3D-DyCAC: Dynamic Numerical-based Mechanism for Reducing Crosstalk Faults in 3D ICs

Zahra Shirmohammadi
shirmohammadi@ipm.ir

School of Computer Science,
Institute for Research in
Fundamental Sciences (IPM)

Hadi Zamani Sabzi
hzama001@ucr.edu

University of California Riverside

Seyed Ghassem Miremadi
miremadi@sharif.edu

Sharif University of Technology

***Abstract*— One of the cost-efficient fabrication approaches for connecting layers in *three-dimensional integrated circuits* (3D ICs) is the use of *through-silicon vias* (TSVs). However, the large and closely spaced nature of TSVs has made them seriously prone to coupling capacitances between TSVs, increasing the probability of crosstalk faults. To reduce crosstalk faults in 3D ICs, this paper proposes a *dynamic numerical-based crosstalk avoidance* mechanism called 3D-DyCAC. The 3D-DyCAC mechanism is applied in two phases. In the first phase, a numerical system-based *crosstalk avoidance code* (CAC) is proposed to reduce the opposite-direction transitions in adjacent TSVs. This numerical-based CAC generates code words with cumulative groups of *1*s and *0*s to minimize adjacent transitions in arranged N × N meshes of TSVs. The proposed CAC has no ambiguity in representing code words and generates a unique code word for each data word. In the second phase, a *triangular window* (TW) is introduced to consider overlaps between the cells of TSVs during the arrangement of code words on TSVs. In the TW area, the content of the victim TSV and/or the content of adjacent bit positions can be dynamically inverted based on the content of data words. Evaluation results show that 3D-DyCAC reduces the area occupation, power consumption, and power-delay product of codec by 10%, 24%, and 46%, respectively, in comparison with state-of-the-art mechanism 3DLAT.**

***Keywords*—** Three-Dimensional Integrated Circuits (3D ICs); Through-Silicon Vias (TSVs); Crosstalk Faults; Crosstalk Avoidance Code (CAC).

I. INTRODUCTION

Increasing the delay gap between interconnections and gates has made interconnection delay the major performance bottleneck in *two-dimensional integrated Circuits* (2D ICs) [1]. This is due to long interconnections between processing elements in 2D ICs that increase the hop count between source and destination during data transmission. To address this problem, designers often add another dimension, shortening the routing hops by providing *three-dimensional integrated circuits* (3D ICs) [1][2]. The 3D ICs have several advantages over 2D ICs, including higher scalability, higher throughput, and lower power consumption [2]. However, with migration to 3D ICs, new evil rises due to coupling capacitance between *through-silicon vias* (TSVs) that connect different stack layers [1]. The coupling capacitances between TSVs cause them to mutually affect the transmitted signal of adjacent TSVs and generate crosstalk faults [3]. Crosstalk faults are due to TSVs' large diameter and their bounded, adjacent, short structure, and crosstalk faults can lead to unwanted voltage glitches and delay and/or speed-up in rising/falling transitions appearing on the center TSV (called the *victim TSV*). These effects can lead to the corruption and misrouting of data and also power dissipation in 3D ICs [3][4].

Mechanisms to address crosstalk faults [5]-[21] have been developed at the physical level [4], transistor level, and *register-transfer level* (RTL) [5]-[21]. Shielding and repeater insertion and skew transitions are examples of physical level and transistor level mechanisms to tackle crosstalk faults. Coding mechanisms at RTL level are among the promising mechanisms. Numerical-based *crosstalk avoidance codes* (CACs) at the RTL are relatively cost-efficient compared to physical-level and transistor level mechanisms, and they reduce crosstalk faults by preventing specific transition patterns [16][21]. However, CACs cannot directly be applied in 3D ICs. This is due to the fact that, in 2D ICs, the victim TSV is surrounded by only two aggressor wires, while in 3D ICs, the victim TSV is surrounded by multiple aggressor TSVs [3]. This makes the TSV-to-TSV coupling capacitance more complex. In other words, the additional dimension results in a significant difference between using CACs in 2D IC and 3D IC designs.

A handful of previous studies on 3D ICs have developed approaches to reduce or omit crosstalk faults by using numerical-based CACs [5][6][12][13], but these approaches have entailed undesirable overheads. To solve this problem, the current paper proposes an overhead-efficient CAC-based mechanism for 3D ICs called *dynamic numerical-based crosstalk avoidance* (3D-DyCAC). The mechanism of 3D-DyCAC is applied in two phases. In the first phase, a numerical system generates code words to minimize adjacent transitions. The numerical system generates a unique code word for each data word and consequently has no ambiguity in representing code words. In the second phase, a *triangular window* (TW) is introduced to consider overlaps between the cells of TSVs. Through the use of the TW during the arrangement process, the content of the victim TSV and/or the content of an adjacent bit position are inverted dynamically based on the content of the data word. Evaluation shows that 3D-DyCAC reduces the area occupation, power consumption, and power-delay product of codec by 10%, 24%, and 46%, respectively, in comparison with state-of-the-art mechanism [5].

978-1-5090-3997-5/17/$31.00 ©2017 IEEE

II. 3D-DyCAC: Dynamic Numerical-based Crosstalk Avoidance Mechanism

Numerical-based CACs are among the cost efficient mechanisms that can prevent the occurrence of specific transition patterns [19]. This is done by encoding *data word* $d = d_N d_{N-1} d_{N-2} d_{N-3\ldots} d_2 d_1$ in a sender and decoding *code word* $c = c_N c_{N-1} c_{N-2} c_{N-3\ldots} c_2 c_1$ in a receiver, where N is the number of TSVs. Using numerical system $b = b_N, b_{N-1}, \ldots, b_1$, data words can be calculated by $c = \sum_{i=1}^{N} d_i \times b_i$ in a sender. In other words, a numerical system is a mathematical notation that uses symbols in a consistent manner to represent numbers of a given set [19]. For example, a numerical system with bases of 9 3 2 2 1 1 1 0 1 maps data word 0 1 1 0 0 0 0 0 0 to code word 0 0 0 0 0 1 1 1 1, in such a way that $3 = 9 \times 0 + 3 \times 0 + 2 \times 0 + 2 \times 0 + 1 \times 0 + 1 \times 1 + 1 \times 1 + 0 \times 1 + 1 \times 1$. However, the structures and arrangements of TSVs make the application of numerical-based CACs complex in 3D ICs. A system of $N \times N$ TSVs is too complex to be analyzed, so this paper employs a coding in partitions of $3 \times N$ meshes of TSVs. These meshes have 3 rows and N columns. This structure can be extended to meshes of $N \times N$ TSVs.

Considering the arrangement of data words on two neighbor meshes of TSVs shown in Figure 1, the sequence of code word $d = d_N d_{N-1} d_{N-2} d_{N-3\ldots} d_2 d_1$ should satisfy the following conditions:

Condition 1: Direct TSVs (in blue in Figure 1) are located at distance d from the victim TSV (in red in Figure 1), and they impose more crosstalk effects than *diagonal TSVs* with $d\sqrt{2}$ distance from victim TSV (in yellow in Figure 1), so it is required that d_2, d_4, d_6 and d_8 have the same values as far as possible.

Condition 2: When the coding mechanism is extended to a neighboring mesh of TSVs, d_8 becomes a new victim in the neighbor mesh. To prevent crosstalk faults and consider overlapping in this condition, at least two TSVs out of d_1, d_2 and d_3 (in neighbor mesh of TSV) should have the same value as d_8. As these $d_i s$ are arranged in triangle, this area is called the *triangular windows* (TWs*)*, shown in Figure 1 in dashed lines.

To satisfy these two conditions, this paper proposes *dynamic numerical-based crosstalk avoidance* (3D- DyCAC). The proposed 3D-DyCAC is applied in two phases. In the first phase, which satisfies the first condition, a CAC based on a numerical system generates code words with cumulative groups of *1*s and *0*s to minimize tandem transitions in arranged 3 × *N* meshes of TSVs. In the second phase, which satisfies the second condition, a dynamic mechanism considers overlaps between cells of TSVs during the arrangement of code words on TSVs.

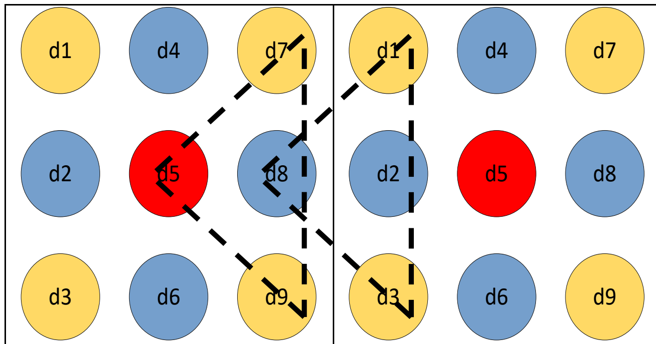

Figure 1. The Arrangement of Data Words in TSVs in Two Neighbor Meshes

In the proposed numerical-system-based CAC, adjacent wires cannot transit in opposite directions when transitioning from one code word to another code word [19]. Thus, transition patterns 01 → 10 and 10 → 01 are avoided. This property can satisfy the first condition, and it generates the code words with cumulative 1s and 0s. This property prevents changing the value of adjacent TSVs. In other words, the bit patterns 010, 101, 1001, and 0110 cannot appear in any of the code words.

The CAC proposed in the first phase of 3D-DyCAC generates bases through Eq. 1:

$$b_i = \begin{cases} 1 & i = 1 \\ 0 & i = 2 \\ g_{i-1} - g_{i-2} & 3 \le i \le N-1 \\ g_{N-3} & i = N \end{cases} \tag{1}$$

Where g_i is the maximum number of code words that can be generated in 3D-DyCAC based on Eq. 2:

$$g_k = g_{N-1} + g_{N-5} \text{ for } N > 6 \tag{2}$$

The initial values of this recursive relation are $g_1 = 2$, $g_2 = 3$, $g_3 = 4$, $g_4 = 5$, and $g_5 = 7$. This numerical system is complete, and for any data word d, $d \in [0, \sum b_i]$, the system generates at least one representation using the numerical system. The completeness is achieved if for all bs, $b_i \le 1 + \sum_{j=1}^{i-1} b$. This numerical system can be extended to any arbitrary number of TSVs.

The mapping algorithm of 3D-DyCAC that maps a data word to a code word using the numerical system is shown in Figure 2. In this mapping algorithm, r_i is the remainder of each step to the next step, and F_i is the ith base of Fibonacci sequence.

Applying CACs in the first phase can reduce the number of transitions by generating cumulative numbers of *1*s and *0*s. However, using this numerical system cannot by itself satisfy the required conditions for applying numerical-based CACs in a mesh of TSVs: the overlaps between the adjacent TSVs are not considered while arranging the data words on a mesh of TSVs. The data words d_2, d_4, d_6, and d_8 should have the same values as far as possible, and at least two out of d_1, d_2, and d_3 should have the same value as d_8. It is important to check the two required conditions after generating code words using CAC. To perform such a check, two counters, called *direct neighbor counter* (DNC) and *triangular counter* (TC), are defined. DNC reports the similarity between a directly adjacent TSV and the victim TSV, and TC reports the number of possible similarities between d_1, d_2, d_3, and d_8. DNC and TC are given by $\frac{X}{4}$ and $\frac{X}{3}$, respectively. Where X shows the similarity of direct

if $d \geq F_{N+1}$ **then**
  $c_N = 1;$
else
  $c_N = 0;$
end if
$r_N = d - c_N \cdot F_{N+1}$
for $k = N - 1$ to 2 **do**
  if $r_{k+1} \geq F_{k+1}$ **then**
    $c_k = 1;$
  else if $r_{k+1} < F_k$ **then**
    $c_k = 0;$
  else
    $c_k = c_{k+1};$
  end if
$r_k = r_{k+1} - c_k \cdot F_k;$
end for
$c_1 = r_2;$
Output: $c_N c_{N-1} \dots c_1$.

Figure 2. Mapping Algorithm of Proposed CAC in the First Phase

neighbor TSVs to d_5 in DNC and possible similarities between d_2, d_1, d_3 and d_8 in TC. As d_5 has four direct neighbors, the denominator in DNC is equal to four, and since, in TW, three data words should be the same as far as possible, the denominator is equal to three.

The flow chart of arranging d_i data words in the second phase is shown in Figure 3. Based on this flow chart, the first step is counting the values of the DNC and TC counters. If DNC $\leq \frac{X}{4}$ or TC $\leq \frac{X}{3}$, then the values of d_2 and d_5 are checked; if they are the same, both d_2 and d_5 are inverted. Otherwise, only the victim TSV, d_5 is inverted dynamically. This mechanism of using CAC with the dynamic inversion of d_2 and/or d_5 is called 3D-Dynamical CAC, or 3D-DyCAC. Through the use of this mechanism, the two required conditions are satisfied.

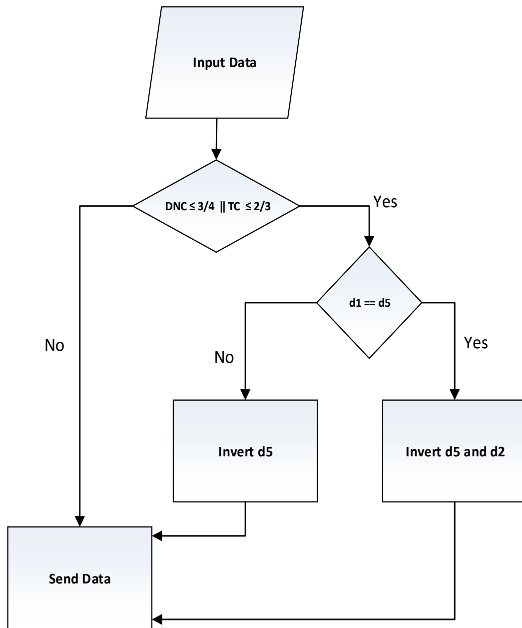

Figure 3. Flowchart of the Second Phase

III. Evaluation and Results

In order to experimentally evaluate the proposed mechanism, the codec of 3D-DyCAC is implemented using VHDL-based simulations and synthesized by Design Compiler in 45 nm technology. To have fair comparisons, the codec of 3DLAT (ω = 4) is implemented, and the results are compared. We assume that the TSVs are arranged in a layout of 3 × *N*. Also, we suppose that the data bandwidth is 64, and thus we need eight 3 × 3 TSV clusters for the data and three clusters for control TSVs. An extra TSV is reserved; since it is required to inform the receiver about the status of inverted TSVs. As shown in Figure 4, results demonstrate, on average, a 10% decrease in the area consumption compared to 3DLAT. Also, 3D-DyCAC can reduce power consumptions of codec in different bandwidths by 23% with respect to 3DLAT, as shown in Figure 5.

To measure the energy consumed per switching event, Figure 6 compares the *power-delay product* (PDP) of 3D-DyCAC with respect to 3DLAT. This comparison confirms that 3D-DyCAC can improve PDP by an average of 46% with respect to 3DLAT. In practice, wire delay and codec critical path are two important parameters in determining performance. Although results are close in low width TSVs, the efficiency of the proposed coding is recognizable with increases in TSV width.

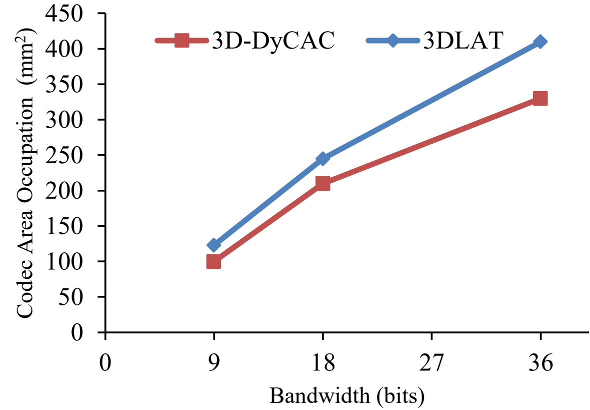

Figure 4. Codec Area Occupations of 3D-DyCAC in Comparison with 3DLAT

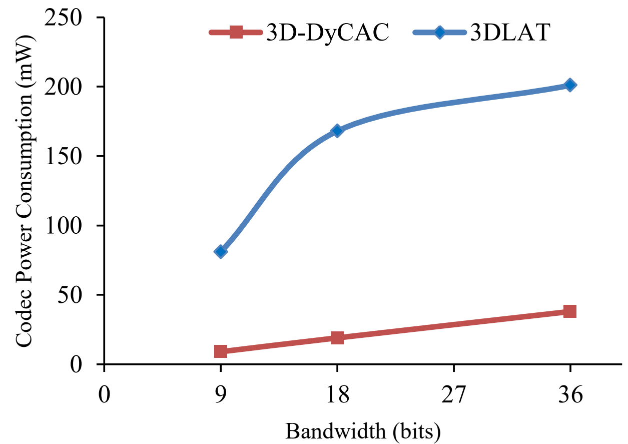

Figure 5. Codec Power Consumption of 3D-DyCAC in Comparison with 3DLAT

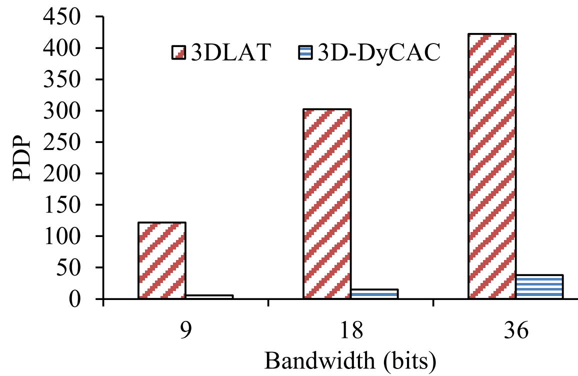

Figure 6. Codec Power Consumption of 3D-DyCAC in Comparison with 3DLAT

IV. Conclusion

To enhance the reliability of 3D ICs, this paper presents a mechanism for avoiding TSV-to-TSV crosstalk faults. This is done by applying a 3D-CAC coding mechanism called 3D-DyCAC. The 3D-DyCAC is applied in two phases. In the first phase, a numerical-system-based CAC generates code words with cumulative groups of *1*s and *0*s to minimize adjacent transitions in arranged meshes of TSVs. This numerical-system-based CAC has no ambiguity in representing code words and generates a unique code word for each data word. In the second phase, a TW is introduced to consider overlaps between the cells of TSVs during the arrangement of code words on TSVs. The TW dynamically inverts the content of the victim TSV and/or the content of adjacent bit positions based on the content of data words. Evaluation results show that 3D-DyCAC reduces the area occupations, power consumption, and critical path of codec by 10%, 24% and 46%, respectively, in comparison to state-of-the-art mechanism 3DLAT.

References

[1] C. S. Tan, "3D Integration for VLSI Systems," Stanford Publishing, 2011.
[2] K. Tu, "Reliability Challenges in 3D IC Packaging Technology," Micro Electronics Reliability (MER), vol. 51, pp. 517–523, 2011.
[3] R. Kumar and S. P Khatri, "Crosstalk Avoidance Codes for 3D VLSI," Proc. IEEE Conf. Design, Automation and Test in Europe (DATE 13), pp. 1673–167, March 2013.
[4] C. J. Akl and M. A. Bayoumi, "Reducing Interconnect Delay Uncertainty via Hybrid Polarity Repeater Insertion," IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol. 16, no. 9, pp. 1230–1239, 2008.
[5] Q. Zou, D. Niu, Y. Cao, and Y. Xie, "3DLAT: TSV-based 3D ICS Crosstalk Minimization Utilizing Less Adjacent Transition Code," Proc. IEEE Conf. Asia and South Pacific Design Automation Conference (ASP-DAC 14), pp. 762-767, January 2014.
[6] Y. Ying Chang, Y. S. Huang, V. Narayanan, and C.T. King, "Shieldus: A Novel Design of Dynamic Shielding for Eliminating 3D TSV Crosstalk Coupling Noise," Proc. IEEE Conf. Asia and South Pacific Design Automation (ASP-DAC 13), pp. 675-680, July 2013.
[7] A. Eghbal, P. M. Yaghini, and N. Bagherzadeh, "Capacitive Coupling Mitigation for TSV-based 3D ICs," Proc. VLSI Test Symposium (VTS 15) l, pp. 1-6, Apri 2015.
[8] M. Khayambashi, P. M. Yaghini, A. Eghbal, and N. Bagherzadeh, "Analytical Reliability Analysis of 3D NoC Under TSV Failure," ACM Journal on Emerging Technologies in Computing Systems, vol. 11, pp. 43-48.
[9] Y. Y. Chang, Y. S. C. Huang, V. Narayanan, and C. T. King, "ShieldUS: A Novel Design of Dynamic Shielding for Eliminating 3D TSV Crosstalk Coupling Noise," Proc. IEEE Conf. Design Automation Conference (ASP-DAC 13), pp. 675-680, January 2013.
[10] A. Eghbal, P. M. Yaghini, S. S. Yazdi, and N. Bagherzadeh, "TSV-to-TSV Inductive Coupling-aware Coding Scheme for 3D Network-on-Chip," Proc. Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT 14), pp. 92-97, October 2014.
[11] W. N. Flayyih, K. Samsudin, S. J. Hashim, Y. I. Ismail, and F. Z. Rokhani, "Adaptive Multibit Crosstalk-Aware Error Control Coding Scheme for On-Chip Communication," IEEE Trans. Circuits and Systems II: Express Briefs, vol. 63, no. 2, pp. 166-170, February 2016.
[12] Y. Zhang, B. Li, B. Zhang and D. Xue, "Novel Crosstalk Minimization Code for 3D IC," Proc. China Semiconductor Technology International Conf. (CSTIC 15), July, pp. 1-3.
[13] Z. Shirmohammadi, N. Rohbani and S. G. Miremadi, "3D-DPS: An Efficient 3D-CAC for Reliable Data Transfer in 3D ICs," Proc. IEEE Conf. European Dependable Computing Conference (EDCC 16), pp. 97-107, September 2016.
[14] Z. Shirmohammadi, F. Mozafari and S. G. Miremadi, "An Efficient Numerical-Based Crosstalk Avoidance Codec Design for NoCs," Microprocessors and Microsystems (MICPRO), vol 50, 2017.
[15] Z. Mahdavi, Z. Shirmohammadi and S. G. Miremadi, "ACM: Accurate Crosstalk Modeling to Predict Channel Delay in Network-on-Chips," Proc. IEEE International Symp. On-Line Testing and Robust System Design (IOLTS 16), pp. 7-8, July 2016.
[16] Z. Shirmohammadi, M. Ansari, S. Kazemian Abhari, S. Safari, S. G. Miremadi, "PAM: a Packet Manipulation Mechanism for Mitigating the Crosstalk Faults in NoCs," Proc. IEEE Conf. Dependable, Autonomic and Secure Computing (DASC 15), pp. 1895- 1902, September 2015.
[17] Z. Shirmohammadi, S. G. Miremadi, "Addressing NoC Reliability through an Efficient Fibonacci-Based Crosstalk Avoidance Codec Design," Proc. IEEE Conf. Algorithms and Architectures for Parallel Processing (ICA3PP 15), pp. 756-770, October 2015.
[18] Z. Shirmohammadi, S. G. Miremadi, "Using Binary Reflected Gray Coding for Crosstalk Mitigation of Network on Chip," Proc. International Symp. Computer Architecture & Digital Systems (CADS 15), pp. 81-86, October 2013.
[19] Z. Shirmohammadi and S. G. Miremadi, "Crosstalk Avoidance Coding for Reliable Data Transmission of Network on Chips," Proc. IEEE Symp. System-on Chip 2013 (SoC 13), Tampere, Finland, pp. 1-4, October 2013.
[20] Z. Shirmohammadi and S. G. Miremadi, "S2AP: An Efficient Numerical-based Crosstalk Avoidance Code for Reliable Data Transfer of NoCs," Proc. IEEE Symp. Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC 15), June - pp. 1-6, July 2015.
[21] Z. Shirmohammadi, S. G. Miremadi, "Addressing NoC Reliability Through an Efficient Fibonacci-based Crosstalk Avoidance Codec Design," Microelectronics Reliability (MR), vol 63, pp. 304-313, 2016.