2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems

A Test Partitioning Technique for Scheduling Tests for Thermally Constrained 3D Integrated Circuits

Spencer K. Millican and Kewal K. Saluja
Department of Electrical and Computer Engineering, University of Wisconsin-Madison, WI
1415 Engineering Dr., Madison, WI 53706
smillican@wisc.edu, saluja@ece.wisc.edu

***Abstract*—Increasing design complexity coupled with new design and manufacturing techniques being used for modern integrate circuits is creating challenges for test environment. The goal of system-on chip (SoC) test scheduling has always been to reduce test application time. Added design constraints for SoC environment are making this scheduling more difficult. This difficulty is increased by manufacturing techniques like 3D stacked integrated circuits. Traditional test schedules for 3D stacked ICs can be either prohibitively long or may not exist without resorting to test partitioning. Partitioning methods proposed in literature have been ad hoc or simplistic. This paper presents a test partitioning method specifically designed for thermally constrained tests for the purpose of reducing test application time of 3D stacked integrated circuits under temperature constraint. The efficiency of the method is demonstrated by comparing it to the ad hoc methods previously investigated in the literature.**

I. Introduction

Reduction in test application time (TAT) has always been an essential goal in integrated circuit (IC) test. Since test is a costly part of the IC manufacturing process, many methods have been proposed to improve test economy. One such method is to reduce the time required to apply tests to a device. Even a small reduction in TAT per device greatly reduces test cost [1]. There have been many approaches to reducing test time, such as using more efficient tests or more efficient test hardware, but the goal of all methods is to decrease test time while keeping high fault coverage.

A model for testing ICs that has been found to be practical is the system-on-chip (SoC) test model. The SoC test model comes from the design practice of creating a large design out of several smaller independently designed cores or modules. Designing SoCs is particularly efficient for fast time-to-market development at low cost by utilizing proprietary Intellectual Property (IP) cores instead of designing new cores for a particular project. To test SoCs, IP core providers make thier test sets available to the designers to apply, with the tests sets having a guaranteed level of fault coverage. Therefore, the goal of a designer using SoC cores is to apply pre-existing tests for each core in the smallest possible period of time without violating given design constraints.

The design technique of 3D stacked ICs (3D-ICs) provides a method to create larger complex circuits while still holding to the traditional SoC test model. 3D-ICs is a process of stacking several IC dies for the purpose of implementing more complex designs with a small footprint. Such designs have not only the benefit of smaller design space but also offer higher performance by having shorter interconnects. From the point of view of SoC test, the stacking of dies adds to the number of cores in the design [2]. Although it may be true that unique design constraints are formed due to the stacking of dies, these new constraints fit the SoC test model since the SoC test model does not depend on the location of a core on a die but only the relationship between individual cores.

As feature sizes of ICs have become smaller, new constraints have been added to SoC test scheduling to address new design concerns, such as resources, power, and temperature. The first constraints given in the SoC test model were hardware constraints [3]. These constraints addressed shared hardware between cores that forbid cores to be tested simultaneously. The goal was to apply all tests for every core without violating a given hardware constraint. As the transistor density of ICs rose due to smaller feature sizes, the power dissipation of ICs during test has became high enough to either damage the device during test or create false failures causing lower device yield [4]. The goal of SoC test then became to apply all tests of every core without the total power dissipation of a device under test (DUT) violating a given power bound while still not violating any hardware constraints. With higher power dissipation comes larger heat dissipation. If the temperature of a DUT becomes too high, damage can be caused to the DUT thereby lowering device yield, therefore temperature constrains have been added along with hardware and power constraints for SoC test.

The introduction of temperature constraint has an unique impact on 3D-ICs due to their stacked nature. The effect of temperature during test of non-stacked dies has been observed [2], [5], [6], and studies have shown that the violation of temperature bound can indeed damage the device. This has also been found in 3D-ICs [2], [7], [8]. However, the stacking of dies in 3D-ICs creates a unique challenge to temperature-constrained test scheduling since stacking dies insulates heat from being dissipated. Even when thermal through-silicon vias (TTSVs) are used to assist in dissipating heat, the stacking of dies will increase the temperature of lower and sandwiched dies in a stack by creating a longer thermal distance between a die and its heat sink.

One possible remedy for reducing TAT of 3D-ICs under temperature constraints is to introduce test partitioning. The partitioning of tests is the practice of dividing a given test into smaller tests (which can often be done since in most cases test vectors can be applied in any order) in order to reduce TAT

1063-9667/14 $31.00 © 2014 IEEE
DOI 10.1109/VLSID.2014.11

[9]. Partitioning to reduce TAT makes use of overlaps that can be created by diving large tests into smaller ones which may otherwise not exist. Another benefit of partitioning, relative to other test compaction methods, is that no extra hardware is needed to implement it and it requires no or minimal changes in the design flow since partitioning is done on already-existing tests. The effect partitioning has on reducing TAT was observed early in hardware-constrained scheduling [9]. It has also been observed in power and temperature-constrained scheduling [10], [11]. Partitioning methods investigated in the past have either been simplistic (equal sized-partitions, partition only when required by constraints) or ad hoc. However, to the best of our knowledge the effectiveness of different partitioning methods or what makes a good partition has never be explored.

The contributions of this paper are as follows:

- To show the significant impact temperature has on 3D-ICs as opposed to non-stacked dies.
- To show that partitioning not only has a significant impact on the quality of schedules but in some cases it is required to obtain temperature constraint schedules.
- To develop a partitioning scheme and to show that it has a significant impact on the quality of SoC test schedules compared to ad hoc partitioning methods.

The rest of the paper is organized as follows: Section II gives a brief history of SoC and 3D-IC test scheduling. Section III provides a partitioning method with the goal of developing superior test schedules compared to other partitioning methods. Section IV gives a design of an experiment to evaluate the effect stacking of dies has on temperature-constrained test scheduling, the effect partitioning has on schedule quality, and the effectiveness of the proposed partitioning method. Section V gives the results of the experiment and a discussion of the results, and the paper concludes with Section VI.

II. Past Work

Initial work on SoC test scheduling focused on how to schedule all tests of an SoC in the shortest possible period of time without violating hardware constraints [3]. Different cores/modules in an SoC often share one or more hardware resources which makes the simultaneous testing of these modules impossible. For instance, two cores may share the same memory or the same I/O ports. The sharing of resources may also be test-specific, such as the sharing of test access mechanisms or built-in self-test (BIST) vector generators. For the purpose of test, a test schedule had to either presume that hardware has already been set [3] or that hardware can be generated by the scheduler to obtain shorter test schedules [12]. In the former case, hardware compatibility is predefined and tests must be scheduled around the given hardware constraints, but in the later new constraints are added (e.g. the total number of test-dedicated pins allowed) and the scheduler defines its own hardware compatibility. In this study the former is assumed since hardware compatibility is not the focus of this study.

To deal with new constraints caused by increased IC complexity, such as power constraints, hardware-constrained test scheduling formulations were expanded to meet the new constraints. One such method that has been used is to declare two modules to be hardware incompatible if their combined power consumption is greater than the given bound [4]. This method is straightforward and functionally correct, but this definition of incompatibility does not scale if more than two tests are scheduled at a given time, therefore every "clique" of compatible tests must be found, which is computationally intensive. This method also lacks the ability to schedule tests outside of sessions which may increase TAT. Instead, power is better defined as a separate constraint where the sum of power dissipation of all modules at any given time during test must be less than the given bound [13].

A power-constrained test scheduling formulation can be extended to temperature-constrained test scheduling using the superposition principle. Formerly, the major difficulty in meeting temperature constraints was the time-consuming simulation of temperatures, which was thought to be required for every possible test schedule. The consequence of this was a temperature profile of any proposed test schedule could only be known after a lengthy temperature-simulation [14]. However, it has been observed that the temperature profile of two tests running simultaneously is the sum of the temperature profiles of the tests running independently [6]. Therefore, the temperature profile of any test schedule can be generated by thermally simulating every test once and only once. From this, the power-constrained test scheduling method can be applied to temperature constraints by requiring the sum of the temperature impact of every test in every core to be less than the given temperature bound at any given time during test.

The introduction of 3D-ICs as a manufacturing technique has led to several different fault modeling techniques and hardware compatibility generation, but from the standpoint of SoC test scheduling stacking dies only increases the number of tests to schedule. 3D-IC testing initially focused on specific faults unique to 3D-ICs [15]. The stacking of dies creates faults involving through-silicon vias that are not present in single-die ICs, and studies attempted to create usable models for these faults. Another focus for 3D-IC studies is the effect stacking has on test-specific hardware and test scheduling [12]. Studies have attempted to find ways to optimally assign test access mechanism pins for the sake of minimizing TAT, which in a 3D-IC requires assigning connections between dies as well as to the base die. However, as stated in Section I this has the effect of assigning hardware compatibility between dies depending on pin assignments, which allows the SoC test model to be applied to 3D-ICs [2].

The special role temperature constraints play in 3D-IC test scheduling has been previously observed. Several studies have explored temperature constrained test scheduling under 3D-ICs [2], [7], [8]. The heat insulating nature of 3D-ICs has been observed to increase the temperature of devices during test. Further, it has been observed that the stacking nature of 3D-ICs makes heat flow between cores on different dies more easily than cores in non-stacked ICs [16]. This is because non-stacked ICs are relatively thin thus heat flow between cores is minimal, whereas the larger surface area of stacked dies allows for more heat flow between stacked dies, which in turn means more heat flow between cores and more difficulty in temperature-constrained test scheduling.

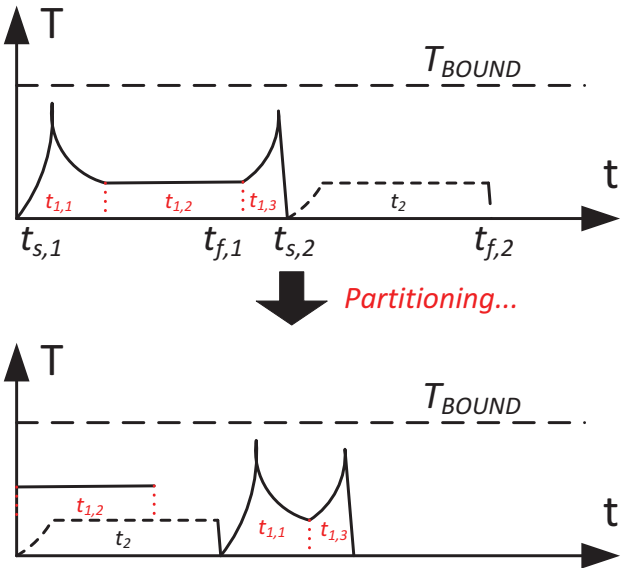

Fig. 1. By partitioning, greater overlap between tests t_1 and t_2 can be achieved.

Partitioning tests to reduce TAT has been studied under hardware and other constraints, but all previous partitioning methods have been very basic (trivial) and the effectiveness of a partitioning method has never been evaluated compared to other methods. In previous studies, partitioning has been clearly observed to provide the ability to reduce TAT. However, in all previous studies the method of partitioning was either simplistic or ad hoc.

III. Proposed Partitioning Method

A. Partition Quality

Before any partitioning method can be proposed, it must first be determined what qualities define a good as opposed to a bad partition.

Since the goal of test scheduling can be stated as creating as much parallelism in applying tests as possible so that TAT can be minimized, the objective of partitioning should be to allow for new parallelism amongst tests. The worst possible test schedule executes only one test at a time, while the best possible test schedule executes every test at $t = 0$. However, the latter often cannot be achieved due to various constraints. Therefore, the goal of partitioning should be to allow for constraints between two tests to be relaxed. In the case of hardware constraints, this cannot be done between two tests (although it is possible with 3 or more [9]) since new partitions will have the same hardware compatibility as their parent tests. It can be said that hardware constraints are "constant" throughout test, but this is not the case with temperature since temperature fluctuates during test. If parts of one test may overlap with parts of another provided those parts occurred in a certain order, than partitioning can give that order to the scheduler (see Figure 1 for an example).

For temperature-constrained test scheduling, two tests cannot be run in parallel if their combined temperature impact is greater than the given temperature bound, therefore the goal of partitioning should be to "lower" the temperature of a test to allow it to run in parallel with many more other tests. In reality, partitioning will not lower the temperature impact of the original test since partitioning will not change the power of the test (with the exception of partitioning overhead). Clearly, each partition of a given test can have no more temperature impact than the original test if the impact of the overhead of partitioning is ignored. Of course, if partitions of a given test are scheduled one after another, the temperature impact of a test will not change at all. However, if the partitions of a tests are instead viewed as new tests whose combined run time are equal to the original test, these new tests may have lower temperatures than the original test and therefore can be scheduled in parallel with other tests that the original test could not be scheduled in parallel with.

Because the peak temperature of a test is what determines the overlapping potential of two tests, another important objective of a temperature-constrained partitioning method should be to partition tests such that their temperature is constant throughout the test. The example in Figure 1 showed the peak temperatures of test t_1 not allowing for any overlap with t_2 with the exception of insignificant proportions of the start and finish of the test. It can be said that if a peak temperature value occurs in a test, then another test cannot overlap "beyond" when that value occurs if the combined temperature at this point violates a given bound. At the same time, if the temperature of a test is constant, then it can be assumed that any partitions of that test will have the same temperature as the original test (this presumption will be addressed later). This implies if two tests with constant temperature are incompatible, then no partitioning can make them compatible. Therefore, it should be the goal of a partitioner to divide tests in such a way that make the temperature of partitions as constant as possible, since partitioning any more will give no better result.

Now that it is known that the temperature of test partitions must be as constant as possible, the goal is to create such partitions in a practical manner, which would normally be difficult due to the simulation-intensive nature of temperature profiles. As suggested in Section II, generating a temperature profile from a given power profile requires a time-intensive thermal simulation. Since partitioning a test creates a new power profile for each partition, simulating the temperature of all relevant partitions whenever a partition is made is impractical.

However, the observation that the temperature of a partition will never be greater than the original test can allow for a reasonable estimation of partition temperatures based on the temperature of the original test. Partitions of a test share the same power profile of the original test (plus scan overhead) and are executed in the same environment as the original test. Therefore, the temperature profile of a partition is guaranteed to be equal to or less than that of the original test. This is because temperature is function of current and past power dissipation, and partitioning can only remove past power values. Additionally, the longer a partition is, the more likely the temperature profile of the partition will match the original test. This is because in the longer partition, past power values will more closely resemble that of the original test.

Based on the observations that the temperature of a partition can be reasonably estimated from the original test and the temperature of each portion should be reasonably constant, a partitioning method is now proposed to achieve such goals.

B. Partitioning Method

The goal of the proposed partitioning method is to split each test into high and low-temperature partitions based on the temperature of the original test, thereby making the temperatures of each partition constantly high or constantly low. By doing this, high-temperature regions of tests that are normally incompatible with every other test due to temperature constraints will be isolated, while low-temperature regions are (temperature) compatible with every other test. This separation will allow for the maximum overlap without violating temperature constraints.

The first step of the partitioning method is to choose a temperature that defines the difference between high and low temperature. This temperature is referred to as the "partitioning temperature". Based on the assertion made in Section III-A, the temperature of these partitions will be lower than the original test or will be a close match to the original test if the partitions are long enough. Note that if every test is a low temperature test, than no test will be partitioned. This is not undersirable, since temperature will not play an important role in the test and therefore will not have a significant role in scheduling. When a partitioning temperature is chosen, each test is partitioned when the temperature of the test crosses the specified temperature as long as other conditions are met.

The first condition to partitioning is that any partition generated is not exceptionally small (in this study, less than $30\mu s$ long), the reason for this being that small partitions will always have low temperature, even if the temperature of the original test is high. Also, if partitioning overhead is a factor, small partitions will have relatively large overhead of partitioning which defeats the purpose of partitioning. In the case of temperature rising above the partitioning temperature and then falling below less than $30\mu s$ later, a single partition will be made thereby eliminating this "temperature peak".

The second condition is that tests are partitioned not at the point where the temperature passes the partitioning temperature, but instead shortly before that point (in this study, $30\mu s$). As stated before, the temperature of a partition is dependent on the past values as well as the current value. If partitioning a test creates two partitions, "partition one" and "partition two", then this will decrease the maximum temperature of partition one while keeping the maximum temperature of partition two the same as long as the length of partition two is not exceptionally small. This is because "past values" are removed from partition one and added to partition two. An example of this is illustrated in Figure 2.

IV. Experiment

To judge the effectiveness of the proposed partitioning method in 3D-ICs, it must be compared against other partitioning methods. Although partitioning has been done in other studies, other studies used simple methods such as partitioning arbitrarily, partitioning for equal-sized partitions, or partitioning only when absolutely required (running of a test by itself violates the given temperature bound). This study implements both of these methods for comparison, as well as scheduling without partitioning.

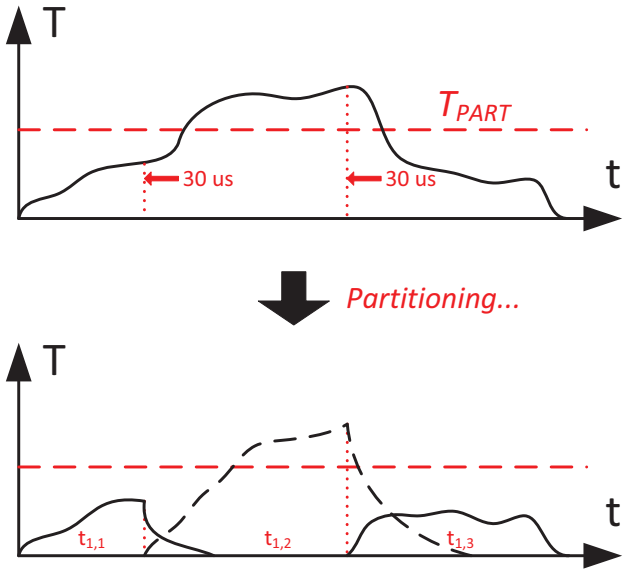

Fig. 2. By partitioning before T_{PART} is met, the temperature of all partitions is reduced.

To fairly compare the proposed partitioning method and partitioning with an equal number of equal-sized partitions, the proposed method is first implemented and the number of partitions created with equal-sized partitioned is set equal to the number of partitions created by the proposed method. This is done because the number of partitions can effect the quality of the schedule, and it would be an unfair comparison if one partitioning method creates more partitions than the other.

To observe the effect the number of partitions on the effectiveness of the the different partitioning methods, partitioning for each benchmark is done twice: once with a fewer number of partitions, and once with a greater number of partitions. As stated earlier, when temperature-based partitioning is done, the number of partitions created for each test is recorded and equal-sized partitioning is done with the number of partitions specified. In this study, it is done twice by partitioning with partitioning temperature equal to 40° C (fewer partitions) and 30° C (larger number of partitions). These two temperatures were chosen for T_{PART} since as T_{PART} increases, the number of partitions created decreases. This is illustrated in Figure 3, which shows the number of partitions created in the selected benchmarks described later. The 3D-IC benchmarks provided in [17] are used to compare the effect of the different partitioning methods, as well as to show the difficulty involved with temperature-constrained test scheduling of 3D-ICs. These 3D-IC benchmarks consist of ITC'02 benchmarks [18], with each ITC'02 benchmark representing a die. In some benchmarks, ITC'02 dies are stacked to create 3D-ICs. The specific stacking method is face-to-back stacking with 5% of die surface area used for evenly-spaced thermal through-silicon vias (TTSVs) for multi-die stacks. "Markov Model" style power traces are generated for each test to imitate actual power traces for tests

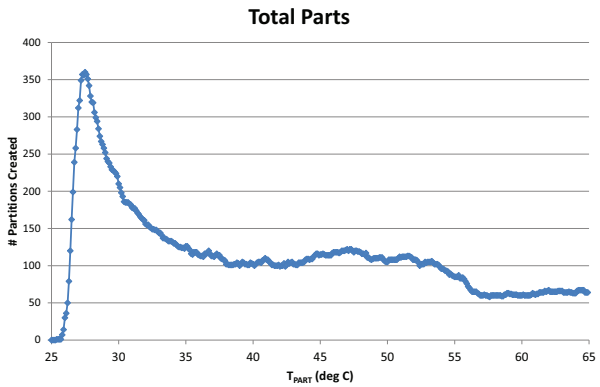

Fig. 3. Effect of T_{PART} on the number of partitions.

TABLE I
BENCHMARK DETAILS

Bench #	3D-IC Bench	# Dies	# Tests	Highest Temperature
1	1-1-LLL	1	9	38.07
2	1-6-LML	1	4	39.25
3	1-7-LHL	1	4	48.66
4	1-9-MML	1	19	42.09
5	2-4-LML	2	12	35.21
6	2-6-MMM	2	23	42.30
7	2-8-MHM	2	35	66.76
8	2-9-HHM	2	63	62.04
9	2-10-MHH	2	38	68.98
10	3-2-MMM	3	33	46.39
11	3-3-LML	3	22	48.47
12	3-4-HMM	3	79	44.14
13	3-5-MHH	3	54	82.80

[6]. The Hostpot Thermal Simulator [19] is used to generate temperature profiles for tests using power traces under a 25° C ambient temperature. Hardware compatibility is pre-defined with the presumptions that each die can have tests applied simultaneously and incompatibility within a die is assigned by "cliques" of cores sharing test hardware. Details on the benchmarks used is listed in Table I, which provides the number of stacked dies in each benchmark, the number of tests to schedule, and the highest temperature achieved (in degrees celsius) by any given test without partitioning. The subset of benchmarks was chosen for both their practical scheduler runtime (since some benchmarks take unduly long time to schedule due to the number of cores in the benchmark), and due to their longer test times, since benchmarks with only short tests will not have any significant impact on the temperature of a device regardless of the power dissipation. To use as a scheduler, the MILP-based formulation proposed in [2] is used. This scheduler was chosen because of its deterministic nature (the same result will always be achieved) and because of its performance. The scheduler is an optimistic scheduler, meaning it may produce a schedule that will violate temperature constraints. Because of this, schedules are re-simulated after they are produced to check validity. T_{BOUND} for scheduling is set to 55° C, which is intentionally set below the temperature bound for some benchmarks. This is done to show the requirement of partitioning for scheduling tests for 3D-ICs. The CPU computation time is not measured in the experiment since the scheduling time is the same across all partitioning methods and the partitioning time is trivial for every proposed partitioning method (except minimal partitioning at T_{BOUND} only, which requires successive temperature simulations).

TABLE II
SCHEDULE RESULTS (TAT) FOR T_{PART} = 40° C IN MS

Bench #	No Parts	Min. Part.	Equal Part.	Temp. Part.
1	257.4	257.4	257.4	257.4
2	28.36	28.36	28.36	28.36
3	311.72	311.72	311.72	311.72
4	37.24	37.24	37.24	37.24
5	16.84	16.84	16.84	16.84
6	42.35	42.35	31.50	31.52
7	X	500.54	358.68	358.68
8	X	335.56	288.4	269.98
9	X	945.02	757.04	757.04
10	19.65	19.65	16.88	17.04
11	226.23	226.23	150.96	138.03
12	40.66	40.66	30.26	30.24
13	X	900.56	771.70	823.42

V. RESULTS

Table II gives the results of partitioning with partitioning temperature equal to 40° C. The table gives the schedule length, i.e. TAT (in ms), provided by the scheduler when no partitioning is done, when minimal partitioning (partition only when T_{BOUND} is violated) is done, when equal partitioning is done, and when temperature-based partitioning is done.

The results from Table II show several trends that not only show the effect stacking dies has on temperature-constrained test scheduling, but also show the proposed partitioning method performing well in conditions where temperature has a significant impact on test scheduling. For every benchmark, the result for the proposed partitioning method is always better than or equal to without partitioning. This is predictable, since the effect partitioning has on possible test schedules has been explored. Also, some benchmarks fail to find a solution without partitioning. This is because these benchmarks have one or more tests which violates the temperature constraint. This is more common as the number of dies in the 3D-IC increases, which is due to the heat-insulating nature of 3D-ICs. For benchmarks which are not 3D-ICs but instead conventional one-die ICs, all benchmarks have no difference between the partitioning methods. This is because the temperature impact of the tests are minimal. When the number of dies increases, the difference between equal and temperature-based partitioning increases, which is again due to the effect stacking dies has on temperature. Although temperature-based partitioning does not always yield the best result, it is more likely to give a better result, which is shown in Figure 4. Table III gives the results when the partitioning temperature is set to 30° C, which shows the effect increasing the number of partitions has on the effectiveness of partitioning. The results of this table show the same general trend as shown in Table II: the partitioning gives better schedules than not partitioning, and temperature-based partitioning is more likely to give superior schedules than equal partitioning, especially when temperature plays a significant role.

Figure 4 gives the normalized TAT of all partitioning methods on relevant benchmarks, which in turn gives insight into the effect increasing the number of partitions has on schedule quality. The normalized TAT of the partitioning

TABLE III
SCHEDULE RESULTS (TAT) FOR T_{PART} = 30° C IN MS

Bench #	No Parts	Min. Part.	Equal Part.	Temp. Part.
1	257.4	257.4	257.4	257.4
2	28.36	28.36	28.36	28.36
3	311.72	311.72	311.72	311.72
4	37.24	37.24	37.24	37.24
5	16.84	16.84	16.84	16.84
6	42.35	42.35	35.84	35.84
7	X	500.54	416.60	373.58
8	X	335.56	289.62	274.68
9	X	945.02	759.12	709.98
10	19.65	19.65	17.32	17.32
11	226.23	226.23	192.46	190.72
12	40.66	40.66	32.38	32.26
13	X	900.56	771.70	792.96

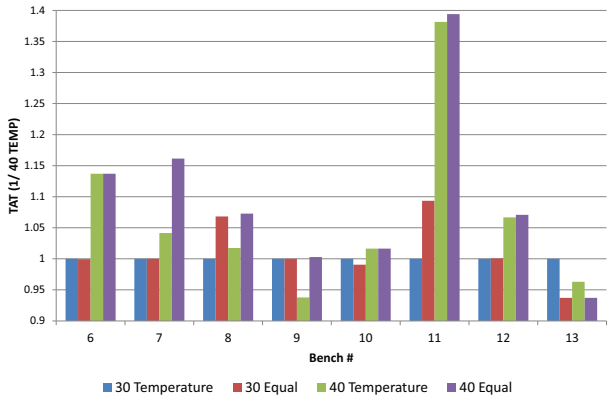

Fig. 4. Normalized schedule results of different partitioning methods.

methods clearly show that more partitions generally lead to a better schedule. This is so because more partitions create more opportunity for test overlap. However, this is not always the case, like with benchmark numbers 4 and 13, since the quality of a partition is dependent on more than just the number of partitions created. Figure 4 also shows that the majority of the time temperature-based partitioning performs better than equal partitioning. Again, this is not always the case, such as in benchmark number 13.

VI. CONCLUSION

The results of test scheduling under temperature constraints show the great effect stacking dies has on the difficulty of scheduling under temperature constraints. The thermal insulating nature of 3D-ICs has a significant impact during test, for the overheating of dies creates yield loss and increases manufacturing costs. Special attention and techniques need to be applied to temperature constraints while testing 3D-ICs, and the partitioning of tests shows a substantial improvement in schedule quality.

The results have shown both the effect partitioning has on test schedule quality and the effectiveness of partitioning with temperature in mind. Although partitioning has always been known to crate higher quality test schedules, specific methods of partitioning were never evaluated. This study has provided such a partitioning method and evaluated it against other methods, and has shown the effect a specific partitioning method can have on the quality of a schedule, especially in environments where temperature has a significant impact such as 3D-ICs.

REFERENCES

[1] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits*. New York, New York, USA: Kluwer Acedemic Publishers, 2000.

[2] S. K. Millican and K. K. Saluja, "Linear Programming Formulations for Thermal-Aware Test Scheduling of 3D-Stacked Integrated Circuits," in *2012 IEEE 21st Asian Test Symposium*. Niigata, Japan: IEEE, Nov. 2012, pp. 37–42.

[3] C. Kime and K. Saluja, "Test Schduling in Testable VLSI Circuits," in *International Symposium on Fault-Tolerant Computing*. Santa Monica: IEEE, 1982, pp. 406–412.

[4] R. Chou, K. K. Saluja, and V. D. Agrawal, "Scheduling Tests for VLSI Systems under Power Constraints," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, no. 2, pp. 175–185, Jun. 1997.

[5] D. R. Bild, S. Misra, T. Chantemy, P. Kumar, R. P. Dick, X. S. Huy, and A. Choudhary, "Temperature-aware test scheduling for multiprocessor systems-on-chip," in *IEEE/ACM International Conference on Computer-Aided Design*. IEEE, Nov. 2008, pp. 59–66.

[6] C. Yao, K. K. Saluja, and P. Ramanathan, "Power and Thermal Constrained Test Scheduling Under Deep Submicron Technologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 2, pp. 317–322, Feb. 2011.

[7] Z. He, Z. Peng, and P. Eles, "A heuristic for thermal-safe SoC test scheduling," in *IEEE International Test Conference*. IEEE, 2007, pp. 1–10.

[8] F. A. Hussin, T. E. C. Yu, T. Yoneda, and H. Fujiwara, "RedSOCs-3D: Thermal-safe test scheduling for 3D-stacked SOC," in *2010 IEEE Asia Pacific Conference on Circuits and Systems*. IEEE, Dec. 2010, pp. 264–267.

[9] G. Craig, C. Kime, and K. K. Saluja, "Test scheduling and control for VLSI built-in self-test," *IEEE Transactions on Computers*, vol. 37, no. 9, pp. 1099–1109, 1988.

[10] C. Yao, K. K. Saluja, and P. Ramanathan, "Partition Based SoC Test Scheduling with Thermal and Power Constraints under Deep Submicron Technologies," in *Asian Test Symposium*. IEEE, 2009, pp. 281–286.

[11] Z. He, Z. Peng, and P. Eles, "Power Constrained and Defect-Probability Driven SoC Test Scheduling with Test Set Partitioning," in *Design, Automation & Test in Europe*. IEEE, 2006, pp. 1–6.

[12] B. Noia, K. Chakrabarty, and S. Goel, "Test-Architecture Optimization and Test Scheduling for TSV-Based 3-D Stacked ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 11, pp. 1705–1718, 2011.

[13] P. Girard, N. Nicolici, and X. Wen, *Power-Aware Testing and Test Strategies for Low Power Devies*. Springer, 2010.

[14] P. Rosinger, B. M. Al-Hashimi, and K. Chakrabarty, "Thermal-Safe Test Scheduling for Core-Based System-on-Chip Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2502–2512, Nov. 2006.

[15] C. Jun, L. Yudong, and L. Guoyuan, "Reliability of package on package (PoP) subjected to thermal and power loadings," in *2010 11th International Conference on Electronic Packaging Technology & High Density Packaging*. IEEE, Aug. 2010, pp. 1023–1026.

[16] Z. He, Z. Peng, P. Eles, P. Rosinger, and B. M. Al-Hashimi, "Thermal-Aware SoC Test Scheduling with Test Set Partitioning and Interleaving," in *IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*. IEEE, Oct. 2006, pp. 477–485.

[17] S. K. Millican and K. K. Saluja, "3D-IC Benchmarks," 2013. [Online]. Available: http://3dsocbench.ece.wisc.edu/

[18] E. Marinissen, V. Iyengar, and K. Chakrabarty, "A set of benchmarks for modular testing of SOCs," in *Proceedings. International Test Conference*. Baltimore, MD: IEEE, Oct. 2002, pp. 519–528.

[19] K. Skadron, M. Stan, W. Huang, and D. Tarjan, "Temperature-aware microarchitecture," in *30th Annual International Symposium on Computer Architecture, 2003. Proceedings.* IEEE Comput. Soc, 2003, pp. 2–13.