design*, Vol. 6, No. 2, Mar. 1987.

[Zorian 03] Y. Zorian, and S. Skoukourian, "Embedded-Memory Test and Repair: Infrastructure IP for SOC Yield," *IEEE Design & Test of Computers*, Vol. 20, Issue 3, pp. 58-66, May 2003.