Reducing Cost of Yield Enhancement in 3-D Stacked Memories Via Asymmetric Layer Repair Capability

Muhammad Tauseef Rab, Asad Amin Bawa, and Nur A. Touba

***Abstract*— One way to organize 3-D memories is cell arrays stacked on logic where the upper die layers contain the cell arrays and the bottom layer implements the peripheral logic. A new degree of freedom exists when constructing 3-D memories, which is that the order of the die in the stack can be selected. This paper proposes a new idea that exploits this additional degree of freedom to reduce the cost of yield enhancement. In the proposed approach, the cell array die with the most defective cells is placed in the lowest layer, followed by the next most defective cells in the second lowest layer, and so forth finishing with the die with the fewest defective cells on the top layer. The bottommost layer (peripheral logic) is designed such that it costs less to tolerate the defects on the lower layers than it does on 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 show that the asymmetric repair capability created by fine tuning the domain of spares in a 3-D integrated circuit allows yield enhancement at a lower cost in terms of number of spares and fuses.**

***Index Terms*— 3D ICs, memory repair, yield.**

I. Introduction

STACKED memories can be constructed in 3-D integrated circuit (3DIC) using through silicon vias (TSVs) to interconnect multiple layers. One approach is to use *stacked banks*, where each stacked die contains a different bank of memory. This organization offers significant reduction in wire-length routing in comparison with a corresponding multibank 2-D memory. Cell arrays stacked on logic (using the term from [12]) is another exciting new approach that becomes possible because of TSVs where the upper die layers contain the cell arrays and 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., n-MOS), 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. Prebond testing of cell array die has added difficultly, but techniques such as the one described in [7] can be used to probe TSVs.

A new idea is proposed in this paper (preliminary results were presented in [8]), which exploits an additional degree of freedom which is that the order of the die in a 3-D memory stack can be selected and optimized. This creates a degree of freedom that is used to lower the cost or repair and improve yield. This degree of freedom can be exploited in die-to-wafer (D2W) or die-to-die (D2D) cell arrays stacked on logic type configurations in the following way. In the proposed arrangement, the lowest die layer will have the dies with the most defective cells, followed by the next most defective cells in the second lowest layer, 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. The bottom most layer, which has the peripheral logic, is designed such that it costs less to tolerate defects on the lower layers than it would on higher layers of the cell arrays. One simple example of this concept is the following. Suppose there are four layers of cell arrays, and each cell array die contains one spare column, and for simplicity, there are no spare rows. Therefore, there are a total of four spare columns. The concept of sharing unused spares among stacked dies used in [1] and [13] could be applied to share the four spare columns globally among all the dies in the stack, which would ensure that any four die with a cumulative total of four defects or less could be stacked together and be repaired. An alternative with asymmetric layer repair capability, however, 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), whereas 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 four (as there are four layers in this example) meaning that $\log_2(\text{four layers}) = 2$ less fuses would be required to implement the reconfiguration logic for each of those spares to select which column it will replace. Because there are two such dedicated spares in this example, the total number of fuses for asymmetric repair is reduced by four compared with 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 four defects or less provided at least one of those die has zero defects. Therefore, if the overall yield of cell array die with

Manuscript received February 16, 2013; revised July 11, 2013; accepted August 12, 2013. Date of publication September 17, 2013; date of current version August 21, 2014.

The authors are with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712-1084 USA (e-mail: tauseefrab@utexas.edu; bawa@utexas.edu; touba@ece.utexas.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2013.2280593

zero defects is expected to be $>25\%$, then there will be at least one die with zero defects that could be allocated to each stack. Results will be shown later in this 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. Note that each array die is identical, and the proposed method only impacts the design of the reconfiguration logic on the peripheral layer.

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 [9]. 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 used to repair multiple defective cells then the fuse cost per defective cell repaired can be minimized. Using SRP symmetrically for all the layers would tend to be inefficient because some layers may have one or zero defective cells thereby completely wasting the extra fuses used to implement SRP. The proposed idea here is, however, 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 use the SRP capability. This would reduce the number of spares that needs to be incorporated in the cell array die thereby reducing the required redundancy to achieve a given defect tolerance. For example, instead of requiring two spare columns per cell array die, SRP could be used to achieve the same yield using only one spare column per cell array die with a little or no increase in the number of fuses. Reducing the cell array size helps to reduce area, delay, and power for the overall memory.

It should be pointed out that the proposed idea does not increase the number of TSVs needed beyond what is conventionally required for a stacked arrays on logic architecture. The difference is only in how the repair circuitry on the logic layer is designed. Therefore, this approach does not impact yield because of any additional TSVs being added. The idea improves spare management and related fuse costs and does not address the TSV yield management.

This paper enhances the original idea presented in [8] in two major ways. First, the asymmetric definition is expanded. In [8], a spare that is considered a local spare was local to one layer. Here, the definition is generalized to allow spares to be local to any subset of layers of the stack or any set of columns within a layer. These enhancements that give a more general domain for the spares to provide improvements in the results compared with the earlier results.

Second, the results in [8] were based on simulating uniform random defects across all the memory locations, which is a worst case conservative approximation. This simplistic model may not be how the actual defects show up in actual silicon. In this paper, we have also investigated how the results would change if the defects are clustered using a commonly used clustering model.

The rest of the paper is organized in the following manner. Section II reviews work in the 3-D memory space to improve yield. In Section III, the key idea, asymmetric repair in 3-D memories, is presented. In Section IV, an algorithm for optimizing the number of fuses and spares to improve the yield of 3-D memories is presented. In Section V, the domain for asymmetric spares is generalized to consider more fine grain optimization. In Section VI, asymmetric repair is combined with SRP. Section VII talks about the experimental setup and Section VIII has the results for various simulations.

II. Related Work

Given the high defect rates in memories, spare rows, and columns are typically used to allow for postmanufacturing repair to enhance yield [10], [15]. The memory is tested and a defect map is generated, suggesting which cells in the memory are defective. With the defect map, the memory is reconfigured to use the spare rows and columns to bypass defective cells [6], [14], [16]. The memory reconfiguration can be done either at manufacture time with fuses, or it can be done with a built-in self-repair scheme [5].

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 3-D memory where multiple dies are stacked together, the idea of using unused spares in one die to help in repairing another die has been proposed in [2], [3] and [4]. In these approaches, if there are too many defective cells to repair using a die's own intradie resources, it can borrow unused spares from other die. This is applicable for any of the integration methods, i.e., wafer-to-wafer, D2W, or D2D. It is, however, especially powerful for D2W and D2D where the specific die to be stacked together can be selected to optimize the 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 four defects could be stacked together with dies containing zero defects, and dies with three defects could be stacked with dies containing one defect, and so forth. Therefore, by categorizing every die in a lot and carefully distributing them among the various 3-D stacks, the number of unusable die can be minimized. The schemes in [3] and [4] were conceived for a *stacked banks* type configuration for 3-D memory using additional TSVs to share spares across layers, but the same concept could be applied for a cell arrays stacked on logic type configuration also where the row decoders, column select logic, and reconfiguration logic are all located on the bottom layer.

In [1], a more general method for sharing spares across layers was presented for processor memory stacks in which a *stacked banks* type of configuration is used for the memory, but on the processor die, a global spare assignment unit is placed, which can allocate spares across the layers. Reference [13] uses a similar approach and also proposes the use of a spare cylinder, which replaces all the cells in a vertical axis across multiple die.

The asymmetric layer repair method proposed in this paper is for the cell arrays stacked on logic memory architecture

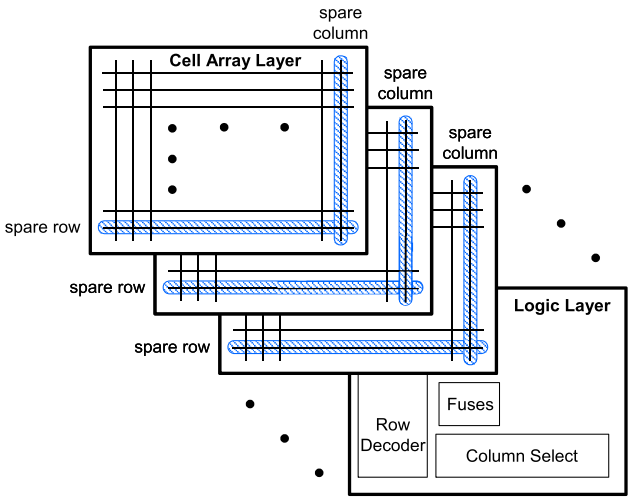


Fig. 1. 3-D multilayer memory organized as cell arrays stacked on logic.

and takes an advantage of the degree of freedom that the order of the die in the stack can be chosen. It differs from the earlier methods in that it optimizes repair configuration logic by exploiting the fact that dies with more defects can be placed in certain layers in the stack, which allows greater yield enhancement at a lower cost both in terms of the number of spares and fuses.

III. Asymmetric Spares

Fig. 1 shows a block diagram of a 3-D multilayer 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 interdie 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 the 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). Thus, the cost of a global spare versus the cost of a spare local to one die is the following:

1) Fuses for global spare $= \log_2[c]$ (or $\log_2[r]$)+ $\log_2[n]$.
2) 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 the local spares or all the global spares. The optimal number of local and global spares and their distribution across the layers depends on the expected distribution of defects. Without the 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

TABLE I
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 II
Way to Construct 250 Four-Layer 3DICs Using the 1000 Die Shown in Table I

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**

in Table I. If there are four layers and four global spares, then the 1000 die can be combined together to construct 250 3DICs with the configuration shown in Table II. As can be seen from Table II, 15 3DICs would combine one die with four defects together with three die having zero defects; 61 3DICs would combine one die with three defects, one die with one defect, and two die with zero defects, and so forth.

Using the proposed approach, we can exploit the degree of freedom of which order the dies 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 II, all the configurations that are used have at least one die with two defects. Thus, the two local spares are fully used. All the configurations have no more than four defects in total, the combination of the two fully used local spares plus the two global spares can repair all the defects in all the configurations. Thus, the yield in this case would be identical to using four global spares, but the advantage is that the number of fuses is reduced by four because each local spare needs $\log_2$(four layers) fewer fuses.

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

Using the proposed method, three local spares could be allocated for the lowest layer, two local spares could be allocated for the second lowest layer, one local spare could be allowed for the third lowest layer, and three global spares could be used to cover the rest of the defects not covered by the local spares. Thus, a total of nine spares are required

TABLE III
EXPECTED DISTRIBUTION OF DEFECTS/DIE FOR 996 DIES WITH 2^k BITS AND BIT ERROR RATE OF 1.5×2^{-k}

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

TABLE IV
WAY TO CONSTRUCT 166 SIX-LAYER 3DICS USING THE 996 DIES SHOWN IN TABLE III

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**

with six of those being local and only three 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 following section.

IV. PROCEDURE FOR ALLOCATING SPARES

Given an expected defect distribution (i.e., the information shown in Tables I and III), the following procedure can be used to configure the die in the layers to construct 3DICs for minimizing the total number of spares required and maximizing the number of local spares (i.e., to obtain the information shown in Tables II and IV).

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

Step 2: Fill the lowest layer of each of the n 3DICs with the n die having the most defects.

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

Step 4: Step 3 is repeated until all the layers are filled at which point the procedure completes. If a point is, however, reached where it is impossible to fill a 3DIC from the population of remaining die without violating the *total defect limit* constraint, then the *total defect limit* is incremented by one 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 of the distributions have zero defects in some layers, 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. In some cases, some of the global spares, however, 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. Thus, a final postprocessing 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 the dies in the considered distribution can be repaired and used (i.e., 100% utilization). If it is acceptable to allow some of the dies 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 four or more defects, then the input distribution for the procedure would only contain the population of die with fewer than four defects. Under this condition, the procedure would minimize the number of total spares and maximize the number of local spares.

V. FINE TUNING DOMAIN OF SPARE

When memories are arranged as cell arrays stacked on logic it allows for a degree of freedom, which can be exploited to optimize the use of spare rows/columns to maximize yield. In Section III, it was shown how this degree of freedom can reduce the cost, in terms of fuses, of yield enhancement. So far in this paper, a local spare has been defined as one, which is restricted to a particular layer in the stacked IC. Similarly, a global spare is defined to be one, which is capable of tolerating a defect in all the layers of the stack. While this approach is successful in reducing the overall cost of repairing defects, it does not completely exploit the degree of freedom offered by this memory architecture. In this section, the domain selection process is generalized to achieve greater reductions.

By allowing spares to cover a smaller domain than a single layer, the cost of repair can be lowered further. For example, a local spare can be restricted to a subset of the columns in a particular layer, thereby lowering the repair cost because fewer fuses are needed to implement it. Similarly, a global spare can also be restricted to any subset of layers as opposed to all the layers. By fine tuning the local spares to a smaller set within the layer and the global spares to a subset of the layers,

TABLE V
DOMAIN SELECTION FOR EACH SPARE

Num.	Config.	Local and Global Domain			
3D-ICs	{L0-1-2-3}	Local	Local	G0	G1
15	4-0-0-0	1	1	L0	L0
61	3-1-0-0	1	1	L0	L1
10	2-2-0-0	1	1	L1	L1
145	2-1-1-0	1	1	L1	L2
19	2-1-0-0	1	1	L1	-

the number of fuses required reduces even more as compared with the approach discussed in Section II, where spares are restricted to only global or local to one layer.

To illustrate this concept, consider the examples from Section II of a four-layer stack with a defect ratio of 1.0 and a total of 1000 dies. It has 250 stacks of four dies in each stack with the distribution shown in Table I.

Using the algorithm described in Section III, it is known that the total number of spares needed to tolerate all the defects in each of the stack is four (maximum of the sum of the defects in each stack). After going through the complete algorithm, it is shown (in Section II) that 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, as shown in Table II. This saves a total of four fuses.

By further fine tuning the domain for local and the global spares, as described earlier in this section, we can reduce one more fuse. Because we have a layer (top layer) with zero defects, none of the spares would need to tolerate a defect in that layer. Similarly, in the next layer, the most defective die has one defect; hence, covering that layer with only of the two global spares should be sufficient. Below we show how we can assign the global and local spares.

One of the global spares (G0 in Table V) only has to repair a defect in either layer 0 (L0 in Table V) or layer 1 (L1 in Table V). Therefore, by fine tuning the domain over which each of the global spares are capable to repair defects, the number of fuses is reduced by one because one of the global spare needs $\log_2$(two layers) fewer fuses. This increases the savings from four fuses to five.

In Section VI, the experimental results for fine tuning the domain of each spare are shown where the cost, in terms of fuses, of yield enhancement is reduced over several defect ratios and stacked ICs.

VI. COMBINING WITH SRP

Another very efficient way that the proposed concept can be used is with the SRP scheme described in [9]. 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 because 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 four partitions of the row address space, then the number of fuses required would be $4\log_2(c)$. If four defects are, however, 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 three defects are repaired, then the number of fuses is higher for SRP with one spare column compared with using three spare columns [i.e., $4\log_2(c)$ versus $3\log_2(c)$]. Therefore, the efficiency of SRP depends on how much of the maximum repair capability of SRP can be used.

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 spare without requiring additional fuses. If SRP's repair capability cannot always be maximally used, then there is some fuse overheads 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 using SRP at the cost of more fuses. Therrefore, 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 described in the previous sections. A subset of the local spares in some layers 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 Section VIII, the results are shown for two cases. One where SRP is used only when it will not increase the number of fuses (i.e., its capacity is fully used), and the other is where SRP is used more aggressively so that the number of spares per cell array die is reduced at some costs in terms of additional fuses.

VII. EXPERIMENTAL RESULTS

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 dies are manufactured, it simply targets some maximum defect ratios for which it is desired that the repair capability be able to handle.

Results are shown in Fig. 2 assuming a uniform distribution of defects across memory cells and die. Results are plotted for five different methods providing repair capability sufficient for the targeted defect ratio. The first is the conventional symmetric repair where all the spares were considered to be global. The second is the proposed asymmetric repair

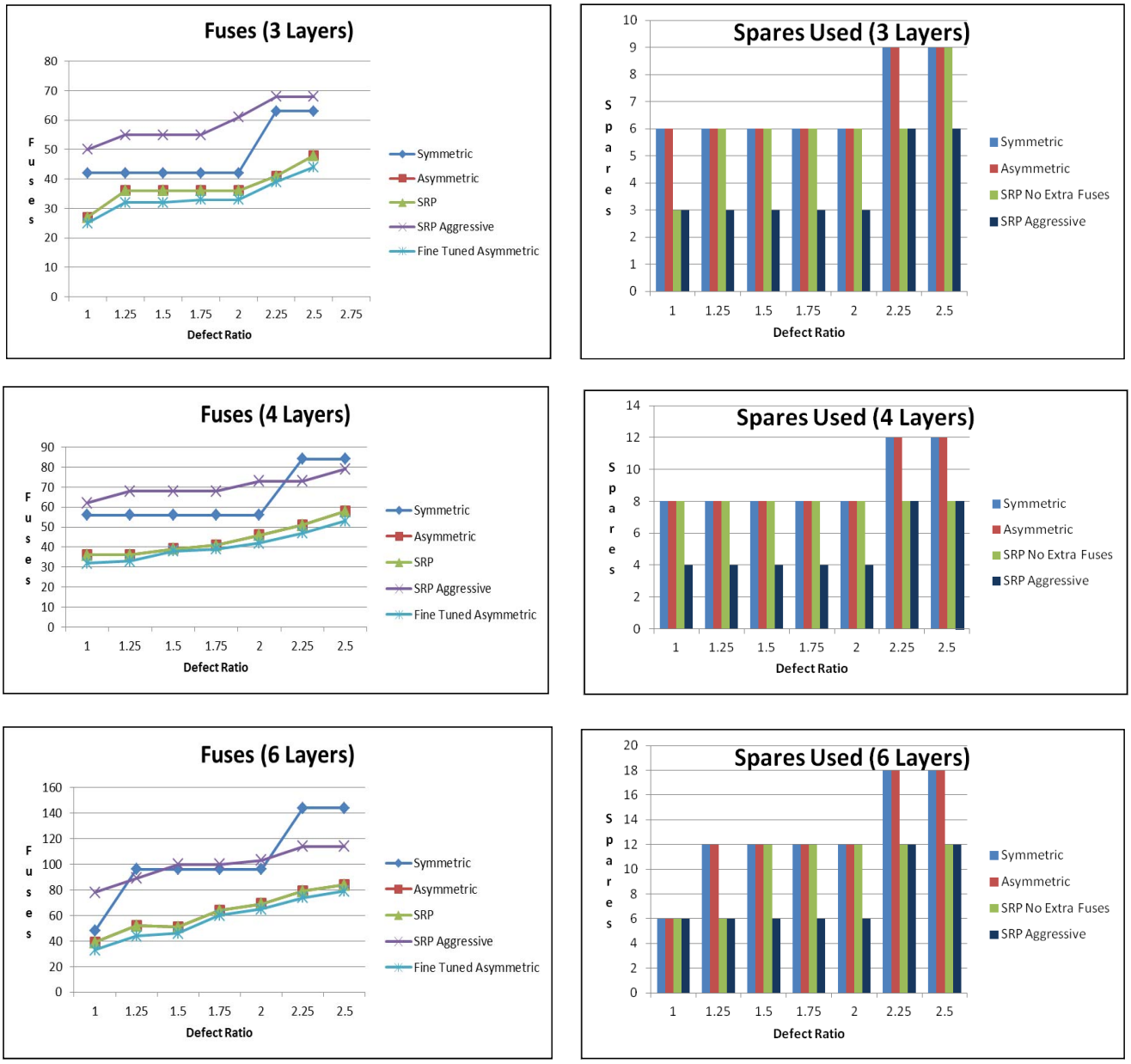


Fig. 2. Comparison of fuse cost and number of spares using different methods targeting different defect ratios.

(Section II) where the procedure in Section III 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 fifth approach is using the technique described in Section IV to improve fine-tune domain selection for each of the spares. The results were generated assuming three, four, and six 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 because each cell array die is identical and contains the same number of spares. Furthermore, only four approaches are shown in the spares results because both the asymmetric approaches (Sections II and IV) use identical number of spares. The savings are only in the number of fuses if truly asymmetric approach is used.

From the results, it can be seen that the number of fuses can be significantly reduced using the proposed asymmetric repair approaches. 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

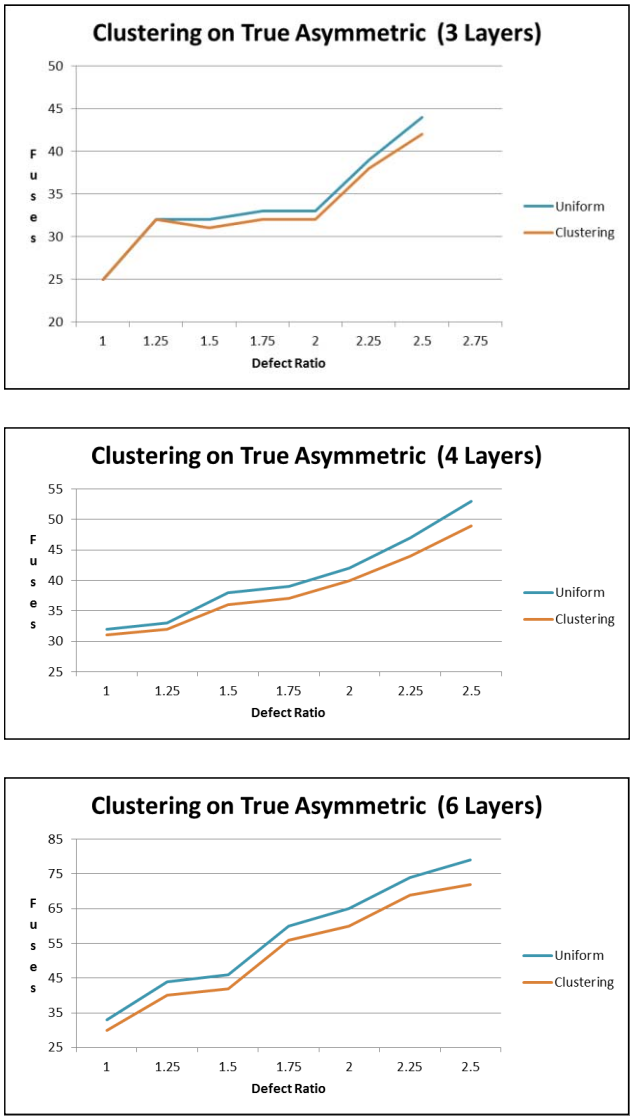


Fig. 3. Truly asymmetric approach with and without clustering.

the number of fuses required for conventional symmetric SRP.

The results in Fig. 2 are conservative in that they assume that the defects are uniformly distributed, which is the worst case for the proposed method. To see how clustering of defects would impact the results, the experiments were performed where defects were modeled using the Polya Eggenberger distribution with $\lambda = 2.130$ and $\alpha = 2.382$ [11] to represent the case with clustered defects. In Fig. 3, the cost of fuses with a clustered distribution is compared against the results with a uniform distribution for the fine-tuned asymmetric case. As can be seen, the results are somewhat better when there is clustering. The improvement increases with more layers.

VIII. Conclusion

The new concept of asymmetric repair described here is made possible by the degree of freedom in multilayer 3DIC memories that the order of the die in the stack can be selected. By matching up die with more defects to the 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 [9] can be efficiently used to also reduce the number of spares per cell array. This can help to reduce the overall cost considerably as reduction in spare are in multiples of the number of layers per stack. The gains are significantly more as compared with only the fuse savings using the asymmetric repair technique by itself. It is further shown that by fine tuning the domain of each spare further cost reduction in fuses is possible. Finally, the results show that the proposed scheme works best when faults are clustered together. This is because clustering increases the number of zero-fault dies and also increases the number of multifault dies. The increase in multifault dies increases the assignment of local spares (Section III), whereas the increase in the number of zero-fault dies helps with the better selection of global spares (Section V).

References

[1] C.-C. Chi, Y.-F. Chou, D.-M. Kwai, Y.-Y. Hsiao, C.-W. Wu, Y.-T. Hsing, L.-M. Denq, and T.H. Lin, "3D-IC BISR for stacked memories using cross-die spares," in *Proc. Int. Symp. VLSI DAT*, Apr. 2012, pp. 1–4.

[2] C.-W. Chou, Y.-J. Huang, and J.-F. Li, "Yield-enhancement techniques for 3D random access memories," in *Proc. Int. Symp. VLSI DAT*, Mar. 2010, pp. 104–107.

[3] Y.-F. Chou, D.-M. Kwai, and C.-W. Wu, "Yield enhancement by bad-die recycling and stacking with through-silicon vias," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 8, pp. 1346–1356, Aug. 2011.

[4] L. Jiang, R. Ye, and Q. Xu, "Yield enhancement for 3D-stacked memory by redundancy sharing across dies," in *Proc. ICCAD*, Nov. 2010, pp. 230–234.

[5] I. Kim, Y. Zorian, G. Komoriya, H. Pham, F. P. Higgins, and J. L. Lewandowski, "Built in self repair for embedded high density SRAM," in *Proc. Int. Test Conf.*, Oct. 1998, pp. 1112–1119.

[6] S.-Y. Kuo and K. Fuchs, "Efficient spare allocation for reconfigurable arrays," *IEEE Design Test*, vol. 4, no. 1, pp. 24–31, Feb. 1987.

[7] B. Noia and K. Chakrabarty, "Pre-bond probing of TSVs in 3D stacked ICs," in *Proc. IEEE ITC*, Sep. 2012, pp. 1–10.

[8] M. T. Rab, A. A. Bawa, and N. A. Touba, "Improving mmeory repair by selective row partitioning," in *Proc. 24th IEEE Symp. Defect Fault Tolerance*, Oct. 2009, pp. 211–219.

[9] M. T. Rab, A. A. Bawa, and N. A. Touba, "Using asymmetric layer repair capablity to reduce the cost of yield enhancement in 3D stacked memories," in *Proc. VLSI-SoC*, Oct. 2012, pp. 195–200.

[10] S. E. Shuster, "Multiple word/bit line redundancy for semiconductor memories," *IEEE J. Solid-State Circuits*, vol. 13, no. 5, pp. 698–703, Oct. 1978.

[11] C. H. Strapper, A. N. McLaren, and M. Dreckmann, "Yield model for productivity optimization of VLSI memory chips with redundancy and partially good product," *IBM J. Res. Develop.*, vol. 24, no. 3, pp. 398–409, May 1980.

[12] M. Taouil and S. Hamdioui, "Layer redundancy based yield improvement for 3D wafer-to-wafer stacked memories," in *Proc. 16th IEEE ETS*, May 2011, pp. 45–50.

[13] X. Wang, D. Vasudevan, and H.-H. S. Lee, "Global built-in self-repair for 3D memories with redundancy sharing and parallel testing," in *Proc. Int. 3DIC*, Aug. 2012, pp. 1–8.

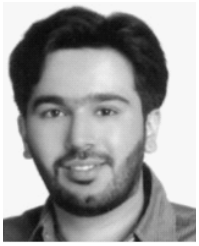
[14] C.-L. Wey and F. Lombardi, "On the repair of redundant RAM's," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 6, no. 2, pp. 222–231, Mar. 1987.

[15] Y. Zorian and S. Skoukourian, "Embedded-memory test and repair: Infrastructure IP for SOC yield," *IEEE Design Test Comput.*, vol. 20, no. 3, pp. 58–66, May 2003.

[16] V. Hemmady and S. M. Reddy, "On repair of redundant RAMs," in *Proc. Design Autom. Conf.*, Jun. 1989, pp. 710–713.

Muhammad Tauseef Rab received the B.S., M.S., and Ph.D. degrees from the University of Texas at Austin, Austin, TX, USA, in 2002, 2005, and 2013, respectively, all in electrical engineering.

He is currently a Design Engineer with Qualcomm, Inc., Austin, TX, USA. He has worked in the semiconductor industry for over ten years. His current research interests include VLSI testing, design for test and yield analysis, and improvements in ICs.



Asad Amin Bawa received the B.S. and M.S. degrees from the University of Texas at Austin, Austin, TX, USA, in 2003 and 2006, respectively, both in electrical engineering. He is currently pursuing the Ph.D. in electrical engineering from the University of Texas at Austin, Austin, TX, USA.

He has worked with the industry in the VLSI field for over ten years. His current research interests include VLSI testing, design for test and yield analysis, and improvements in ICs.

Nur A. Touba (SM'05–F'09) received the B.S. degree from the University of Minnesota, Minneapolis, MN, USA, in 1990, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, USA, in 1991 and 1996, respectively, all in electrical engineering.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX, USA.

Dr. Touba was a recipient of the National Science Foundation Early Faulty CAREER Award in 1997, the Best Paper Award in the 2001 VLSI Test Symposium, and the 2008 Defect and Fault Tolerance Symposium. He served as a Program Chair for the 2008 International Test Conference and a General Chair for the 2007 Defect and Fault Tolerance Symposium. He currently serves on the program committee for the Design Automation and Test in Europe Conference, International On-Line Test Symposium, European Test Symposium, Asian Test Symposium, and Defect and Fault Tolerance Symposium.