An Optimal Probing Method of Pre-Bond TSV Fault Identification in 3D Stacked ICs

Bei Zhang and *Vishwani D. Agrawal*
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849, USA
bzz0004@auburn.edu, vagrawal@eng.auburn.edu

***Abstract*— A fast TSV identification algorithm is proposed in this work to reduce the test time of pre-bond TSV probing. The speeding up of the algorithm comes from two aspects. First, any unnecessary session during the test is skipped. Second, the test terminates as soon as either all TSVs have been identified or a pre-specified maximum number of faulty TSVs have been identified. Experimental results demonstrate that instead of testing all sessions as stated in previous work, the algorithm always finishes the pre-bond TSV test after only a small portion of all sessions. The algorithm reduces pre-bond TSV test time and is expected to greatly reduce the pre-bond testing and the overall 3D device manufacturing costs.**

I. Introduction

Pinpointing TSV resistive defects before die bonding is important for yield assurance of 3D ICs. Current BIST based approaches [1], [2], [3], [10] for pre-bond TSV test require dedicated test circuit to be added to each TSV and the area overhead could be huge since there can be tens of thousands of TSVs on a chip [4]. Moreover, the BIST circuits themselves are prone to the effects of process variations. References [7] and [8] propose the use of a large probe needle with active driver to short many TSVs together and conduct a parametric test. The set of TSVs contacted at once by the probe needle is referred to as a *TSV network*. The number of TSVs within a network is typically less than 20 based on the relative diameter and pitch of probe needles and TSVs [4], [9]. High measurement resolution, low hardware overhead and robustness to process variations make this technique likely to be used in practice.

Figure 1 shows a circuit model of the test set up [7], [8] for a 4-TSV network. TSV i is represented by its resistance R_i and capacitance C_i. R_c represents the contact resistance between TSV and the probe. A gated scan flip-flop (GSF) is inserted between TSV i and the system logic which is in accordance with the developing IEEE P1838 standard [5], [8]. All GSFs can be loaded up or read out through a boundary scan mechanism. In the normal mode, all GSFs are made transparent by opening B2 and closing B1 switches by a "bypass" signal. In the pre-bond TSV test mode, all GSFs drive respective TSVs. In Figure 1, TSVs 1 and 2 are receiving TSVs and TSVs 3 and 4 are sending TSVs. A receiving TSV receives signal from the other die and drives the on-chip logic while a sending TSV is driven by the on-chip logic and sends a signal to the other die. A GSF in scan mode drives a receiving TSV during pre-bond TSV probing

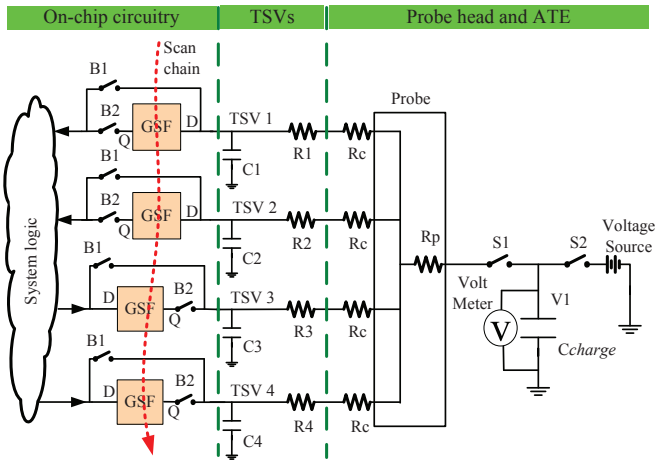

Fig. 1. Circuit model for pre-bond TSV probing.

when both B1 and B2 switches are closed. A GSF drives a sending TSV when B1 is opened and B2 is closed.

Pre-bond TSV resistance measurement starts by scanning in all GSFs with "1." C_{charge} and all TSVs are then discharged through the probe needle. By configuring the switches of a GSF, a charge sharing circuit is constructed between that GSF and C_{charge} through its corresponding TSV (either sending or receiving TSV). The charging rate of C_{charge} is compared to a calibrated curve of a good TSV to determine the resistance of the TSV under test. Parallel TSV test can also be conducted by configuring multiple GSFs at a time. Now, C_{charge} is charged faster and the measurement terminates quicker. However, the number of TSVs tested in parallel cannot exceed a constant "r" due to minimum measurement resolution constraint [7], [8]. This resolution of measurement refers to the minimum change in TSV resistance that can be detected by the technique and it is adversely affected by the number of TSVs tested in parallel. We call the TSVs tested in parallel within the same TSV network a *test session*. Based on this probing technique, any faulty TSV within a session will cause the session test to fail but we cannot tell which TSV(s) is (are) faulty. On the other hand, a good parallel test implies that all TSVs within the session are fault-free. The probe needle remains contacted with the corresponding TSV network, and the charging and discharging process continues until either all TSVs within the network are identified or a certain

number of TSVs with resistive defects are pinpointed within the network. All TSV networks are tested in two groups. Networks in a group are tested simultaneously by a probe head containing a large number of needles, each making contact with a single network. Once all contacted networks are tested, the probe head is lifted and repositioned to test the remaining group [7], [8].

II. A FAST TSV IDENTIFICATION ALGORITHM

The goal of TSV probing is to identify up to a certain number, m, of faulty TSVs within a T TSV network where m is the number of redundant TSVs in the TSV network being tested. If the number of identified faulty TSVs exceeds m, then not all faulty TSVs in the network can be repaired, and the chip would be discarded. Otherwise, the on-chip redundant TSVs are sufficient to replace all identified faulty ones. This goal of TSV probing can be achieved by testing each TSV individually though with unnecessarily long test time. Large time savings occur if we test TSVs in parallel without losing the capability of identifying up to m faulty TSVs, and also guarantee that the size of each session does not exceed r. There are existing methods to generate such sessions [6], [8], [12]. However, those assume that all sessions need to be tested and they do not identify TSVs based on the sessions. This work proposes a fast TSV identification algorithm to further speed up pre-bond TSV test from two aspects. First, during the identification process, any "currently unnecessary" session is skipped. Second, TSV test is terminated as soon as either all TSVs have been identified or the number of identified faulty TSV exceeds m as the chip can be discarded due to lack of redundant TSVs and further test is useless.

The pseudo-code of the algorithm is shown in Figure 2 where argument t represents the test time of sessions. Test time of a session in this work only refers to the charging time of C_{charge}, and it is related to the session size [6], [8], [12]. The algorithm starts by initializing 3 empty lists named "Good", "Bad", and "F_C", respectively. The "Good" and "Bad" lists are used to contain the identified good and faulty TSVs, respectively. The faulty candidate list "F_C" is used to contain any failing session. The algorithm enumerates all the sessions generated by [8] or [12] and skip any "currently unnecessary" session which refers to a session where either all TSVs in the session have been identified so far or there is at least one identified bad TSV in the session. "currently unnecessary" session does not provide any information of TSV identification. We define a *fault map* ρ as positions of defective TSVs within a TSV network. Although a session may be "currently unnecessary" for identifying some fault maps of a TSV network, it could be essential for identifying other fault maps of the same TSV network. So, none of the "currently unnecessary" sessions can be deleted. If a session is not skipped, it will be tested. If a session passes the test, all TSVs in the session are added to "Good", and we then use "Good" to refine "F_C." Here, the refinement

```
Algorithm Fast_TSV_Identification (All_sessions, T, m, t)
1.  Good=[ ]; Bad=[ ]; F_C=[ ]; test_time=0; tested_sessions=0;
2.  foreach session in All_sessions
        // Skip any currently unnecessary test session
3.      if (all TSVs in session have been identified) or
           (there is at least one bad TSV in session)
4.          Continue;
5.      test_time+=t(session);  // Test time accumulation
6.      tested_sessions+=1;     // Test session accumulation
        // Handle a passing session
7.      if session is tested as being good
8.          Add all TSVs in session to Good;
9.          foreach FC_session in F_C
10.             Remove any good TSV from FC_session;
11.             if length(FC_session)==1
12.                 Add the TSV in FC_session to Bad;
13.                 Remove the entire FC_session from F_C;
        // Handle a failing session
14.     else if session is tested as being bad
15.         Remove any good TSV from session;
16.         if length(session)==1
17.             Add the TSV in session to Bad;
18.         else
19.             Append session to F_C;
        // Termination conditions
20.     if ((length(Good)+length(Bad))==T or (length(Bad)>=m+1)
21.         Break;
22. Return  test_time, tested_sessions;
```

Fig. 2. A dynamically optimized TSV identification algorithm.

refers to removing any identified good TSV from the targeted session (see line 10 of Figure 2). If after refinement any failing session in "F_C" contains only one TSV, that TSV is identified as defective and added to "Bad." If a session fails the test, "Good" is again utilized to refine this failing session (line 15 of Figure 2). If the session after refinement contains only one TSV, that TSV is added to "Bad." Otherwise, the refined failing session is appended to "F_C." The above procedure terminates as soon as any condition shown on line 20 in Figure 2 is satisfied.

III. EXPERIMENTAL RESULTS

Table I shows the results of the proposed algorithm applied to various TSV networks. Column 1 shows parameters T (network size), m (redundant TSVs in network) and r (resolution). Column 2 gives the number of faulty TSVs (ϕ) within the network. Column 3 shows the total number of sessions and total test time (in μs) for exhaustive application of sessions optimized by ILP [12]. The test time calculation is detailed in references [6], [8] and [12]. For given ϕ, we enumerate all possible fault maps and obtain the test time and number of tested sessions using the proposed algorithm of Figure 2. Column 4 shows the average number of tested sessions and average test time for identifying all fault maps containing ϕ faulty TSVs. Column 5 shows the relative reduction in column 4 over column 3. Column 6 shows the maximum number of sessions tested and the corresponding test time for identifying a fault map. Column 7 shows the relative reduction in column 6 over column 3.

TABLE I
EXHAUSTIVE [12] AND DYNAMICALLY OPTIMIZED (FIGURE 2) APPLICATION OF TSV TEST SESSIONS CONSTRUCTED BY ILP.

Parameters T, m, r	Number of faulty TSVs (ϕ)	Optimum exhaustive test [12] (# sessions, time in μs)	Dynamically optimized test			
			Av. test sessions (# used, time in μs)	Average reduction (sessions, time)	Worst case sessions (# used, time in μs)	Worst case reduction (sessions, time)
8, 2, 3	0	(8, 3.36)	(5.0, 2.10)	(37.5%, 37.5%)	(5, 2.10)	(37.5%, 37.5%)
	1		(5.3, 2.25)	(32.8%, 32.8%)	(6, 2.52)	(25.0%, 25.0%)
	2		(6.4, 2.71)	(19.1%, 19.1%)	(8, 3.36)	(0.0%, 0.0%)
	3		(7.5, 3.17)	(5.3%, 5.3%)	(8, 3.36)	(0.0%, 0.0%)
12, 3, 3	0	(16, 6.72)	(7.0, 2.94)	(56.2%, 56.2%)	(7, 2.94)	(56.2%, 56.2%)
	1		(7.5, 3.14)	(53.1%, 53.1%)	(9, 3.78)	(43.7%, 43.7%)
	2		(8.7, 3.65)	(45.5%, 45.5%)	(12, 5.04)	(25.0%, 25.0%)
	3		(10.3, 4.32)	(35.5%, 35.5%)	(14, 5.88)	(12.5%, 12.4%)
	4		(11.8, 4.97)	(25.9%, 25.9%)	(16, 6.72)	(0.0%, 0.0%)
15, 4, 3	0	(25, 10.50)	(8.0, 3.36)	(68.0%, 68.0%)	(8, 3.36)	(68.0%, 68.0%)
	1		(9.6, 4.03)	(61.6%, 61.6%)	(14, 5.88)	(44.0%, 44.0%)
	2		(11.1, 4.68)	(55.3%, 55.3%)	(17, 7.14)	(32.0%, 32.0%)
	3		(12.6, 5.33)	(49.2%, 49.2%)	(20, 8.40)	(20.0%, 20.0%)
	4		(14.3, 6.03)	(42.5%, 42.5%)	(23, 9.66)	(8.0%, 8.0%)
	5		(15.8, 6.66)	(36.5%, 36.5%)	(25, 10.50)	(0.0%, 0.0%)
20, 4, 4	0	(25, 9.50)	(9.0, 3.42)	(64.0%, 63.9%)	(9, 3.42)	(64.0%, 63.9%)
	1		(10.8, 4.10)	(56.8%, 56.7%)	(15, 5.69)	(40.0%, 39.9%)
	2		(12.3, 4.68)	(50.6%, 50.6%)	(18, 6.83)	(28.0%, 27.9%)
	3		(13.9, 5.31)	(44.0%, 44.0%)	(21, 7.97)	(16.0%, 15.9%)
	4		(15.1, 5.76)	(39.3%, 39.3%)	(24, 9.11)	(4.0%, 3.9%)
	5		(18.0, 6.85)	(27.8%, 27.8%)	(25, 9.49)	(0.0%, 0.0%)

We make four observations from Table I. First, the average number of tested sessions and average test time is much less than the total number of sessions and total test time for any $\phi \leq m$ (reparable TSV network) or any $\phi > m$ (irreparable TSV network). For example, the average percentage reduction reaches 68.0% for parameters $T = 15$, $m = 4$, $r = 3$, and $\phi = 0$. On average, the proposed algorithm greatly speeds up pre-bond TSV identification process. Second, as ϕ increases the average percentage reduction decreases. This is expected as pinpointing larger number of faulty TSVs within a TSV network generally requires more sessions to be tested and cost more time. Third, in most cases even the maximum number of tested sessions is less than the total number of sessions. Fourth, as expected, the maximum number of tested sessions increases as ϕ increases for a given TSV network. From column 7, reduction in the worst case can be small for large ϕ, requiring all sessions to identify a fault map. This scenario occurs when fault map contains m or more faulty TSVs. The probability of such large number of faulty TSVs within a small localized silicon area may be negligible for a mature manufacturing process. Thus, the worst case percentage test time reduction could be quite significant.

IV. CONCLUSION

The proposed TSV identification algorithm has two main advantages. First, the average number of tested sessions and test time are guaranteed to be small factions of total sessions and test time. Second, even for the worst fault map, for which most sessions are needed, not all sessions may be used. Reducing pre-bond TSV test time helps reduce pre-bond test cost in real silicon. A forthcoming paper combines this technique with several related optimizations [11].

Acknowledgment This research is supported in part by the National Science Foundation Grant CCF-1116213.

REFERENCES

[1] P. Chen, C. Wu, and D. Kwai, "On-Chip Testing of Blind and Open-Sleeve TSVs for 3D IC Before Bonding," in *Proc. 28th IEEE VLSI Test Symposium (VTS)*, 2010, pp. 263–268.

[2] M. Cho, C. Liu, D. H. Kim, S. K. Lim, and S. Mukhopadhyay, "Design Method and Test Structure to Characterize and Repair TSV Defect Induced Signal Degradation in 3D System," in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2010, pp. 694–697.

[3] S. Deutsch and K. Chakrabarty, "Non-Invasive Pre-Bond TSV Test using Ring Oscillators and Multiple Voltage Levels," in *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2013, pp. 1065–1070.

[4] M. Jung, J. Mitra, D. Z. Pan, and S. K. Lim, "TSV Stress-Aware Full-Chip Mechanical Reliability Analysis and Optimization for 3D IC," in *Proc. 48th Design Automation Conference (DAC)*, 2011, pp. 188–193.

[5] E. J. Marinissen, J. Verbree, and M. Konijnenburg, "A Structured and Scalable Test Access Architecture for TSV-Based 3D Stacked ICs," in *Proc. 28th IEEE VLSI Test Symposium (VTS)*, 2010, pp. 269–274.

[6] B. Noia and K. Chakrabarty, "Identification of Defective TSVs in Pre-Bond Testing of 3D ICs," in *Proc. 20th Asian Test Symposium (ATS)*, 2011, pp. 187–194.

[7] B. Noia and K. Chakrabarty, "Pre-Bond Probing of TSVs in 3D Stacked ICs," in *Proc. International Test Conference (ITC)*, 2011, pp. 1–10.

[8] B. Noia and K. Chakrabarty, *Design-for-Test and Test Optimization Techniques for TSV-based 3D Stacked ICs*. Springer, 2014.

[9] K. Smith, P. Hanaway, M. Jolley, R. Gleason, and E. Strid, "Evaluation of TSV and Micro-Bump Probing for Wide I/O Testing," in *Proc. International Test Conference (ITC)*, 2011, pp. 1–10.

[10] J. You, S. Huang, D. Kwai, Y. Chou, and C. Wu, "Performance Characterization of TSV in 3D IC via Sensitivity Analysis," in *Proc. 19th Asian Test Symposium (ATS)*, 2010, pp. 389–394.

[11] B. Zhang and V. D. Agrawal, "SOS3: Three-Step Optimization of Pre-Bond TSV Test for 3D Stacked ICs," in *Proc. 32nd IEEE Int. Conf. on Computer Design (ICCD)*, Oct. 2014.

[12] B. Zhang and V. D. Agrawal, "Test Session Optimization for Pre-Bond TSV Probing in 3D Stacked ICs," in *Proc. 23rd Asian Test Symposium (ATS)*, 2014. submitted.