ISSN: 2321-8169
546 - 553

Robust Optimization of Test-Access Architectures for 3-D Stacked ICs

Sonika Nayan
Department of Information Technology
Indian Institute Of Engineering Science and Technology (IIEST), shibpur,Howrah
sonikanayan@gmail.com

Sonali Nayan
Department of Information Technology
Indian Institute Of Engineering Science and Technology (IIEST), shibpur,Howrah
sonalinayan25@gmail.com

Dinesh Hemant Bhere
Department of Computer Engineering
Vishwatmak Omgurudev College Of Engineering, Mumbai
vogcetpo2015@gmail.com

***Abstract* :-** Technology climb up for higher production and lower power use is being hampered today by the congestion of interconnect lengths. 3D integrated circuits (3D ICs) based on through-silicon vias (TSVs) have emerged as a important solution for overcoming the interconnect congestion. However, testing of 3D ICs remains a importantt challenge, and development in test technology are needed to make 3D integration commercially viable. This paper presents a survey of test challenges for 3D ICs and describes recent innovations on various aspects of 3D testing, DFT (Design for test-ability) and different optimization technique and their limitations. Design of core based three dimensional (3D) system-on-chip (SOC) is gaining a marked attention in modern day's semiconductor industry. Testing of 3D SOC is considered as one of the important challenge and hence efficient test techniques are required. In our work, we have proposed a heuristic approach to group number of cores of different SoCs based on n, 2n, 4n configurable multiple-width TAM in such a way that the total test time is reduced under maximum TAM width. So, the robust optimization may work more correctly with optimal solutions.

*****

I. INTRODUCTION

A 3D IC is a chip that comprises of numerous levels of diminished dynamic 2D incorporated circuits (2D ICs) that are stacked, reinforced, and electrically associated with vertical visa framed through silicon or oxide layers and whose game plan inside the levels is one of a kind. The term level is utilized to recognize the exchanged layers of a 3D IC from outline and physical layers and is the utilitarian area of a chip or wafer that comprises of the dynamic silicon, the interconnect, and, for a silicon-on-oxide (SOI) wafer, the covered oxide (BOX).The TSV (through silicon by means of) is a required element of the 3D IC technology[1] and is the vertical-electrical association shaped amongst levels and through silicon or oxide. A TSV is shaped by adjusting, and carving a hole between two levels to. The centers of 3D SOC are kept at numerous layers, consequently the 3D test access design confronts a more perplexing con figuration environment than 2D SOC. Test assets, similar to number of test pins, TSV tally, directing territory for intra and entomb layer test center booking and so forth are exceptionally constrained if there should arise an occurrence of 3D structure. Testing of the centers of a SOC requires test access instrument (TAM) for transporting test examples and test reactions between SOCs test sticks and center I/Os. Again the test pins are set just at the base layer, thusly test access to centers present at the upper layers require TSVs. An exceptionally constrained works have been done on the aforementioned issue of vertically stacked structure till date[2]. This persuades our work for outlining a productive test booking plan of centers for streamlining of general test time of the SOC considering diverse requirements.

Vigorous enhancement is a generally new way to deal with advancement issues an affected by instability, yet it has effectively demonstrated so valuable in genuine applications that it is hard to handle such issues today without considering this capable strategy. When all is said in done, enhancement strategies being used today accept that exceptionally essential info parameters, for example, center test time and power utilization amid test are known at the outline stage. Presently the issue is, these parameters can change subsequent to assembling and in that situation the initially outlined test design may never again be ideal. In this way instability parameters become an integral factor, test time of the SoCs (System-on-Chip). Hence, hearty arrangements are better than single point arrangements as far as normal test time when there are instabilities in the estimations of information parameters [6]. In this printed material a heuristic methodology has been proposed to gathering N number of centers of various SoCs in light of n, 2n, 4n configurable different width TAM in a manner that the aggregate test time is decreased under most extreme TAM width. In this way, the strong enhancement may work all the more effectively with ideal arrangements.

Traditional climb of semiconductor chips also improves signal propagation speed. However, scaling from current manufacturing and chip-design technologies [4] has become more difficult, in part because of power-density constraints, and in part because interconnects do not become faster while transistors do. 3-D integrated circuits were invented to address the scaling challenge by stacking 2-D dies and connecting them

ISSN: 2321-8169
546 - 553

in the 3rd dimension. This promises to speed up com-munication between layered chips, compared to planar layout. 3D ICs promise many significant benefits.

- Heterogeneous integration

3D integration supports heterogeneous stacking because different types of components can be fabricated separately, and layers can be implemented with different technologies.

- Shorter interconnect

The average wire length is reduced. The reduction of wire length due to 3D integration can result in two clear benefits: latency improvement and power reduction.

- Power Reduction

The reduction of the wire length translates into the power saving in 3D IC design.. Shorter wires also reduce power consumption by producing less parasitic capacitance. Reducing the power budget leads to less heat generation, extended battery life, and lower cost of operation.

- Latency Improvement

Latency improvement can be achieved due to the reduction of average interconnect length and the critical path length Bandwidth

3D integration allows large numbers of vertical vias between the layers. This allows construction of wide bandwidth buses between functional blocks in different layers.

- Circuit security and footprint

Sensitive circuits may also be divided among the layers in such a way as to obscure the function of each layer. More functionality into a small space.

- Cost-effective

. The heterogeneous integration capability that 3D provides can also help reduce the cost.3D stacking can potentially provide a cost-effective integration solution, compared to traditional technology scaling.

Test access optimization

Test access is a major problem for stacked 3D ICs. Test access is given to the scan and I/O terminals of logic cores, memory BIST (Built-in self-test) controllers, TAP controllers, and other test structures. Recent research on this issue has addressed the dual problems of designing scan chains for cores placed on multiple layers [7] and the optimization of the TAM for a core-based 3D IC [8]. The TAM design needs to be optimized to minimize the test application time.

II. LITERATURE SURVEY

Today's technology allows for the integration of many cores in a single die, for instance, in core-based SoCs, and an even larger number of cores are likely to be integrated over multiple layers in a 3D stack[5]. In order to minimize test cost, the test architecture in a core-based SOC is optimized for minimum test time Optimization methods in use today assume that all relevant input parameters, such as core test time and power consumption during test, are known at the design stage. However, these parameters can change after manufacturing and, in that scenario; the originally designed test architecture may no longer be optimal. Moreover, conventional op-timization methods have to consider worst-case estimates for all input parameters to ensure feasibility, which can result in less and hence costly solutions. They propose the use of robust optimization [9] for test-architecture design and test scheduling. Traditional (non-robust) optimization of test architectures for SoCs as well as for 3D ICs has been studied in the past [10-12]. The proposed methods [7] assume that all input parameters are known and will remain constant after the design has been fixed. This may not be true in practice, resulting in non-optimal solutions. The robust optimization method proposed in [13] takes uncertainties of design parameters into account. Here, they formulate, for the first time, a robust optimization problem for SOC testing. The robust optimization method is scalable if the problem can be modelled as LP (linear programming), since LP problems are solvable in polynomial time. To solve these problems optimally, they use ILP (Integer linear programming) for reasonably-sized problem instances. They formulate an ILP model for robust optimization of test architecture design and test scheduling. This model is practical for systems, such as SoCs or 3D ICs, which contain a relatively small number of cores. They show that robust optimization provides solutions that are on average better than non-robust solutions in terms of test time. For larger systems, for which the exact solution of the ILP model becomes not solvable, they propose a randomized divide-and-conquer heuristic for robust optimization that uses the exact ILP method to solve sub-problems. The proposed heuristic scales linearly with the number of cores and hence practical for large systems. In addition, since the iterations during the randomized divide-and-conquer procedure are freet from each other, the performance of their heuristic can be increased by running multiple instances (or threads) in parallel.

Three-dimensional stacking using through-silicon vias (TSVs) has enabled a new way to integrate even more devices in a single package and sustain Moore's law. TSVs are metal vias that provide electrical connections from one die to another through the silicon substrate [15]. TSVs outperform wire bonds used in conventional 3D stacking in interconnect density, performance and power consumption. Recent work on the optimization of 3D test architectures [14] to reduce test cost su ffer from the drawback that they ignore potential uncertainties in input parameters. In a 3D setting, a die can be used in multiple stacks each with variable properties. As a result, the originally designed test architecture might no longer be optimal, which leads to an unnecessary increase in the test cost. Optimization of 2D SOC test architectures has been explored well in the previous [16-19]. The proposed methods use ILP as

well as heuristics such as rectangle packing in order to optimize the test architecture for minimum test time.

Here they propose [8] an advancement approach that considers instabilities in information parameters and gives an answer that is productive within the sight of info parameter varieties. In their investigations, the business ILP solver was not ready to give an ideal arrangement even to a 8-center bite the dust inside 24 hours. In this manner, they propose a productive heuristic to handle kicks the bucket with more than 20 centers. Consequently hearty enhancement works taking into account heuristic strategy, reproduced strengthening.

Because of procedure varieties and the genuine force farthest point can be di afferent from the quality taken at the outline stage. For this situation, tests may should be rescheduled to satisfy the force oblige and improve the test time. Dubious parameters are the test design, test power and test time, appropriateness of test. Power utilization gives overheating and over testing, when information parameters change, prompting expanded expense. Since substantial group of work on SoCs test planning, test design issues in genuine situations for center based 3D ICs require more investigation.

As 3D stacking offers incorporation of numerous bites the dust in a solitary bundle, the issue of high power utilization (and coming about warmth dissemination) turns out to be much more basic than in 2D SoCs. Late few works have distinguished this issue and a few strategies for improved cooling in 3D-SICs have been proposed [20]. By and by, force utilization amid 3D-SIC test must be constrained to avoid over-warming and over-testing. This cut-off may not be known before the 3D-SIC has been produced; henceforth we should depend on assessments of center force utilization acquired through reproduction.

The essential issue in test planning is to dole out a begin time for all tests while taking all imperatives [11]. All together lessen the test time it is vital to plan the tests all the while as could be allowed. In any case, impediments must be precisely considered. For example, the tests might be in with each other because of the sharing of test assets; and force utilization must be controlled, generally the framework might be harmed amid test. A test is an arrangement of test vectors created or put away at a test source (set on-chip) and the test reaction from the test is assessed at a test sink (set on-chip). On the off chance that a testable unit is tried by a few test sets (frequently an outer test set and an on-chip test set are required to achieve high test quality), a test struggle happen and one and only test set can be connected at once.

As of late, Muresan et al. [17] proposed a methodology where test time is minimized while test clashes and test power utilization are considered and Chakraborty proposed a methodology minimizing test time for center based frameworks.

As of late, some early work has been accounted for on testing of 3D SICs. Heuristic techniques for outlining center wrappers in 3D-SICs were created in [18]. ILP models for test engineering outline for every bite the dust in a stack is introduced in. While these ILP model stake into record a portion of the limitations identified with 3D - SIC testing, for example, a TSV limit, this methodology does not consider the reuse of bite the dust level TAMs. A TAM wire-length minimization method taking into account mimicked toughening is introduced in[3]. Heuristic strategies for lessening weighted test cost while considering the requirements on test pin widths in pre-security and post-security tests are portrayed in.In most earlier work on 3D-SIC testing, TAM advancement is performed at pass on level just, which prompts wasteful TAMs and non-ideal test plans for halfway finish stack test[20]. Besides, all past strategies expect that the originator can make TAM structures on every bite the dust amid improvement, which may not be conceivable in all cases. This paper considers test design improvement for the whole stack and considers 3D stacks with both hard and delicate kicks the bucket. We likewise investigate the impact of accessible test pins and TSVs utilized on TAM plan and test booking.

III. PROPOSED WORK

In order to optimize test time, we have proposed a heuristic approach or algorithm where grouping of N number of cores of different SOC based on n, 2n, 4n configurable multiple-width TAM is done. Generally, with the increasing number of n (TAM width) test time decreases and that is the target of our algorithm. Now the proposed algorithm namely Algorithm 1 and optimization function namely Algorithm 2 which is used in Algorithm 1, are discussed below

Proposed Algorithm

A session is formed by a set of cores that can be tested simultaneously and the test time of this session is the maximum among all these cores. The algorithm tries to find the minimum test time under n, 2n, 4n configurable maximum available TAM width within a session. To create a session first we have to choose the core of the maximum test time with the TAM width which is half of the maximum available tam width (Algorithm 1 line no.8). The test time will be the test time of the corresponding session if only all the remaining cores are allocated within the remaining TAM width. After choosing the first core with the maximum test time among the cores with the same TAM width, the core having remaining TAM width and less or equal test time than the first core will be chosen and if less or equal time is not found then the maximum test time among two cores will be saved (Algorithm 2). Now the two cores will run parallel.

In same the way in the next session a core will be chosen from the remaining cores with the maximum test time and with the same TAM width that the first core has. The same process will be continued until all the cores are allocated and the test times of the different sessions are added as the total test time of that iteration.

Now for the next iteration the first TAM width will be decreased by one (Algorithm 1 line no.11) and the above mentioned same process will be continued for the whole iteration. A comparison between the two test times from the two iterations will be measured and the lesser test time value with the iteration will be saved for further use. Then in the same way the test time of the iteration having TAM width increased by one (Algorithm 1 line no.14) will be calculated and compared with the test time of other iteration. Thus with all possible measures of TAM width, the test time will be calculated and compared with each other for different iterations. The least test time will finally be saved (Algorithm 1 line no.16).In the end the optimized test time is accepted. Now the whole of the above mentioned process will be done for 2n and 4n configurable multiple-width TAM also. The optimized test time of the different configurations will be added and said to be the total test time (Algorithm 1 line no.18).

Algorithm 1: Test Time Optimization

Input: Num of TAM width (W_a), num of core (N).

Output: Total test time (T_{tot})

1. Begin
2. S→ Set of cores;
3. W_{max}= Maximum available TAM width;
4. T_{tot}= 0;
5. C=S; S_n= 0; /* Initial session */
6. For x=0 to 2 do
7. $W_{max}=2^x * n$; /* For n,2n,4n configurable multiple-width TAM */
8. $W_a=W_{max}/2$; /* Take mid value of input TAM width */
9. $W_x=W_a$;
10. For i=1 to Wa do
11. T_i=func(W_a-i); /* Function called to get minimal test time Ti when TAM width value subtracted from the mid value of TAM width */
12. End For
13. For j=1 to Wa do
14. T_j=func(W_a+j); /* Function called to get minimal test time Tj when TAM width value added with the mid value of TAM width */
15. End For
16. If (T_i <T_j) do

a.Take T_i as the final test time(T) of cores

else do

c.Take T_j as the final test time(T) of cores

17. EndIf
18. $T_{total}=T_{total}$+T; /* Where T=T_i or T_j */
19. $S_n=S_n$+1; /*Next session */
20. End For
21. End

Algorithm 2: Optimization Function

Function func(W_x)

a. Begin

b.Tn =0;

c. do

d. If C =i AND W_i-1>W_i do

e. choose core i C : T_i of W_x is max. /* Core with maximum time value is selected*/

f. $T_n=T_n+T_i$; /* Time value is added to compute total time */

g. $W_x=W_{max}-W_x$; /* Tam width is subtracted */else do

i. if T_i > T_i-1

j. Take T_i of W_x is max /* Time is considered if Wx is maximum*/

k.else do

l. Take T_i-1 is max /* The previous time value is considered */

m. C= C-i; /* Core number is subtracted from the set of cores */

n. EndIf

o. while(C= ∅) /* Above statements are executed until set of core C is not null*

p. End

Generalized equation to calculate overall test time of SoC:

$$S = S + Max(t_j)$$

where j=1 to p, for $2^x * n$, where x=0 to m.

s=Total test time

t_j=total test time of each core

An example

Assume an SOC having six cores, namely C1, C2, C3, C4, C5 and C6. Maximum available TAM width is 16 and the test time (in clock cycles) of these cores with respect to di erent TAM widths.

First assume the value of n is 2 (for n, 2n, 4n configurable multiple-TAM width). Now the mid value is 1.According to Algorithm1, among all possible combinations of TAM width (1, 1) gives the minimum test time. Here, core C4 is selected as it has the maximum test time among all the cores with respect to TAM width 1. Now, the algorithm will look for another core to be tested in parallel with the remaining available TAM width 1. C5 is selected as it has next maximum test time with respect to TAM width 1 again(a core can be tested in parallel with first core of that session if it has less or equal test time with the first core and if a less or equal value is not found then the maximum value between two cores will be selected). Now among the unscheduled

 ISSN: 2321-8169
546 - 553

cores, C3 and C2 are selected for the next session and get a test time. Next, C1 and C6 are selected for next the schedule and get a test time. After the scheduling of all the six cores, the final arrangements of cores are shown in Figure:1 and the overall test time is shown in Figure 2, that is t2=(9000+6000+4470)=19470 clock cycles. Combinations of TAM width (2, 2) gives the minimum test time. In the same way (C4,C5), (C2,C3), (C1,C6) are selected to be tested in parallel and get test time in different sessions. After the scheduling of all the six cores, the final arrangements of cores are shown in Figure3 and the overall test time is shown in Figure 4, that is t4=(8100+4700+3567)=16367 clock cycles. Now, the TAM width is 8. The mid value is 4. According to Algorithm 1, among all possible combinations of TAM width (6,2) gives the minimum test time. In the same way (c4, c5),

Table 1:Different cores with TAM Width

TAM	C1	C2	C3	C4	C5	C6
1	3345	4884	6000	9000	6785	4470
2	1920	4000	4700	8100	6000	3567
3	1400	3976	4700	7900	5000	2300
4	1000	2500	4700	7738	4219	2000
5	987	2300	4000	6280	4100	1800
6	800	2300	3333	6000	4100	1700
7	800	2000	3213	5400	3793	1700
8	800	2000	2387	5400	3500	1700
9	749	2000	2000	5200	3500	1700
10	600	2000	2000	5000	3500	1700
11	600	2000	2000	5000	3500	1700
12	600	2000	1874	5000	3500	1700
13	590	1990	1872	4950	3488	1660
14	580	1990	1870	4950	3486	1650
15	580	1970	1865	4930	3470	1648
16	550	1940	1830	4925	3465	1630

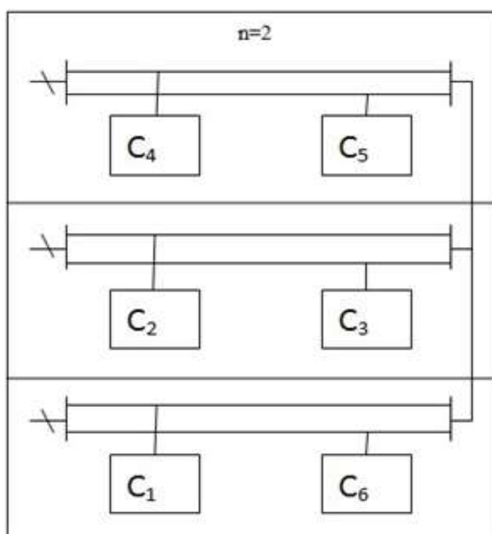

Figure1: TAM width assignment of cores

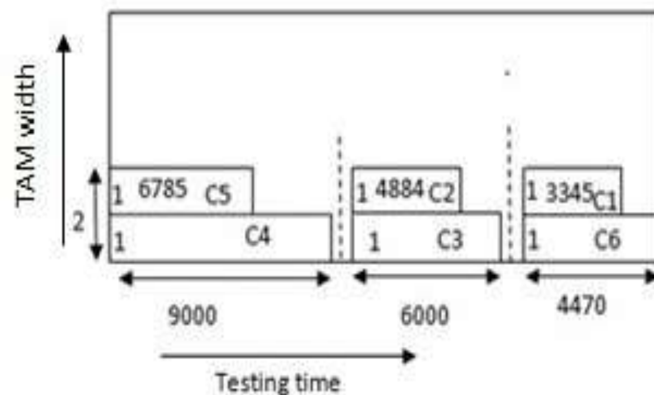

Figure 2: Core test scheduling with TAM width 2

(C1,C3), (C2,C6) are selected to be tested in parallel and get test time in different sessions. After the scheduling of all the six cores, the final arrangements of cores are shown in Figure5 And the overall test time is shown in Figure 6, that is t8=(6000+3333+4470)=13803 clock cycles. Here, T_{total} =(t2+t4+t8)=49640 clock cycles.Again, assume the value of n is 4 (for n, 2n, 4n configurable multiple-TAM width). Similarly we will get the test time of t4, t8, t16. And T_{total}(16367+13803+11781)= 41951 clock cycles.

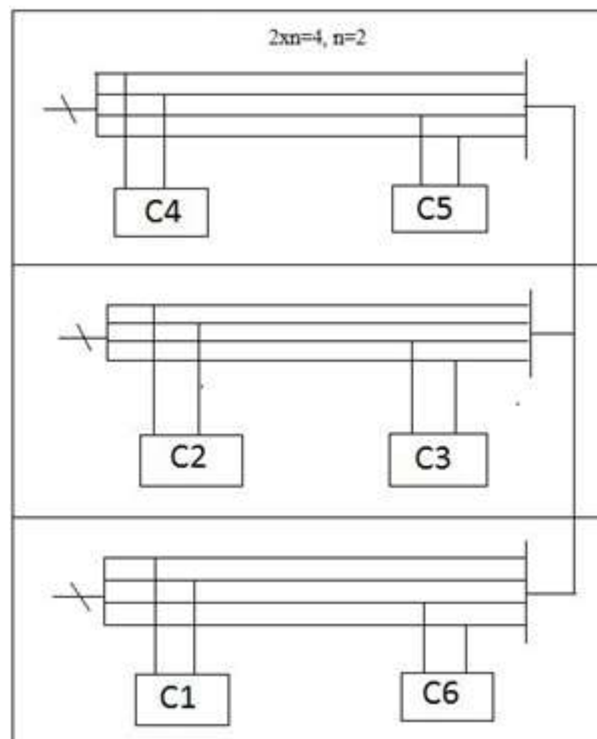

Figure 3 : TAM width assignment of cores

ISSN: 2321-8169
546 - 553

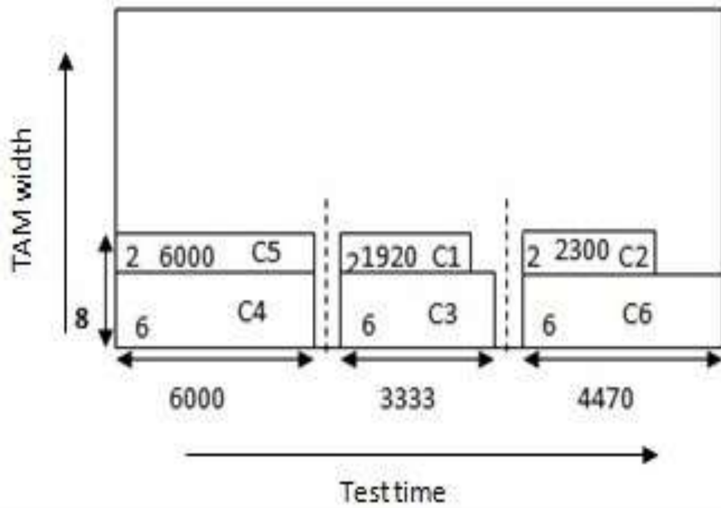

Figure 4: Core test scheduling with TAM width

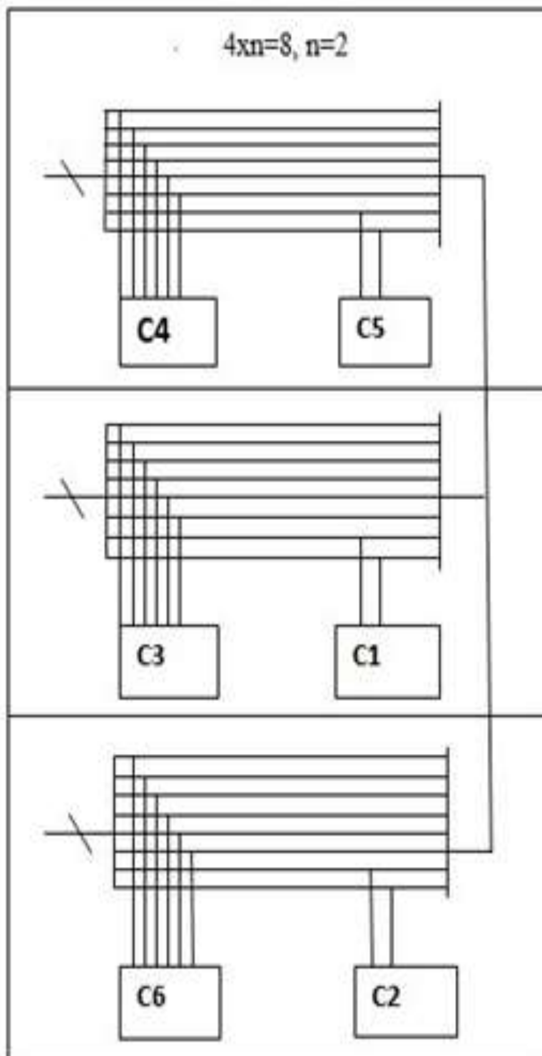

Figure 5: TAM width assignment of cores

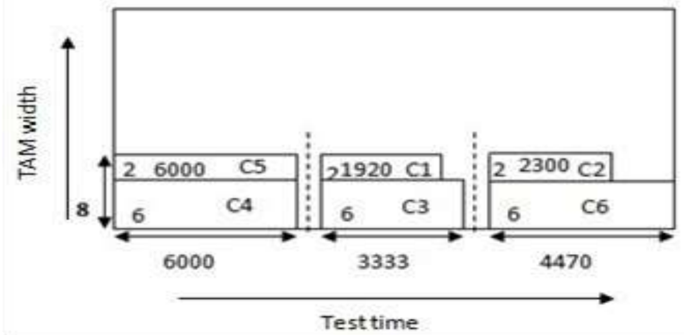

Figure 6: Core test scheduling with TAM width

Discussion of Experimental Results

For simulation the proposed algorithm is implemented in C++ language, G++ compiler and run on an Intel Core i3 processor having 4GB RAM on a Linux operating system. In our experiment we have used five SoCs of ITC'02 SOC Benchmarks[43]. The benchmark SoCs are 1. d695 2. p22810 3. p34392 4. p93791 5. f2126. Now for n, 2n, 4n configurable multiple-TAM width, we have used 4,6,8...upto 16 n values and got their respective total test time. The results are shown in tabular form as follows

Table2: Test time table for the benchmark SOC d695

d695				
TAM Width	Testing time (T_n) (clockcycles)	Testing time (T_{2n}) (clock cycles)	Testing time (T_{4n}) (clockcycles)	Total testing time T_{total} (clock cycles)
n=4	183318	66290	42130	291738
n=6	93207	46842	27050	167099
n=8	66290	42130	25910	134330
n=10	52204	30916	23252	106372
n=12	46842	27050	23179	97071
n=14	42365	26095	23105	91565
n=16	42130	25910	22511	90551

Table 3: Test time table for the benchmark SOC p22180

P22180				
TAM Width	Testing time (T_n) (clockcycles)	Testing time (T_{2n}) (clock cycles)	Testing time (T_{4n}) (clockcycles)	Total testing time T_{total} (clock cycles)
n=4	2087298	731380	432868	3251546
n=6	1056253	501354	376921	1934528
n=8	731380	432868	334045	1498293
n=10	574752	381762	330091	1286605
n=12	501354	376921	324048	1202323
n=14	459321	352473	318262	1130056
n=16	432868	334045	302967	1069880

Table 4: Test time table for the benchmark SOC p343924

P343924				
TAM Width	Testing time (T_n) (clockcycles)	Testing time (T_{2n}) (clock cycles)	Testing time (T_{4n}) (clockcycles)	Total testing time T_{total} (clockcycles)
n=4	4285725	1564308	982242	6832275
n=6	2279176	1100232	893818	4273226
n=8	1564308	982242	881481	3428031
n=10	1261632	913062	865369	3040063
n=12	1100232	893818	862705	2856755
n=14	1078658	881510	862704	2822872
n=16	982242	881481	862526	2726249

Table 5: Test time table for the benchmark SOC p937915

P937915				
TAM Width	Testing time (T_n) (clock cycles)	Testing time (T_{2n}) (clock cycles)	Testing time (T_{4n}) (clock cycles)	Total testing time T_{total} (clock cycles)
n=4	7742924	2632639	1296126	11671689
n=6	3919818	1648513	918123	6486454
n=8	2632639	1296126	675108	4603873
n=10	1984447	1000101	659343	3643891
n=12	1648513	918123	376614	2943250
n=14	1337970	688515	355822	2382307
n=16	1296126	675108	355575	2326809

Table 6: Test time table for the benchmark SOC f2126

d34392				
TAM Width	Testing time (T_n) (clockcycles)	Testing time (T_{2n}) (clock cycles)	Testing time (T_{4n}) (clockcycles)	Total testing time T_{total} (clock cycles)
n=4	1474458	716440	382445	2573343
n=6	760952	382445	382445	1525842
n=8	716440	382445	382445	1481330
n=10	404199	382445	382445	1169089
n=12	382445	382445	382445	1147335
n=14	382445	382445	382445	1147335
n=16	382445	382445	382445	1147335

From the above experimental results, we observe that the test time decreases with the increasing TAM width values. It is also observed that Tables: 8.1, 8.2, 8.3, 8.4 give the total test time in descending order but the Table: 8.5 does not follow after TAM width (n) value 10. The test time comes with the same values for n=12, 14 and 16. The reason behind this scenario is that the number of cores is 4, and core 3 and core 4 have the same clock cycle time after TAM width 2. Notably, many cores may have the same clock cycle time with respect to the same TAM width

value. Therefore, the test time of the cores remains the same even if the TAM width value increases.

IV. CONCLUSION

In this thesis work, in order to minimize the test time, grouping of N number of cores of different SoC based on n, 2n, 4n configurable multiple-width TAM is done. An algorithm or a heuristic approach is applied to get the optimized results. The proposed algorithm is discussed with an example. Finally, simulation results are applied on5 Socs of ITC'02 SOC Benchmarks. The result shows the optimized test time in tabular form.

Future Scope of Work

In our future work, the focus will be on how to consider uncertainty in parameter like temperature in Robust Optimization. In our work, we have chosen two cores to be tested in parallel and this may not produce that much of heat to cause damage but we have to take care of it in future. Later, we may have to think how to take more than two cores to optimize the test time more efficiently.

REFERENCES

[1] James Burns, Three Dimensional System Integration, Springer US: 246, 2011
[2] Knickerebocker JU, Andry PS, Dang B, Horton RR, Interrante MJ, Patel CS et al (2006) Three-dimensional silicon integration. IBM J Res Dev 50(4/5):553–567
[3] Yuan Xie, Yibo Chen, Guangyu Sun, Jin Ouyang, Three-dimensional Integrated Circuits Design, EDA, and Architecture, Computer Science and Engineering DepartmentUniversity Park, PA16802, USA
[4] Hsien-Hsin S. Lee, Krishnendu Chakrabarty, "Test Challenges for 3D Integrated Circuits", IEEE Design & Test of Computers, vol.26, no. 5, pp. 26-35, September-october doi:10.1109/MDT.2009.125 September-October2009.
[5] J.W. Joyner, P. Zarkesh-Ha, and J.D. Meindl. A stochastic global net-length distribution for a three-dimensional system-on-a-chip (3D-SoC). In Proc. 14th Annual IEEE International ASIC/SOC Conference, September 2001.
[6] Jin Ouyang, Guangyu Sun, Yibo Chen, Lian Duan, Tao Zhang, Yuan Xie, and Mary Irwin.Arithmetic unit design using 180nm TSV-based 3D stacking technology. In IEEE Interna- tional 3D System Integration Conference, 2009.
[7] B. Black et al. die stacking 3D microarchitecture. In MICRO, pages 469-479, 2006.
[8] Y. Xie, G Loh, B Black, and K. Bernstein. Design space exploration for 3architectures.ACM Journal of Emerging Technologies in Compuing Systems, 2006.
[9] T.M. Mak, "Testing of 3D Circuits," Handbook of 3D Integration: Technology and Appli- cations Using 3D Integrated Circuits, P. Garrou, C. Bower, and P. Ramm,eds., Wiley-CVH,2008.
[10] X. Wu et al., "Scan-Chain Design and Optimization for Three-Dimensional Integrated Circuits," ACM J. Emerging Technologies in Computing Systems, vol. 5, no. 2, 2008, article no.9.
[11] X. Wu et al., "Test-Access Mechanism Optimization for CoreBasedThreeDimensionalSOCs," Proc. IEEE Int'l Conf. Computer Design, IEEE CS Press, 2008, pp. 212-218.
[12] Deutsch, S.; Chakrabarty, K., "Robust optimization of test-architecture designs for core- based SoCs," Test Symposium (ETS), 2013 18th IEEE European , vol., no., pp.1,6, 27-30 May 2013 doi: 10.1109/ETS.2013
[13] Sergej Deutsch, Krishnendu Chakrabarty, Erik Jan Marinissen, "Uncertainty-aware robust optimization of test-access architectures for 3D stacked ICs", ITC, 2013, 2013 IEEE Inter- national Test Conference (ITC), 2013 IEEE International Test Conference (ITC) 2013, pp.1-10, doi:10.1109/TEST.2013.6651905
[14] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios. Robust Optimization of Large-Scale Systems. Operations Research, 43(2):264–281, 1995.
[15] E. J. Marinissen, S. K. Goel, and M. Lousberg. Wrapper Design for Embedded Core Test.In ITC, pages 911–920, 2000.
[16] B. Noia et al. Test-Architecture Optimization and Test Scheduling for TSV Based 3-D Stacked ICs. TCAD, 30(11):1705 –1718, November 2011.
[17] K. Chakrabarty. Test Scheduling for Core Based Systems Using Mixed- Integer Linear Pro-gramming. TCAD, pages 1163–1174, 2000.
[18] Y. Huang et al. Resource Allocation and Test Scheduling for Concurrent Test of Core-Based SOC Design. In ATS, pages 265–270, 2001.
[19] K. Banerjee et al. 3-D ICs: A Novel Chip Design for Improving Deep- Submicrome- ter Interconnect Performance and Systems-on-Chip Integration. Proceedings of the IEEE, 89(5):602–633, May 2001.
[20] E.J. Marinissen, S.K. Goel, and M. Lousberg. Wrapper Design for Embedded Core Test. In ITC, pages 911–920, 2000.