Test-Architecture Optimization for 3D Stacked ICs

Brandon Noia and Krishnendu Chakrabarty

I. Problem and Motivation

TSV-based 3D-SICs significantly impact core-based system-on-chip (SOC) design. Testing of core-based dies in 3D-SICs introduces new challenges [1], [2]. In order to test the dies in a stack, the embedded cores, and the TSVs, a test access mechanism (TAM) must be included on the die to transport test data to the cores, and a 3D TAM is needed to transfer test data to the die from the stack input/output pins. In a 3D-SIC, a test architecture must be able to support testing of individual dies as well as testing of partial and complete stacks [2]. Furthermore, test-architecture optimization must not only minimize the test time, but also minimize the number of dedicated test TSVs used to route the 3D TAM, as each TSV has area costs associated with it and is a potential source of defects. Test bandwidth constraints due to a limited number of package pins available only at the lowest layer in a 3D stack must also be considered.

Compared to two-dimensional ICs that typically require two test insertions, namely wafer test and package test, 3D stacking introduces a number of natural test insertions [2]. Since the die-stacking steps of thinning, alignment, and bonding can introduce defects both to each die itself and the vertical connections between the dies, there is a need to test multiple subsequent (partial) stacks during assembly.

In this summary, we focus on the problem of test-architecture optimization and test scheduling for soft-dies considering multiple post-bond test insertions. Our work has covered test-architecture optimization with hard dies, firm dies, and post-bond die-external tests as well [3], [4], though space limitations prevent discussion of that work here. For the soft-die optimization problem considering multiple test insertions, which we will refer to as P^S_{MTS}, we allow the 2-D test architecture on each die to be co-optimized with the 3D TAM architecture and test schedule. Compared to the hard and firm die models, P^S_{MTS} is the most computationally difficult.

The key contributions of the work presented in this summary are as follows:

- Generalized and rigorous optimization methods to minimize test time for multiple test insertions. These methods also provide optimal solutions for several additional problem instances of interest, e.g., final stack test only, and testing of any subset of partial stacks. The optimization models are rigorously derived and their optimality is formally established in [3], though these results are not included in this summary due to space constraints.
- Methods to support multiple test schedules for a test-access architecture that is defined to be globally optimal for multiple test insertions.
- Optimization methods that are compatible with the proposed die-wrapper architecture [5] being considered by the IEEE P1838 test workgroup, instead of requiring unrealistic assumptions on a specific wrapper for each case.

II. Background and Related Work

As interest in 3D-SICs has increased, many papers on the topic of 3D-SIC testing have been published. Heuristic methods for designing core wrappers in 3D ICs were developed in [6]. ILP models for test architecture design for each die in a stack are presented in [7]. While these ILP models take into account some of the constraints related to 3D-SIC testing such as the TSV limit, this approach does not consider the reuse of die-level TAMs, multiple test insertions, or TSV tests. A TAM wire-length minimization technique based on simulated annealing is presented in [8]. While this work allows both pre-bond and post-bond tests, TAMs can start and end on any stack tier, which is unlikely in a 3D-SIC. Heuristic methods for reducing weighted test cost while taking into account the constraints on test pin widths in pre-bond and post-bond tests are described in [9], but it still fails to allow for multiple schedules in multiple test insertions and die-external test during stacking.

In most prior work on 3D-SIC testing, TAM optimization is performed at die-level only, which leads to inefficient TAMs and non-optimal test schedules for partial/complete stack test. Furthermore, all previous methods assume that the designer can create TAM architectures on each die during optimization, which may not be possible in all cases. In [5] a die-level wrapper and associated 3D architecture is presented to allow for all pre-bond and post-bond tests. This approack relies on die-level wrappers and it leverages current standards, IEEE 1149.1 and IEEE 1500. In addition to functional and test modes, die-level wrappers allow bypass of test data to and from higher die in the stack and reduced test bandwidth during pre-bond tests. This is a realistic and practical look at test architectures in 3D-SICs, but it offers no insight into optimization and test scheduling. The optimization methods presented in this summary are compatible with the test architecture of [5], and they do not make any unrealistic assumptions on die wrappers or the 3D TAM.

This summary includes only a small subset of the totality of the solutions we have provided for the 3D test-architecture optimization problem. Work regarding hard die, firm die, die external tests, proofs of correctness, and more can be found in [3], [4]. These two papers can also be used as reference for those unfamiliar with the concepts in this summary. To provide more insight into the P^S_{MTS} problem, we provide an optimzation example below.

A. Example for P^S_{MTS}

In a 3D IC, the lowest die is usually directly connected to chip I/O pins, therefore it can be tested using package pins.

To test the other dies in the stack, TAMs that enter the stack from the lowest die should be provided. To transport test data up and down the stack, "test elevators" need to be included on each die except for the highest die in the stack [2]. The number of test pins and test elevators, as well as the number of TSVs used, affect the total test time for the stack as well as the area overhead associated with test. All test data to dies higher in the stack must be routed through each die lower in the stack, and all test data enters and exits the stack through external test pins on the lowest die in the stack.

It can be shown that optimizing only for the final stack test does not result in optimum test times for multiple test insertions. Consider an SIC with three dies from the ITC'02 SOC Benchmarks [10] as shown in Figure 1. There are 40 test pins available to the stack and a limit of 40 TSVs per side of each die as the maximum number of TSVs to use for the test infrastructure. The test time for each die is determined by its test architecture, which relies on Daisychaining in this example.

Figure 1(a) shows the resulting test elevator widths and the number of TSVs used if the stack is optimized to reduce the test time of the final stack test after all die have been bonded. This architecture allows for the testing of all three die in parallel, which results in a final stack test time of 1313618 cycles with individual die test times of 1313618, 1303387, and 658834 cycles, from top to bottom. The architecture uses 38 out of the 40 allowed test pins and 64 TSVs. When both possible stack tests are considered, with the first stack test done after the middle die is bonded to the lower die and the final stack test, the total test time becomes 2617005 cycles (Die 1 and Die 2 are tested in parallel for the first stack test).

Figure 1(b) shows the test architecture created if all stack tests are considered during optimization, which uses all 40 test pins and 62 TSVs. This architecture allows Die 1 and Die 2 to be tested in parallel for the first stack test, and then Die 2 and Die 3 to be tested in parallel after Die 1 is tested for the final stack test. The test times for the dies from top to bottom are 1338480, 700665, and 75004, respectively. This results in a final stack test time of 1413484 cycles, an increase in time over the previous example. However, when we consider all stack tests, this architecture results in a total test time of 2114149 cycles (700665 cycles for the first stack test), a considerable saving over the previous example. This example clearly shows the impact of the optimization target on the architecture and test time. Therefore, a test-architecture optimization algorithm for 3D-SICs has to minimize the test time while taking into account 3D design constraints as well as all of the post-bond tests that will be run on the 3D-SIC. The resulting optimized 3D TAM must also allow for different test schedules for each post-bond test.

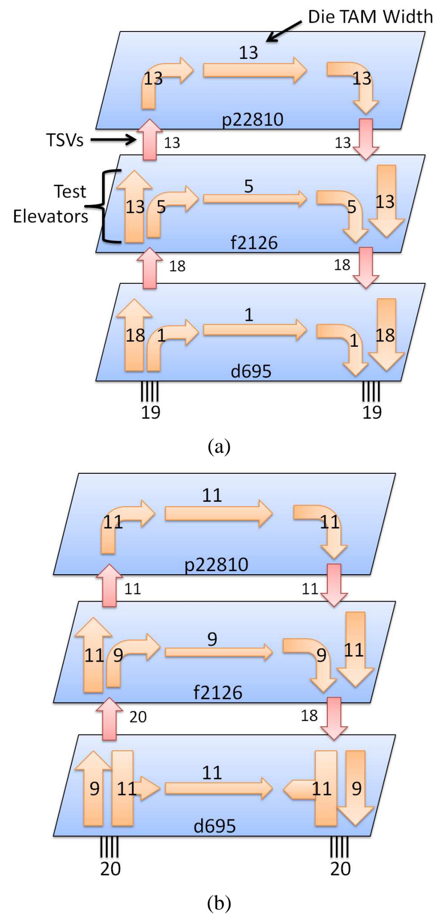


Fig. 1: Example 2: 3D-SIC with three soft dies.

III. Approach and Uniqueness

We formally define the P^S_{MTS} problem as follows: We are given a stack with a set M of dies and the total number of test pins W_{max} available for test. For each die $m \in M$, we are given its tier number l_m in the stack, a maximum number of TSVs ($TSVmax_m$) that can be used for TAM design between die $m-1$ and m ($m > 1$), and the total number of cores c_m. Furthermore, for each core n, the number of inputs i_n, outputs o_n, total number of test patterns p_n, total number of scan chains s_n, and for each scan chain k, the length of the scan chain in flip flops $l_{n,k}$ are given. The goal is to determine an optimal TAM design and test schedule for each stage of stacking, as well as for each die, such that the total test time T for the stack is minimized and the number of TSVs used per die does not exceed $TSVmax_m$.

This problem is $\mathcal{NP}$-hard from "proof by restriction" [11], as it can be reduced using standard techniques to the rectangle packing problem, which is known to be $\mathcal{NP}$-hard [12]. For example, if we remove the constraints related to maximum number of TSVs and consider only the final stack test insertion, each die can be represented as a set of rectangles with different widths and heights, where width is equal to its test time for a given TAM width and height equal to the number of required test pins. Now we need to pack all these rectangles (dies) into a bin with width equal to the total number of test pins and height equal to the total test time for the stack, which needs to be minimized.

We solve the P^S_{MTS} using Integer Linear Programming (ILP). In ILP, a problem is defined as an objective function to be minimized or maximized subject to a serires of linear constraints. ILP produces provably optimal results to the problem as defined. The difficulty of using ILP to solve an $\mathcal{NP}$-hard problem such as P^S_{MTS} is in appropriately defining

the problem in such a way that solution times remain reasonable. We have managed to define the problem such that its complexity largely grows with the number of dies in the 3D IC, and only grows only slightly with increasing TSV count, test pin count, and so forth. Thus, P^S_{MTS} can be solved optimally in reasonable time since the number of layers in a 3D-SIC is expected to be limited, e.g., up to four layers have been predicted for logic stacks [13]. A significent benefit of using ILP beyond optimal results is that integrating new parameters, for example thermal or power constraints, into the problem is as simple as adding more contraints to the problem definition.

A. ILP Formulation for Problem P^S_{MTS}

To create an ILP model for this problem, we need to define the set of variables and constraints. We first define a binary variable x_{ijk}, which is equal to 1 if die i is tested in parallel with die j for a test insertion when there are k die in the stack, and 0 otherwise. There are $M-1$ test insertions, one for each additional die added to the stack such that k ranges from 2 to M. Constraints on variable x_{ijk} can be defined as follows:

$$x_{iik} = 1 \qquad \forall k, i \leq k \tag{1}$$

$$x_{ijk} = x_{jik} \qquad \forall k, \{i,j\} \leq k \tag{2}$$

$$1 - x_{iqk} \geq x_{iqk} - x_{jqk} \geq x_{ijk} - 1 \qquad \forall k, \{i,j,q\} \leq k, i \neq j \neq q \tag{3}$$

The first constraint indicates that every die is always considered to be tested with itself for evert test insertion. The second constraint states that if die i is tested in parallel with die j for insertion k, then die j is also tested in parallel with die i for insertion k. The last constraint ensures that if die i is tested in parallel with die j for insertion k, then it must also be tested in parallel with all other dies that are tested in parallel with die j for insertion k.

Next, we define a second binary variable y_{ik}, which is equal to 0 if die i is tested in parallel with die j on a lower layer ($l_i > l_j$) for insertion k, and 1 otherwise. The total test time T for the stack is the sum of test times of all dies that are tested in series plus the maximum of the test times for each of the sets of parallel tested dies for all test schedules at every test insertion. The test time t_i for die i is a function of the TAM width w_i assigned to it. Using variables x_{ijk}, y_ik, and t_i, the total test time T for all test insertions with the set of dies M can be defined as follows.

$$T = \sum_{k=2}^{|M|} \sum_{i=1}^{k} y_{ik} \cdot \max_{j=i..k} \{x_{ijk} \cdot t_j(w_j)\} \tag{4}$$

It should be noted that Equation (4) has several non-linear elements. To linearize this equation, first we must define the test time function. For this purpose, we introduce a binary variable $g_{in} = 1$ if $w_i = n$, and 0 otherwise. We then linearize this expression using the variable v_{ijk} for $x_{ijk} \cdot \sum_{n=1}^{k_i} (g_{jn} \cdot t_j(n))$. It should be noted that, although the variable x_{ijk} can change for different test insertions depending on the test schedule for each insertion, there must be a single value of test pins used for each die to reflect an architecture that can support all test schedules used throughout all test insertions. The variable c_{ik} takes the value of the max function for each die i for each test insertion k and the variable u_{ik} represents the product $y_{ik} \cdot c_{ik}$. Since w_j is now a decision variable, we linearize $x_{ijk} \cdot w_j$ using a new variable z_{ijkj}. We represent the max function by the variable d_i as before. By using the variable z_{ijkj}, the TAM width that can be given to each die can be constrained by an upper limit, which is the number of available test pins. We represent this with the following inequality.

$$\sum_{j=i}^{k} z_{ijkj} \leq W_{max} \qquad \forall k, i \leq k \tag{5}$$

As number of test pins used for parallel testing of dies should not exceed the given test pins W_{max} across all test schedules for every test insertion, a constraint on the total number of pins used to test all dies in a parallel set in any given test insertion can be defined as follows for all k.

$$\sum_{j=1}^{k} x_{ijk} \cdot w_j \leq W_{max} \qquad \forall i \leq k \tag{6}$$

Similarly, the total number of used TSVs should not exceed the given TSV limit ($TSVmax_m$) for each die face across all test insertions. It should be noted that $TSVmax_2$ is the limit for the upper face of die 1 and the lower face of die 2, $TSVmax_3$ is for the upper face of die 2 and lower face of die 3, and so forth. The number of TSVs used to connect layer i to layer $i-1$ is the maximum of the number of pins required by the layer at or above layer i that takes the most test pin connections, and the sum of parallel tested die at or above layer i in the same parallel tested set across all test insertions. Based on this, we can define the constraint on the total number of TSVs used in a test architecture as follows.

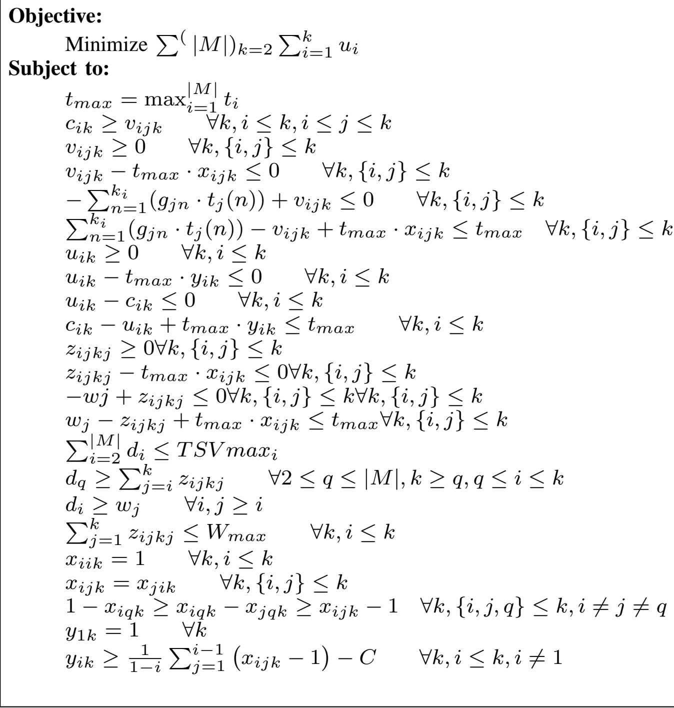
$$\max_{i<=k<=|M|} \{w_i, \sum_{j=i}^{k} w_j \cdot x_{kj}\} \leq TSVmax_i \qquad \forall i \geq 2, \tag{7}$$

We can linearize the above set of constraints by representing the max function by a variable d_i. Finally, to complete the ILP model for Problem P^H_{MTS}, we must define constraints on binary variable y_{ik} and the relationship between binary variable y_{ik} and x_{ijk}. For this purpose, we first define a constant C that approaches but is less than 1. We then define y_{ik} as follows:

$$y_{1k} = 1 \qquad \forall k \tag{8}$$

$$y_{ik} \geq \frac{1}{1-i} \sum_{j=1}^{i-1} (x_{ijk} - 1) - C \qquad \forall k, i \leq k, i \neq 1 \tag{9}$$

The first equation forces y_{1k} to 1, since the lowest layer can not be tested in parallel with any layer lower than itself. Constraint 9 defines y_{ik} for the other layers. To understand this constraint, we first make the observation that the objective function (as shown in Equation (4)) would be minimized if each y_{ik} is zero. This would make the objective function value equal to 0, which is an absolute minimum test time. Thus, we only need to restrict y_{ik} to 1 where it is absolutely necessary, and then we can rely on the objective function to assign a value 0 to all unrestricted y_{ik} variables. This equation considers the range of values that the sum of x_{ijk} can take. The fraction



Objective:

Minimize $\sum^(|M|)_{k=2} \sum_{i=1}^{k} u_i$

Subject to:

$t_{max} = \max_{i=1}^{|M|} t_i$
$c_{ik} \geq v_{ijk} \quad \forall k, i \leq k, i \leq j \leq k$
$v_{ijk} \geq 0 \quad \forall k, \{i,j\} \leq k$
$v_{ijk} - t_{max} \cdot x_{ijk} \leq 0 \quad \forall k, \{i,j\} \leq k$
$-\sum_{n=1}^{k_i} (g_{jn} \cdot t_j(n)) + v_{ijk} \leq 0 \quad \forall k, \{i,j\} \leq k$
$\sum_{n=1}^{k_i} (g_{jn} \cdot t_j(n)) - v_{ijk} + t_{max} \cdot x_{ijk} \leq t_{max} \quad \forall k, \{i,j\} \leq k$
$u_{ik} \geq 0 \quad \forall k, i \leq k$
$u_{ik} - t_{max} \cdot y_{ik} \leq 0 \quad \forall k, i \leq k$
$u_{ik} - c_{ik} \leq 0 \quad \forall k, i \leq k$
$c_{ik} - u_{ik} + t_{max} \cdot y_{ik} \leq t_{max} \quad \forall k, i \leq k$
$z_{ijkj} \geq 0 \forall k, \{i,j\} \leq k$
$z_{ijkj} - t_{max} \cdot x_{ijk} \leq 0 \forall k, \{i,j\} \leq k$
$-wj + z_{ijkj} \leq 0 \forall k, \{i,j\} \leq k \forall k, \{i,j\} \leq k$
$w_j - z_{ijkj} + t_{max} \cdot x_{ijk} \leq t_{max} \forall k, \{i,j\} \leq k$
$\sum_{i=2}^{|M|} d_i \leq TSVmax_i$
$d_q \geq \sum_{j=i}^{k} z_{ijkj} \quad \forall 2 \leq q \leq |M|, k \geq q, q \leq i \leq k$
$d_i \geq w_j \quad \forall i, j \geq i$
$\sum_{j=1}^{k} z_{ijkj} \leq W_{max} \quad \forall k, i \leq k$
$x_{iik} = 1 \quad \forall k, i \leq k$
$x_{ijk} = x_{jik} \quad \forall k, \{i,j\} \leq k$
$1 - x_{iqk} \geq x_{iqk} - x_{jqk} \geq x_{ijk} - 1 \quad \forall k, \{i,j,q\} \leq k, i \neq j \neq q$
$y_{1k} = 1 \quad \forall k$
$y_{ik} \geq \frac{1}{1-i} \sum_{j=1}^{i-1} \left(x_{ijk} - 1\right) - C \quad \forall k, i \leq k, i \neq 1$

Fig. 2: ILP model for 3D TAM optimization Problem P^S_{MTS}.

in the equation normalizes the sum to a value between 0 and 1 inclusive, while the summation considers all possible cases for a die being in parallel with die below it. The complete ILP model for Problem P^S_{MTS} is shown in Figure 2.

We shall illustrate the ILP model for P^S_{MTS} in Figure 3, though for simplicity we fix the TAM width for each die to a single value. The 3D-SIC in Figure 3 consists of three die, with test times of 1333098, 700665, and 106391 cycles moving up the stack. There are 22 test pins available, so W_{max} is 22. TSV_{max} is set to 20, such that there can be no more than 20 dedicated test TSVs between any two die (this limits the TSVs per die to 40). There are two test insertions, the first when Die 1 and Die 2 are stacked and the second for the complete stack. In the first test insertion (k = 2), we calculate the optimal solution that Die 1 and Die 2 are tested in parallel. As such, $x_{1,1,2}$, $x_{1,2,2}$, $x_{2,1,2}$, and $x_{2,2,2}$ are all equal to 1. Die 2 is tested in parallel with a die below it (Die 1), so $y_{2,2}$ is 0 and $y_{1,2}$ is 1. The test time for this test insertion is 1333098 cycles, since $u_{1,2}$ is 1333098 and $u_{2,2}$ is 0.

For the second test insertion (k = 3), the optimal solution is to test Die 1 and Die 2 in parallel, and then test Die 3. Since Die 1 and Die 2 are tested in parallel again, $x_{1,1,3}$, $x_{1,2,3}$, $x_{2,1,3}$, and $x_{2,2,3}$ are equal to 1. For Die 3, $x_{3,3,3}$ is 1. All other x_{ijk} variables are 0. As before, $y_{2,3}$ is 0 since Die 2 is tested in parallel with Die 1. The variables $y_{1,3}$ and $y_{3,3}$ are 1. The test time for this test insertion is 1439489, as $u_{1,3}$ is 1333098, $u_{2,3}$ is 0, and $u_{3,3}$ is 106391. In this architecture, d_2 is 20 and d_3 is 14, neither of which violate the TSV limit. All 44 test pins are utilized.

IV. Results and Contributions

For use as benchmarks, we have handcrafted two 3D SICs, as shown in Figure 4, using several SOCs from the ITC'02

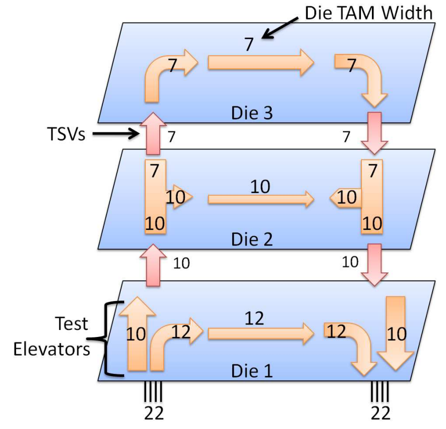


Fig. 3: Example of a test architecture produced from the P^S_{MTS} optimization with fixed die TAM widths.

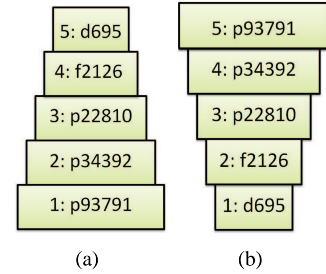


Fig. 4: Two 3D SIC benchmarks.

SOC test Benchmarks as dies inside SICs. The SOCs used are d695, f2126, p22810, p34292, and p93791. In SIC 1, the most complex die (p93791) is placed at the bottom, with die complexity decreasing as one moves up the stack. The order is reversed in SIC 2. To determine the test architecture and test time for a given die (SOC) with a given TAM width, we have used the TAM design method in [14] for daisychain TestRail architectures [15].

Table I compares optimal results produced using ILP with those produced using a greedy algorithm for P^S_{MTS} performed on SIC 1. These results are performed for only the final stack test, or the last post-bond insertion before packaging. Column 1 and Column 2 give the maximum number of dedicated test TSVs allowed and the number of test pins available to the stack, respectively. Column 3 and Column 4 provide the test length in clock cycles and test schedule determined from the optimized architecture for the ILP formulation of P^S_{MTS}. Column 5 and Column 6 provide the same results for the greedy algorithm. Column 7 gives the percent difference between the two test lengths.

The greedy algorithm contains a subroutine that attempts to combine dies in parallel tested sets, starting with those that would lead to the greatest reduction in test time. If more than one combination results in the same reduction, it prioritizes those combinations that result in the smallest test resource use. The complete greedy algorithm utilizes this subroutine, and begins with an assignment of equal number of test pins to each of the dies and optimizes the 2D and 3D TAM under these constraints. It then randomly adds and removes random numbers of test pins from each die, each time balancing the

TSV_{max}	W_{max}	ILP Solution Test Length (cycles)	ILP Solution Test Schedule	Greedy Solution Test Length (cycles)	Greedy Solution Test Schedule	Percentage Difference in Test Length
140	30	4795930	1‖2‖3‖4,5	7842000	1‖2,3,4‖5	63.5
140	35	4237100	1‖2‖3‖4,5	7633580	1‖3,2‖4,5	80.1
140	40	3841360	1‖2‖3‖4,5	6846400	1‖3,2‖4,5	78.2
140	45	3591550	1‖2‖3‖4,5	6379510	1‖2‖3,4‖5	77.6
140	50	3090720	1‖2‖3‖4,5	6041270	1‖2‖3,4‖5	95.5
140	55	2991860	1‖2‖3‖4‖5	5873430	1‖2‖3‖4,5	96.3
140	60	2873290	1‖2‖3‖4,5	5821900	1‖2‖3‖4,5	102.6
140	65	2784050	1‖2‖3‖4‖5	5705410	1‖2‖3‖4,5	104.9
140	70	2743320	1‖2‖3‖4‖5	5638140	1‖2‖3‖4,5	105.5
140	75	2629500	1‖2‖3‖4‖5	5638140	1‖2‖3‖4,5	114.4
140	80	2439380	1‖2‖3‖4‖5	5496200	1‖2‖3‖4,5	125.3
140	85	2402330	1‖2‖3‖4‖5	5447190	1‖2‖3‖4,5	126.7
140	90	2395760	1‖2‖3‖4‖5	5447190	1‖2‖3‖4,5	127.4
140	95	2383400	1‖2‖3‖4‖5	5447190	1‖2‖3‖4,5	128.5
140	100	2369680	1‖2‖3‖4‖5	5351480	1‖2‖3‖4,5	125.8

TABLE I: Comparison of optimization results between P^S_{MTS} and a greedy algorithm for SIC 1.

result to use the maximum number of test pins, and optimizes again. It checks for reductions in test time, returning to the best solution so far if no test time reduction is produced or constraints are violated. It terminates after 10000 iterations of no improvement.

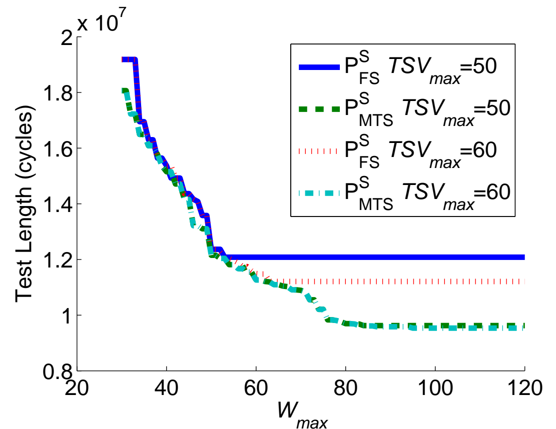
As can be seen in Table I, the optimal ILP solution tends to be much better in terms of test length than the heuristic solution, simply due to the size of the solution space and the difficulty of breaking the problem into smaller optimal choices that a greedy algorithm can exploit. Compared to a greedy algorithm, the P^S_{MTS} ILP formulation produces significantly faster test lengths, often with an improvement above 100%.

Figure 5 demonstrates the benefits of an optimization method that can take into account multiple test insertions. Figure 5 shows test length resuts for a fixed TSV_{max} value and varying test pint number (W_{max}). The test length is determined for all post-bond test insertions. Results are presented for the P^S_{MTS} optimization presented in this summary and an ILP solution that only considers the final stack test (P^S_{FS}). Figure 5(a) shows results for SIC 1, and Figure 5(b) shows results for SIC 2.

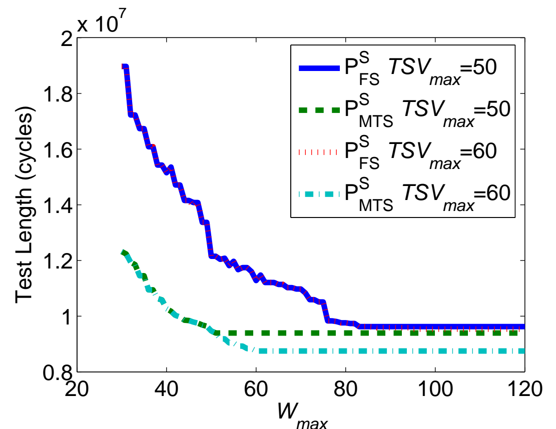
As can be seen from Figure 5, both TSV_{max} and W_{max} determine which dies should be tested in parallel, and thus the total test time for the stack. We see that optimizing for the final stack test does not always reduce test time when we consider multiple test insertions. In fact, we see cases where the test time is higher when we increase the values of TSV_{max} and W_{max}, simply due to the final stack optimization being sub-optimal when considering multiple test insertions. For P^S_{MTS}, we see continued reductions in test time until increasing W_{max} no longer impacts the result.

For optimization of 3D-SICs with soft dies, Pareto-optimality is almost non-existent when W_{max} is varied, which is not the case when considering dies with fixed 2D TAMs. This is due to the fact that as dies in the stack are soft, it is almost always possible to find one die for which adding an extra test pin reduces the overall test time. When test time stops decreasing, the TSV limits for the problem instance have been reached and no further optimization is possible.

The results presented in this summary show the significant contributions of this work. Utilizing a well-developed ILP problem definition, optimal results can be produced for a difficult problem of 2D and 3D TAM and test schedule co-optimization for any and all post-bond test insertions. These results are drastically better than a greedy approach to the problem solution. This work is currently the definitive work on post-bond test architecture optimization for 3D ICs with die wrappers. Further optimizations and results for more varied problems, as well as proofs of correctness for P^S_{MTS} presented in this summary, can be found in [3], [4].



(a) SIC 1



(b) SIC 2

Fig. 5: The test time with respect to TSV_{max} and W_{max} for SIC 1 and SIC 2 with soft dies and multiple test insertions.

REFERENCES

[1] H.-H. S. Lee and K. Chakrabarty, “Test Challenges for 3D Integrated Circuits”, *IEEE Design & Test of Computers*, vol. 26, pp. 26-35, September/October 2009.

[2] E.J. Marinissen and Y. Zorian, “Testing 3D Chips Containing Through-Silicon Vias”, *International Test Conference*, E 1.1, 2009.

[3] B. Noia, K. Chakrabarty, S. K. Goel, E. J. Marinissen, and J. Verbree,“Test-Architecture Optimization and Test Scheduling for TSV-Based 3D Stacked ICs”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, pp. 1705-1718, November 2011.

[4] B. Noia, K. Chakrabarty and E. J. Marinissen, “Optimization Methods for Post-Bond Testing of 3D Stacked ICs”, *Journal of Electronic Testing: Theory and Applications*, vol. 28, pp. 103-120, February 2012.

[5] E.J. Marinissen, J. Verbree, and M. Konijnenburg, “A Structured and Scalable Test Access Architecture for TSV-Based 3D Stacked ICs”, *VLSI Test Symposium*, 2010.

[6] B. Noia, K. Chakrabarty, and Y. Xie, “Test-Wrapper Optimization for Embedded Cores in TSV-Based Three-Dimensional SOCs”, *Proc. IEEE International Conference on Computer Design*, pp. 70-77, 2009.

[7] X. Wu, Y. Chen K. Chakrabarty, and Y. Xie, “Test-access Mechanism Optimization for Core-based Three-dimensional SOCs,” *ICCD* 2008.

[8] L. Jiang, L. Huang, and Q. Xu, “Test Architecture Design and Optimization for Three-dimensional SoCs,”' *DATE*, pp. 220-225, 2009.

[9] L. Jiang, Q. Xu, K. Chakrabarty, and T.M. Mak, “Layout-driven Test-architecture Design and Optimization for 3D SoCs under Pre-bond Test-pin-count Constraint”, *International Conference on Computer-Aided Design*, pp. 191-196, 2009.

[10] E.J. Marinissen, V. Iyengar, and K. Chakrabarty. “A Set of Benchmarks for Modular Testing of SOCs”, In *International Test Conference*, pp. 519-528, Oct. 2002.

[11] M.R. Garey and D.S. Johnson, “Computers and Intractability A Guide to the Theory of NP-Completeness”, New York: Freeman, 1979.

[12] E.G. Coffman, Jr., M.R. Garey, D.S. Johnson and R.E. Tarjan. “Performance bounds for level-oriented two-dimensional packing algorithms”, *SIAM J. Computing*, vol. 9, pp. 809826, 1980.

[13] X. Dong and Y. Xie. “System-level Cost Analysis and Design Exploration for 3D ICs”, *Proceedings of Asia-South Pacific Design Automation Conference (ASP-DAC)*, pp.234-241, Jan. 2009.

[14] S.K. Goel and E.J. Marinissen, “Control-Aware Test Architecture Design for Modular SOC Testing”, *European Test Workshop*, pp. 57-62, 2003.

[15] S. K. Goel and E. J. Marinissen. “SOC Test Architecture Design for Efficient Utilization of Test Bandwidth,”' *ACM Transactions on Design Automation of Electronic Systems*, 8(4):399429, 2003.