Is TSV-based 3D Integration Suitable for Inter-die Memory Repair?

Mihai Lefter, George R. Voicu, Mottaqiallah Taouil, Marius Enachescu, Said Hamdioui and Sorin D. Cotofana
Delft University of Technology, Delft, The Netherlands
E-mail: M.Lefter, G.R.Voicu, M.Taouil, M.Enachescu, S.Hamdioui, S.D.Cotofana @tudelft.nl

***Abstract*—In this paper we address lower level issues related to 3D inter-die memory repair in an attempt to evaluate the actual potential of this approach for current and foreseeable technology developments. We propose several implementation schemes both for inter-die row and column repair and evaluate their impact in terms of area and delay. Our analysis suggests that current state-of-the-art TSV dimensions allow inter-die column repair schemes at the expense of reasonable area overhead. For row repair, however, most memory con gurations require TSV dimensions to scale down at least with one order of magnitude in order to make this approach a possible candidate for 3D memory repair. We also performed a theoretical analysis of the implications of the proposed 3D repair schemes on the memory access time, which indicates that no substantial delay overhead is expected and that many delay versus energy consumption tradeoffs are possible.**

I. Introduction

Recent enhancements in Integrated Circuits (ICs) manufacturing process enable the fabrication of three dimensional stacked ICs (3D-SICs) based on Through-Silicon-Vias (TSVs) as die-to-die (D2D) interconnects, which further boost the trends of increasing transistor density and performance. 3D-SIC is an emerging technology, that, when compared with planar ICs, allows for smaller footprint, heterogeneous integration, higher interconnect density between stacked dies, and latency reduction mostly due to shorter wires [1].

3D memories have been proposed ever since the technology was introduced, one of the reason being their regular structure that allows them to be easily folded accross bitlines/wordlines and spread over multiple layers in a 3D embodiment [2]. Moreover, the typical area of a System on a Chip (SoC) is memory dominated, and, as the ITRS roadmap predicts that the trend of memory grow continues [3], it is expected that memories will play a critical role in 3D-SICs as most of the layers in the stack are likely to be allocated for storage.

As technology keeps shrinking towards meeting the requirements of increased density, capacity, and performance, IC circuits, memory arrays included, are more prone to degradation mechanism [4], and different sorts of defects during the manufacturing process [5]. In addition, the utilization of the still in its infancy 3D stacking technology increases the risk of low yield. To deal with this issue several works proposed inter-die memory repair, i.e., sharing redundant elements (rows/columns) between layers, in an attempt to increase the compound yield of memories [6], [7], [8], [9], [10], [11].

Up until now, all the work targeting inter-die memory repair primarily discussed the idea in principle, with no real implications being studied. The proposed approaches have been only evaluated via fault injection simulations and the obtained repair rate improvements form an upper bound. In order to achieve inter-die repair, a certain infrastructure has to be embedded into the memory such that spares can be made available to memory arrays in need that are located on remote dies. The added infrastructure must not affect the normal operation of the memory and may incur certain penalties in terms of area and/or delay which have not be studied.

In this paper we build upon previous work proposals and we further investigate the real implications of inter-die memory repair based on redundancy sharing. We first provide a classification of the possible access scenarios to memory arrays stacked in a 3D memory cube. Next, we propose several implementation schemes both for inter-die row and column repair in which we detail the circuit infrastructure required to support these access scenarios. For each scheme we propose the infrastructure, highlight its advantages and disadvantages, and discuss its impact on memory area and delay.

The area overhead is mostly dependent on the TSV size rather than on the extra logic. From our analysis it results that current state-of-the-art TSV dimensions allow inter-die column repair schemes with reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down with at least one order of magnitude to make this approach applicable in practical 3D memory systems. We also performed a theoretical analysis of the implications of the proposed 3D repair schemes on the memory access time. Assuming a 20ps TSV delay our analysis indicates that for row repair the overhead is negligible and for column repair it can be in the same order of magnitude. This indicates that no substantial delay overhead is expected and that many delay versus energy consumption tradeoffs are possible.

The remaining of the paper is organized as follows. Section II briefly describes general memory repair techniques and related work regarding 3D memory repair. Section III defines the 3D memory repair architecture and the associated framework for inter-die redundancy. Section IV introduces the circuit infrastructure necessary to support inter-die memory repair. Section V considers various trade-offs and cost overhead in terms of area and delay. Finally, Section VI concludes the paper.

II. Redundancy Based Memory Repair

State of the art memory repair relies on the addition of several redundant resources to the memory arrays. These resources do not affect the interface or capacity of the memory, but can be later on utilized to substitute memory cells affected by, usual, permanent errors. Based on the physical placement of the spare elements we can broadly distinguish two types, that are not excluding one another, of memory redundancy: (i) *external redundancy*, in which a special smaller memory, external to the initial one is present, and where, based on a fault table, bad addresses are remapped by a Built-In Repair

978-3-9815370-0-0/DATE13/©2013 EDAA

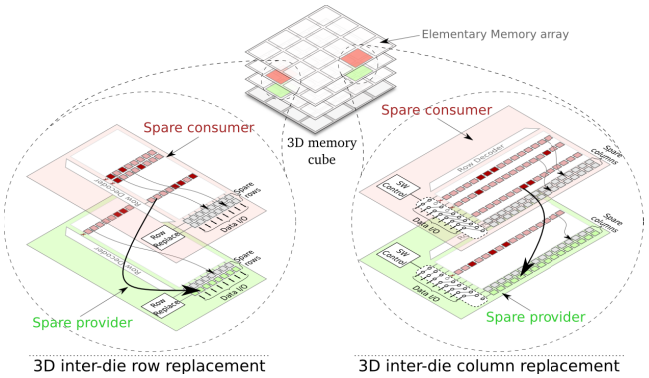


Fig. 1: 3D inter-die memory repair - general idea.

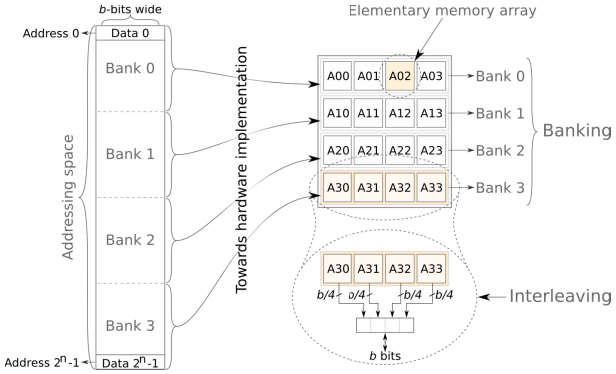


Fig. 2: Memory partitioning.

Analysis (BIRA) unit [12]; and (ii), *internal redundancy*, in which spare elements in the form of redundant rows and/or columns, are placed inside the memory alongside the normal columns and/or rows.

In this paper we consider *internal redundancy* only. Here, the mechanisms involved for row and column repair are quite different. For *row replacement*, detected faulty row addresses are stored in special registers. Whenever the memory is accessed, the incoming address is first compared with those stored in the special registers to check if a defective row is to be accessed. If this is the case, the output of the comparator disables the row decoder and activates the spare wordline. For *column replacement*, in general, a column switching mechanism is present to isolate the faulty column and to forward data from non-defective cells [13].

To create extra opportunities for memory repair various inter-die approaches have been proposed. In [7] a Die-to-Die (D2D) stacking flow algorithm is presented which assumes that each die is beforehand locally repaired such that the number of available (not utilized for local repair) spares are made available as inputs for the global repair algorithm. This method for inter-die column replacement is suitable only for the particular case where arrays are simultaneously accessed with the same address. A similar D2D stacking approach is considered in [11] where the die stacking flow is modeled as a bipartite graph maximal matching problem. Several global D2D matching algorithms without local repair first are introduced and compared in [8]. An interesting approach is introduced in [9] where the authors propose to recycle irreparable dies (i.e., dies with arrays that are not repairable if only local spares are considered) in order to create good working memories.

III. 3D Inter-die Memory Repair Architecture

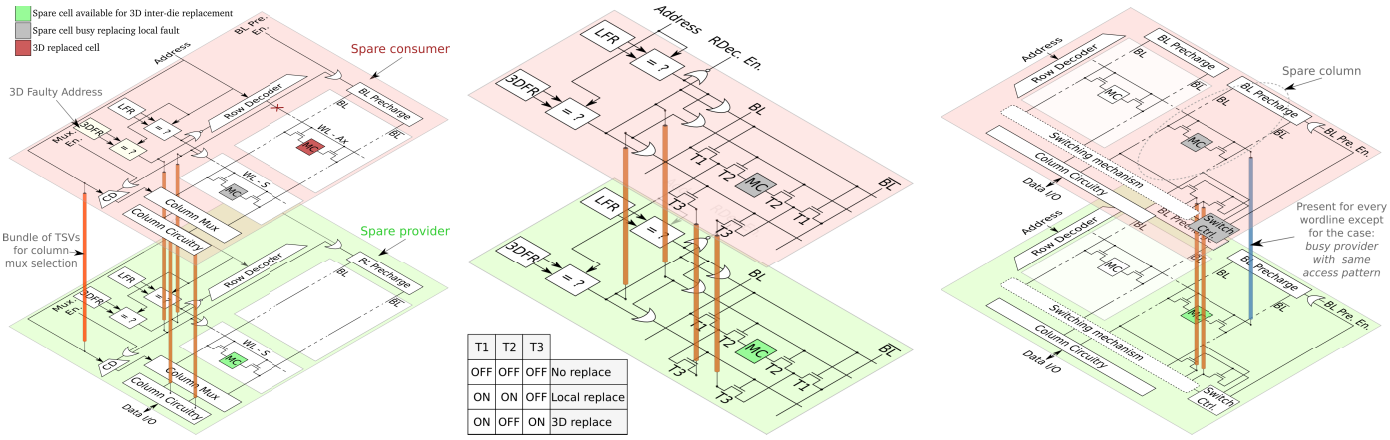
The considered memory arrangement resembles a memory cube, as depicted in Fig. 1. The cube employs 3D *array stacking* with the identical memory arrays being equipped with redundant rows and/or columns. In this organization, a situation may arise in which arrays with insufficient redundancy are in the vertical proximity of arrays that still have unutilized redundant elements. Supporting the replacement of faulty cells by using redundant resources from arrays from other dies, i.e., inter-die spare replacement, results in extended memory reparability rates [8]. This can be observed on the lower part of Fig. 1, where the top arrays have utilized all their available spare rows/columns (two in this case) and still have one faulty row uncovered. However, the bottom arrays can provide the necessary spare rows/columns to replace the faulty ones on the top arrays to make the memory defect free.

We define the arrays that have available spare resources as *spare providers* and arrays which make use of the externally available spare resources as *spare consumers*. For the 3D memory repair to function correctly the consumer must be able to retrieve/store data from/on the provider in a transparent manner, i.e., the provider must be able to function normally, despite its spares being accessed by a neighboring die. In addition, it is important that the inter-die repair infrastructure does not disrupt the functionality of the memory cube when no repair takes place. Therefore, the required infrastructure that assures the memory repair mechanism is highly dependent on the exact internal structure of the memory arrays.

In order to balance area, delay, and power tradeoffs, a large memory is usually constructed in a hierarchical manner and is composed out of several banks, with each bank being further divided in several arrays. An example is presented in Fig. 2 where the partitioning employs banking and interleaving. Each bank can be accessed either concurrently with independent addresses, or sequentially, where one bank is accessed while the rest remain idle. For interleaving, however, all the subarrays of a bank are concurrently accessed with the same address.

As the internal organization of the memory cube is defined at design time, a fixed memory partitioning implies three exclusive situations in which two memory arrays, a *provider-consumer* pair, can be accessed: (i) **Idle provider** - the two arrays are located in different banks that are never concurrently accessed; we use the term idle to denote that the two arrays are never accessed at the same time; from the *consumer*'s perspective this is equivalent with the *provider* being always idle; (ii) **Busy provider with different access pattern** - the two arrays are located in different banks that are concurrently accessed with independent addresses; (iii) **Busy provider with same access pattern** - the two arrays are part of the same bank with interleaving, therefore the accessing address is the same.

We add that, although our proposal is general and can in principle be applied to more than two adjacent dies, in this paper we consider that inter-die replacement is performed between exclusive pairs of adjacent dies. The reasons behind this restriction are as follows: (i) the infrastructure overhead grows with the number of dies involved in the spare sharing process, and, (ii) two dies spare replacement is enough to sustain a satisfactory yield [8], since the die yield has a high value after repair. In the next section we introduce the infrastructure for the above identified *provider-consumer* repair schemes.



(a) (b) (c)

Fig. 3: Inter-die row (a,b) and column (c) replacement infrastructure.

IV. 3D Inter-die Memory Repair Infrastructure

In this section we detail the inter-die repair schemes for each of the three *provider-consumer* pair scenarios introduced in Section III for row and column replacement.

A. Inter-die Row Replacement

1) Idle provider: Fig. 3a depicts the situation for the case in which the *provider* is idle. On the *consumer* side, the local spare row is already allocated and another faulty row needs to be replaced remotely. A register is required to store its address (3DFR - 3D Fault Register), in a similar fashion as in the local replacement scheme. Furthermore, a comparator and several logic gates are introduced in the design to disable the local row decoder and to activate the spare wordline on the *provider* side whenever the incoming address is equal to the value stored in 3DFR. We propose to place the data TSVs after the column multiplexer. This requires the column address to be transfered through TSVs and the column decoder (CD) on the *provider* to be enabled. In this manner fewer TSVs are required when compared to the case when TSVs are placed for every bitline.

2) Busy provider with different access pattern: When both *consumer* and *provider* can be accessed in parallel the constraints imposed to the inter-die memory repair interface are tighter, making the infrastructure more complex. In particular, when an inter-die replacement occurs, both *consumer* and *provider* need to use the *provider*'s bitlines, giving rise to a conflict. For this reason the data TSVs cannot be placed after the column muxes and extra transistors (denoted by T2 and T3) are required in every spare memory cell, as in Fig. 3b.

3) Busy provider with same access pattern: A particular case of *provider* and *consumer* parallel access arises when their address is the same (i.e., in an interleaving organization of memory banks, see Section III). In contrast with the previous scheme, the infrastructure can be reduced in terms of logic. However, each spare cell still needs to be augmented with 4 extra transistors and 2 TSVs. Thus, even if we assume that future TSV manufacturing process will be greatly improved to a negligible size, the cell area almost doubles.

B. Inter-die Column Replacement

The general infrastructure required for inter-die column replacement depicted in Fig. 3c comprises all the cases introduced in Section III. The common part for all the cases consists of the TSV pair utilized for bitline value transmission. They are enabled by the switching control block, which needs to be adapted to control also the inter-die replacement mechanism.

A special TSV is required for every wordline whenever the *provider* is busy accessing a different address, or when it is idle, in order to assert the required wordline for the *consumer*. When the *provider* is busy it is also mandatory to decouple the *provider's* wordline such that no bitline conflict arises because of multiple wordlines assertion. For brevity this action is not represented in Fig. 3c. For the case in which the *provider* is idle, the TSV required for the wordline activation may be discarded if the *provider*'s row decoder is enabled. However, this requires the consumer's address to be driven onto the TSVs. Nevertheless, the gain is a significant TSV reduction.

The easiest and most convenient inter-die column replacement scheme in terms of TSV requirements is by far when the *consumer* and the *provider* are busy accessing the same address. Here, the same wordline is asserted in both arrays and no bitlines conflicts occur.

V. Discussion

In this section we discuss the overhead of the 3D inter-die memory repair schemes in terms of area and delay.

Area represents a sensitive issue in memory design and the memory cell is particulary the subject of severe scaling. SRAM bit cell has followed Moores's law, with an area shrinking rate of about $1/2$ for every generation, reaching $0.081\ \mu m^2$ for the $22\ nm$ technology [14]. This rate is expected to last even in the realm of post-CMOS devices [15]. TSVs dimensions are predicted to scale down too, but not that steep as SRAM bit cells. The predictions from [16] suggest a gradually decreasing trend with a shrinking ratio of about $1/4$ for every 3 years, reaching a minimum diameter of $0.8\ \mu m$ and a pitch of $1.6\ \mu m$ by 2018. Nowadays manufactured TSVs have a diameter bewteen 3 and $10\ \mu m$ and a pitch of about $10\ \mu m$ [17], [18], [19]. From their large size it is clear that TSVs represent the major contributor to the 3D memory repair area overhead.

Table I presents the TSV requirements for all the scenarios introduced in Section III. The scenarios that have the least number of TSVs are "idle *provider*" for row redundancy and "busy *provider* with same access" for column redundancy. All the other scenarios require a large number of TSVs that make them absolutely impractical. Even for the row redundancy with the "idle *provider*" scenario the practicality is problematic. Fig. 4 depicts the area of one redundant row and its required

TABLE I: TSV REQUIREMENTS FOR PROPOSED SCHEMES

Memory access scenarios	Number of TSVs
Row replacement	
Idle *provider*	$2 \times spares + 2 \times dw + cd\_bits$
Busy *provider* with different address	$spares \times (2 + 2 \times columns)$
Busy *provider* with same address	$spares \times (2 + 2 \times columns)$
Column replacement	
Idle *provider*	$2 \times spares + log_2(rows)$
Busy *provider* with different address	$2 \times spares + rows$
Busy *provider* with same address	$2 \times spares$

* *dw* = data width (data I/O); *cd_bits* = column decoder input bits.

TSVs. The redundant row area is drawn for different technology nodes using memory widths varying between 128 and 2048 bits. The TSV area is independent of the technology node and is calculated for a TSV pitch between 0.5 and 3.5 μm and a memory data width of 32 bits. Given that, inter-die redundancy may be profitable only if the redundant row area is smaller than the TSV area. Fig. 4 clearly suggests that for a large TSV pitch inter-die redundancy becomes impractical.

It is interesting be to find the required TSV pitch for which inter-die row replacement becomes advantageous. For example, in case the column width is 512 and the data output width is 32, the TSV pitch must be at most 534 nm. For the worst case considered configuration, with a column width of 512 and data output width of 64, TSV pitch needs even to scale further down to 388 nm. Therefore, current TSV sizes in the order of 3 μm need to be shrinked severely for inter-die redundancy to be beneficial for a wide range of memory configurations.

The access time (T_{N2D}) for a normal memory read operation (Eq. (1)) is determined by: address decoding (T_{dec}), wordline generation (T_{WL}), bitlines discharge (T_{BL}), column multiplexing (T_{mux}), and data sensing (T_{SA}). If row redundancy is present and the redundant row is accessed the access time changes to T_{R2D} (Eq. (2)), because the access goes through the comparator (T_{cmp}) instead of the decoder. For 3D row redundancy, extra time is required to transfer data to the *consumer* through the TSVs (T_{TSV}), resulting in T_{R3D} (Eq. (3)). The time overhead for 3D row redundancy (D_{OR}) can be computed as in Eq. (4).

For 3D inter-die column redundancy, the access time increases as in Eq. (6). Thus, there is always a delay overhead (D_{OC}) due to TSV propagation and switching time.

$$T_{N2D} = T_{dec} + T_{WL} + T_{BL} + T_{mux} + T_{SA} \quad (1)$$

$$T_{R2D} = T_{cmp} + T_{WL} + T_{BL} + T_{mux} + T_{SA} \quad (2)$$

$$T_{R3D} = T_{R2D} + 2 \times T_{TSV} \quad (3)$$

$$D_{OR} = \frac{max(T_{N2D}, T_{R3D}) - max(T_{N2D}, T_{R2D})}{max(T_{N2D}, T_{R2D})} \quad (4)$$

$$T_{C2D} = T_{N2D} + T_{switching} \quad (5)$$

$$T_{C3D} = T_{C2D} + T_{switching} \quad (6)$$

$$D_{OC} = \frac{T_{switching} + T_{TSV}}{T_{C2D}} \quad (7)$$

As the delay of a TSV is in the order of 20 ps [20], we expect the following to hold true: $T_{R2D} < T_{R3D} < T_{N2D}$. Therefore, no delay penalty for row repair is expected. For column repair however, the following inequation holds: $T_{N2D} < T_{C2D} < T_{C3D}$. The overhead is determined by the delay of switching muxes and a TSV ($T_{switching} + T_{TSV}$) which is expected to be minimal.

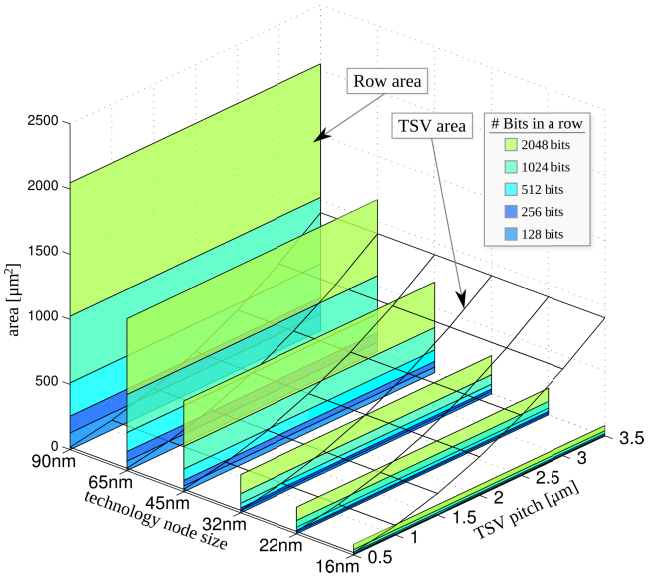


Fig. 4: TSV area overhead vs. row area for 32-bit data I/O.

VI. CONCLUSIONS

In this paper, we presented a study of inter-die repair schemes for TSV based 3D-SICs, i.e., of using repair in the vertical dimension.The paper provided an overview of general repair schemes and subsequently, proposed a memory framework for inter-die redundancy based on a *provider-consumer* pair scheme. Our analysis suggests that for state-of-the-art TSV dimensions inter-die column-based repair schemes could result in yield improvements at a reasonable area overhead. For row repair, however, most memory configurations require TSV dimensions to scale down at least with one order of magnitude to be utilized in 3D memory systems.

REFERENCES

[1] P. Garrou, *Handbook of 3D integration : technology and applications of 3D integrated circuits*. Weinheim: Wiley-VCH, 2008.
[2] K. Puttaswamy *et al.*, "3D-Integrated SRAM components for high-performance microprocessors," *TC*, 2009.
[3] "ITRS - System Drivers," Tech. Rep., 2011. http://www.itrs.net
[4] S. Rusu *et al.*, "Trends and challenges in VLSI technology scaling towards 100nm," in *VLSI Design / ASPDAC*, 2002.
[5] S. R. Nassif, "The light at the end of the CMOS tunnel," *ASAP*, 2010.
[6] R. Anigundi *et al.*, "Architecture design exploration of three-dimensional (3D) integrated DRAM," in *ISQED*, 2009.
[7] C. Chou *et al.*, "Yield-enhancement techniques for 3D random access memories," in *VLSI-DAT*, 2010.
[8] L. Jiang *et al.*, "Yield enhancement for 3D-stacked memory by redundancy sharing across dies," in *ICCAD*, 2010.
[9] Y.-F. Chou *et al.*, "Yield enhancement by bad-die recycling and stacking with though-silicon vias," *TVLSI*, 2011.
[10] C.-W. Wu *et al.*, "On test and repair of 3D random access memory," in *ASPDAC*, 2012.
[11] S. Lu *et al.*, "Yield enhancement techniques for 3-dimensional random access memories," *Microelectronics Reliability*, 2012.
[12] N. Axelos *et al.*, "Efficient memory repair using cache-based redundancy," *TVLSI*, 2011.
[13] M. Horiguchi *et al.*, *Nanoscale Memory Repair*. Springer, 2011.
[14] K. Smith *et al.*, "Through the looking glass: Trend tracking for ISSCC 2012," *M-JSSC*, 2012.
[15] H. Iwai, "Roadmap for 22nm and beyond," *Microelectronic Eng.*, 2009.
[16] "ITRS - Interconect," Tech. Rep., 2011. http://www.itrs.net
[17] C. L. Yu *et al.*, "TSV process optimization for reduced device impact on 28nm CMOS," in *TVLSI*, 2011.
[18] G. Katti *et al.*, "3D stacked ICs using cu TSVs and die to wafer hybrid collective bonding," in *IEDM*, 2009.
[19] H. Chaabouni *et al.*, "Investigation on TSV impact on 65nm CMOS devices and circuits," in *IEDM*, 2010.
[20] D. H. Kim *et al.*, "Through-silicon-via-aware delay and power prediction model for buffered interconnects in 3D ICs," in *SLIP*, 2010.