2015 28th International Conference on VLSI Design and 2015 14th International Conference on Embedded Systems

Optimal Test Scheduling of Stacked Circuits Under Various Hardware and Power Constraints

Spencer K. Millican & 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*—As Integrated Circuits (ICs) become more complex through smaller semiconductor feature sizes and higher performance requirements, the thorough testing of silicon devices is becoming a greater economic challenge. System-on-Chip (SoC) test schedules not only need to achieve the shortest possible test application time, they must also satisfy new design constraints which are increasing test scheduling complexity, such as device power. Although many past studies of SoC test scheduling have addressed individual issues during test, none have present a model of test scheduling that has allowed for many different constraints to be enforced at once. By presenting a test scheduling formulation that allows the enforcement of many separate power and hardware constraints, including issues of test pins, as well as allowing the use of modern Dynamic Voltage and Frequency Scaling (DVFS) hardware to further compact test schedules, this study provides a generalized test scheduling formulation that will not only allow the enforcement of multiple testing constraints but will also allow for the further compaction of test schedules and reduction of test cost.**

I. Introduction

A continuing challenge in IC test is the reduction of manufacturing test cost through the reduction of IC test time. Ever since complex circuits were implemented on silicon dies, great resources have been spent to find flaws and defects created by the IC manufacturing process. Although one would prefer to spend ample time to find all possible faults in the circuit, more time spent testing a device leads to greater testing costs due to the high cost of testing equipment [1]. Unfortunately, as the complexity of ICs has increased with the decrease in silicon feature sizes, so has the number of faults on a given die, so much so that testing of all individual faults in an entire circuit has become impractical.

The introduction of SoC designs has not only assisted with the IC design process, but have also provided a simpler model for the reduction of IC Test Application Time (TAT). A SoC is a design which is partitioned into several distinct cores, with each core having distinct role in the circuit. Although SoC design methodology has been used substantially in the past due to its ability to simplify the IC design process, SoC designs have become more commonplace through the introduction of Intellectual Property (IP) core companies which specialize in selling pre-made cores. The benefit of buying pre-made IP cores is the elimination of development time of individual cores, while the IP core provider benefits by re-selling their core to multiple parties. In order for a SoC designer to test their finished design, the IP core provider provides a series of tests to the SoC designer, with the guarantee that these tests will achieve near one-hundred percent fault coverage. The SoC tester's goal is to apply tests for all cores in the shortest possible time.

As feature sizes of ICs are reduced, new constraints, like power during test, increase the complexity of SoC test scheduling. One consequence of the decreasing feature sizes of ICs is that the power during circuit operation has increased to the point where it cannot be ignored, especially during test when the power consumption of ICs is more than during normal operation [2]. An IC which consumes too much power during test can either cause a "false failure" by consuming more power than the power supply can provide, or high power consumption can lead to higher device temperatures, which in turn can lead to silicon damage. A SoC test scheduler must therefore produce a schedule which does not damage the device or produce a false failure.

The complexity of testing SoCs is further increased with the introduction of new hardware packaging methods, such as 3D-Stacked ICs (3D-ICs). In a 3D-IC, multiple silicon dies are stacked before being packaged together in the final design. This design method has several benefits, namely allowing a device to have a smaller footprint (a must in mobile computing applications) and decreased die interconnect lengths, which can give higher device operating speeds. Although such designs are useful from a designers point of view, they create difficulty for SoC testers, as they add new hardware constraints to the test scheduling process [3].

However, new technologies, like DVFS, have given new ways to reducing the test time of ICs. DVFS was originally introduced for the purpose of reducing device power consumption of high-performance processors when their full performance is not required, initially done by scaling down the device frequency to reduce dynamic power consumption. Later, voltage was scaled down as well, since high voltage values are not needed to operate devices at lower frequencies, which in turn lead to lower dynamic and static power consumption. DVFS can be utilized in multiple ways for testing purposes, such as to reduce the power consumption of tests, thereby allowing tests that normally could not be executed together to do so.

Although many past studies have addressed individual issues of SoC test schedules, few have incorporated multiple issues at once. Most previous SoC test scheduling studies have chosen

1063-9667/15 $31.00 © 2015 IEEE
DOI 10.1109/VLSID.2015.88

to select only a single issue to remedy at a given time, only occasionally choosing to combine more than one issue. For instance, several studies address 3D-IC testing [1], [3], [4] and several address DVFS in test scheduling [5]–[7], but none (to the authors' knowledge) address both. Also, many previous test scheduling studies make unnecessary presumptions on the test scheduling environment, thus limiting their ability to find truly optimal test schedules.

By failing to incorporate several issues at once or by making unnecessary assumptions on the test scheduling process, not only are all testing constraints not being enforced, but also not all methods of reducing TAT are being utilized. As the goal of SoC test scheduling is always to reduce TAT to its minimum in order to reduce testing costs while not providing an invalid test schedule, a test scheduling method must be developed to do so. This paper presents an optimal test scheduling formulation which allows for static and dynamic hardware constraints, power constraints, 3D-IC hardware constraints, and the use of DVFS during test.

The remainder of this paper is organized as follows. Section II provides past work in the field of SoC test scheduling and the issues they addressed. Section III provides an optimal SoC test scheduling formulation that addresses multiple scheduling constraints. Section IV describes a series of experiments to evaluate the provided formulation, and Section V gives the results of these experiments. Section VI give a discussion about the incorporation of other various constraints in the formulation, and the paper concludes with Section VII.

II. RelatedWorks

The first study on SoC test scheduling focused on pre-defined hardware constraints created by shared hardware between SoC cores. It is common practice for SoCs to share common hardware modules between cores, especially memory and input/output core, but this shared hardware makes it impossible to apply tests for cores that share hardware simultaneously. To solve this, [8] proposed creating a "hardware compatibility graph" and scheduling tests in such a way that only connected modules on the graph could be tested simultaneously. This problem was solved optimally by using optimal graph sub-clique algorithms.

Further studies into hardware-constrained SoC testing introduced the concept of generating hardware constraints during the testing process by giving control of test hardware allocation to the scheduler in the form of Test Access Mechanism (TAM) pin allocations. Since resources like Built-in Self-test (BIST) or TAM pins are test-specific, it may be possible for the test scheduler to allocate these resources for achieving better TATs in SoC test schedules which static hardware constraints cannot. A SoC test scheduler can be given a constraint in the form of the total number of TAM pins allowed in the design, and the scheduler can then schedule tests while enforcing this constraint. In general, having more TAM pins available to the scheduler can greatly decrease the TAT [9], but there is often a constraint on the number of TAM pins allowed since allocating space for test-specific pins is expensive.

As new design methods were introduced, like 3D-ICs, hardware-constrained test scheduling formulations were expanded for these new environments. The act of testing a single die in a 3D-IC is practically the same as testing a non-3D-IC, as the pins needed to test the die are more or less available when the die is not stacked, but the testing of an entire stack must still be done to confirm no faults are created during the stack process and to reduce overall testing costs [1]. In order to test entire stacks, test schedulers have implemented more complex TAM constraints based on the new hardware required to access TAM pins on stacked dies [10]. An example of scheduling tests using such new hardware is shown in [3].

As the power of devices increased to critical levels, the power constraint was enforced by interpreting power as a hardware constraint, i.e. two or more tests are hardware incompatible if their combined power is above a given power bound [11]. As DVFS was introduced for power management, frequency-only scaling was introduced [5] into the test scheduling problem, followed by voltage and frequency scaling [6], [7], [12]. In these methods, and others like them, the power of all tests executed at any given time is accumulated, and this accumulation is forced to be less than the given power bound.

Due to the complexity (NP-hard) of finding optimal test schedules, many SoC test scheduling methods are driven by heuristics. The main motivation behind using a heuristic, as opposed to an optimal solver, is the reduction in time to find a solution. Most SoC test scheduling heuristics take the form of a list-based algorithm, where the final schedule depends on the order of the input test list.

The scheduling method presented in this paper will not only incorporate all the previous SoC constraints into a single optimal scheduling formulation, but will also overcome the sessions-based scheduling barrier to achieve higher quality schedules. Although many of the scheduling issues described here have been addressed individually, only some of these issues have been addresses together. Testing 3D-IC with power constraints has been addressed [4], testing under power constraints with TAM pin constraints has been addressed [13], but never have all of these issues been addressed at once, especially without making sub-optimal assumptions involving test sessions and heuristics. The formulation given in Section III will allow for optimal scheduling under all of these constraints.

III. Test Scheduling Formulation

The goal of any scheduling formulation is to minimize the total TAT of the schedule (t_{finish}) which is the maximum of the finish time of each test ($F(t)$). This constraint can be stated in a linear manner as follows:

$$\forall t \in T : t_{finish} \geq F(t)$$

The finish time of each test is a function of the start time of each test ($S(t)$) and the runtime of each test ($L(t)$).

$$\forall t \in T : F(t) = S(t) + L(t)$$

$$\forall t \in T : S(t) \geq 0$$

The runtime of each test is a function of two variables: what voltage/frequency a test is run at, and the number of TAM pins available to the test. This is noted by the variable $\psi(t, n, c)$. This formulation presumes that each test t has a given number of choices available as to what voltage/frequency it is executed at ($n \in N_t$) and how many pins can be assigned to the core ($c \in C_t$). Here, n and c are not the specific voltage/frequency and number of pins, but only an index (e.g., for a 6 voltage/frequency choices and 4 TAM pin amount choices, $n = 1 \ldots 6 = |N_t|$ and $c = 1 \ldots 4 = |C_t|$). The actual voltage/frequency corresponding to each n is irrelevant for scheduling, and the number of TAM pins associated with each pin count c is discussed later. Below, the constant $L_{t,n,c}$ denotes the runtime of test t when executed at DVFS index n and pin index c.

$$\psi(t, n, c) = \begin{cases} 1 & \text{if the test } t \text{ is executed at frequency selection } n \text{ with pin selection index } c \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in T : \sum_{\forall n \in N_t} \sum_{\forall c \in C_t} \psi(t, n, c) = 1 \qquad (1)$$

$$\forall t \in T : L(t) = \sum_{\forall n \in N_t} \sum_{\forall c \in C_t} \psi(t, n, c) \cdot L_{t,n,c}$$

To enforce many constraints, it is necessary to know which tests overlap with each other. To achieve this, the method from [14] is borrowed. This method uses a binary variable $\eta(t_1, t_2)$ to give a time relationship between every pair of tests, which in turn can be translated to a binary variable $\Omega(t_1, t_2)$ which describes if two tests overlap. Below, λ_L is the TAT of the longest possible test schedule (i.e., ever test executed serially at their lowest frequency with the fewest number of pins).

$$\eta(t_1, t_2) = \begin{cases} 1 & \text{if the test } t_1 \text{ finishes before the test } t_2 \text{ begins} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1, t_2 \in T : \eta(t_1, t_2) + \eta(t_2, t_1) \leq 1$$

$$\forall t_1, t_2 \in T : F(t_1) \leq S(t_2) + (1 - \eta(t_1, t_2)) \cdot \lambda_L$$

$$\forall t_1, t_2 \in T : S(t_2) \leq F(t_1) + \eta(t_1, t_2) \cdot \lambda_L$$

$$\Omega(t_1, t_2) = \begin{cases} 1 & \text{if the test } t_1 \text{ overlaps with the test } t_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1, t_2 \in T : \Omega(t_1, t_2) = 1 - \eta(t_1, t_2) - \eta(t_2, t_1)$$

To enforce power and TAM pin constraints, it is useful to note that the enforcement of such constraints is only necessary when a test begins. Since the number of TAM pins cannot increase when a test is in execution, and neither can the power consumed by a test (presuming a constant power model or maximum power model), checking a power or TAM pin constraint during test execution is unnecessary since this constraints cannot change. The checking of these constraints when a test finishes is also not necessary, since the power usage and TAM pin usage can only decrease under the same presumptions. To assist in the checking of constraints only when tests begin, a new binary variable "$\pi(t_1, t_2)$" is introduced.

$$\pi(t_1, t_2) = \begin{cases} 1 & \text{if the test } t_1 \text{ begins before or when the test } t_2 \text{ begins} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1, t_2 \in T : S(t_1) \leq S(t_2) + (1 - \pi(t_1, t_2)) \cdot \lambda_L$$

$$\forall t_1, t_2 \in T : S(t_1) > S(t_2) - \pi(t_1, t_2) \cdot \lambda_L$$

$$\forall t \in T : \pi(t, t) = 1$$

Using $\pi(t_1, t_2)$, constraints can be enforced when any test starts its execution by way of the following. A new variable "$P(t_1, t_2)$" is introduced, which is equal to the power of the test t_2 when test t_1 starts its execution. If the tests t_1 and t_2 do not overlap, $P(t_1, t_2)$ must be zero. The same is true if t_2 starts its execution after t_1 starts. However, if both of these conditions are not the case, then $P(t_1, t_2)$ is equal to the power of t_2. Below, the constant $P_{t,n}$ denotes the power of test t executes at DVFS index n. Below, λ_P is the highest possible power achievable in any test schedule (i.e., the power of every test executed simultaneously at the highest possible voltage/frequency).

doubt

$$\forall t_1 \in T : \sum_{\forall t_2 \in T} P(t_1, t_2) \leq P_{BOUND}$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \leq \Omega(t_1, t_2) \cdot \lambda_p$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \leq \pi(t_2, t_1) \cdot \lambda_p$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \geq \sum_{\forall n \in N_{t_2}} \sum_{\forall c \in C_{t_2}} \psi(t_2, n, c) \cdot P_{t_2,n} - (1 - \Omega(t_1, t_2)) \cdot \lambda_p - (1 - \pi(t_2, t_1)) \cdot \lambda_p$$

A TAM pin constraint can be enforced using the same method which enforces the power constraint. Below, $C(t_1, t_2)$ is the number of pins used by test t_2 at the time when test t_1 begins, and the constant $C_{t,c}$ is the number of pins used by test t at TAM pin index c. Below, λ_c is the maximum possible number of pins that can be utilized in any test schedule (i.e., the sum of all tests using their maximum number of pins).

$$\forall t_1 \in T : \sum_{\forall t_2 \in T} C(t_1, t_2) \leq C_{BOUND}$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \leq \Omega(t_1, t_2) \cdot \lambda_c$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \leq \pi(t_2, t_1) \cdot \lambda_c$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \geq \sum_{\forall n \in N_{t_2}} \sum_{\forall c \in C_{t_2}} \psi(t_2, n, c) \cdot C_{t_2,c} - (1 - \Omega(t_1, t_2)) \cdot \lambda_c - (1 - \pi(t_2, t_1)) \cdot \lambda_c$$

In order to test stacked dies in a 3D-IC, Through-Silicon Vias (TSVs) are required to transport test data between dies.

Testing a stacked die requires the availability of TAM pins on the lowest die as well as sufficiently many TSVs beneath lower dies to transport data upwards. Below, $V(d)$ denotes the number of TSVs (available for testing purposes) under each die $d \in D$. Since the first (bottom) die requires TSVs to access it, its value is pre-determined ($V(1) = 0$). Every die above the first (bottom) die has the requirement that enough TSVs must exist to test that die and every die above at any time in the scheduling process. For example, if there is a time in the schedule when 8 pins and 7 pins are required to test the second and third dies receptively, then the second die requires 15 TSVs beneath it and the third die requires 7 TSVs beneath it. Therefore, $V(d)$ in every die (except the bottom die) must have the property that at any given time it must be larger than the total number of TAM pins currently being utilized by all dies above it. Since the number of pins used by any given test can only increase when a test starts, $C(t_1, t_2)$ can be re-used to check the number of pins in use at any given time by checking it for every test t_1. Below, T_d denotes all tests in die d and dies above it.

$$\forall d \neq 1 \in D, t_1 \in T_d : V(d) \geq \sum_{\forall t_2 \in T_d} C(t_1, t_2)$$

$$\sum_{\forall d \in D} V(d) \leq TSV_{BOUND}$$

IV. Experiments

The purpose of the first experiment is to evaluate the usefulness of the formulation when constraints are added or removed. Since there are no other scheduling formulations which incorporate all of the constraints incorporated in this formulation, a fair comparison of this formulation against any other is not possible. However, the effect on the final schedule result can be observed when constraints are removed and/or added, which will show the effectiveness of the scheduling method in selected environments. The first experiment will show the effect of the scheduling method when no power constraint is enforced, when no DVFS is allowed during test (i.e., only 1 V and its corresponding frequency is allowed during scheduling), and when tests are required to be in sessions. In the case when no power constraint is enforced, power violations in the schedule will also be recorded.

The second set of experiments will observe the effect the TAM pin and TSV constraints have on the overall schedule quality. It is clear that allowing for more TAM pins and more TSVs during the scheduling process can lead to a shorter test scheduling, but given the high cost of TSVs and TAM pins during the design process, adding such hardware may not be worth a marginal increase in schedule quality. The second experiment will show specifically how allowing such hardware during the scheduling process will affect the test schedule quality.

This study will use benchmarks derived from the ITC'02 SoC benchmarks [15] with added power and with some benchmarks being stacked. Each ITC'02 benchmark consists of a set of cores, with each core having one or more tests.

TABLE I
Benchmark Information

Bench	ITC'02 Benchmarks	# Tests	Max Pow	P_{BOUND}
1	u226	9	0.78	1.02
2	g1023	14	3.26	11.40
3	f2126	4	4.26	3.84
4	q12710	4	34.50	41.86
5	a586710	7	3.90	11.30
6	u226, g1023	23	3.26	12.43
7	u226, f2126	13	4.26	4.87
8	u226, q12710	13	34.50	42.88
9	u226, a586710	16	3.90	12.32
10	g1023, q12710, u226	27	34.50	54.29
11	f2126, q12710, u226	17	34.50	46.73
12	q12710, a586710, u226	20	34.50	54.18
13	g1023, q12710, a586710	25	34.50	64.56
14	g1023, f2126, a586710	25	4.26	26.55
15	g1023, f2126, q12710	22	34.50	57.11
16	u226, q12710, a586710	20	34.50	54.18

Each core also has pin and scan chain information. However, with few exceptions, the ITC'02 benchmarks do not include power information. In this study, power values are assigned to cores based on their pin information. It is assumed that the more pins a core has, the larger the core area, and therefore the larger power consumption when the core is active. It is presumed that the largest ITC'02 benchmark (a586710) has a die 10 mm x 10 mm footprint and a 1.5 W/mm^2 test power density, which in turn gives a W/pin value to use across all benchmarks. This power value presumes a standard operating frequency and voltage of 120 MHz and 1 V. A power bound of 1 W/mm^2 is set for each benchmark, since it is presumed that the full power of all devices on a die being simultaneously tested will either damage the device or cause a false failure [2]. 3D-ICs are created by combining several ITC'02 benchmarks together. Specific information on the benchmarks can be found in Table I, with stacked dies listed in order of lowest to highest.

Other scheduling constraints are as follows. It is presumed that when scheduling tests under DVFS, voltages of 1, 0.9, 0.8 and 0.7 V are available to the scheduler for each core (i.e., $N_t = 4$ for every test t), and the standard testing frequency is 120 MHz. The frequencies corresponding to these voltages (required to find the constants $P_{t,n}$ and $L_{t,n,c}$) are found using the method given in [6], which is the highest possible operating frequency for a given voltage. To find the number of relevant pin choices available for individual tests and the associated TAT (at 1 V and the corresponding highest frequency) corresponding to these pin choices, the method from [9] is used. Mixed-Integer Linear Programming (MILP) formulations are solved used IBM/ILOG CPLEX.

V. Results

Table II gives the results of the first experiment described in Section IV. These results have a constraint of 5 data-carrying TAM pins per die and 5 data-carrying test TSVs per die (the number of pins and TSVs which control test execution are not incorporated as the number need not change depending on the schedule [3]). The table gives the TAT (in milliseconds)

TABLE II
SCHEDULE RESULTS

Bench	Proposed TAT (ms)	No DVFS TAT (ms)	No Pow TAT (ms)	Slack (W)
1	1365.462	1364.605	1365.462	-0.02159
2	77.516	78.285	75.573	-0.530
3	644.236	644.236	644.236	0.07999
4	4739.51	5576.448	3940.106	-31.184
5	158898.107	165546.541	158898.107	-3.584
6	1365.667	1364.703	1365.667	0.032
7	1794.498	1794.498	1794.498	0.12
8	5363.602	6405.573	4908.438	-9.2422
9	158898.107	171561.368	158898.107	-1.304
10	7204.977	7209.633	6536.602	-11.984
11	7204.977	7209.633	6536.602	-8.944
12	177631.934	177631.934	177631.934	-9.226
13	158898.107	158898.107	158898.107	2.968
14	158898.107	167038.047	158898.107	-5.264
15	4739.51	6121.938	4100.844	-16.060
16	158898.107	158898.107	158898.107	2.968

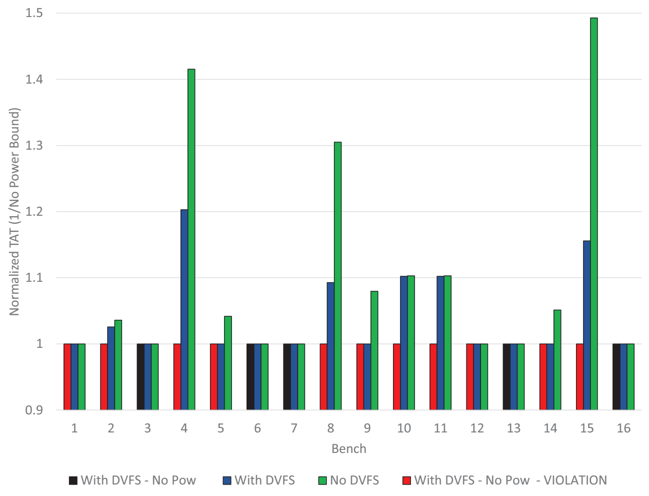

Fig. 1. Normalized TAT results with invalid schedules marked.

of each benchmark of the proposed formulation, the proposed formulation without DVFS, and when no power constraint is enforced. The table also gives the power slack (i.e., the power bound minus the peak test power, in Watts) when no power bound is enforced, with a negative slack implying that the schedule violates the power bound.

Table II shows interesting trends with regards to the impact DVFS has on the schedule quality, and with regards to the effect a power bound has on the TAT. When comparing the schedule quality when DVFS is and is not allowed, it is clearly that DVFS can have a benefit on TAT. Although DVFS can make individual TATs longer for each test (since 1 V allows for the fastest execution of any single test), it appears that DVFS creates enough opportunities for overlap under power constraints to reduce the TAT substantially. Of course, when the power bound is removed completely, the TAT is at its lowest. However, doing so appears to create an invalid schedule most of the time, even though a schedule with the same TAT can be achieved without violating the power bound, which implies enforcing a power bound will not always require increasing TAT. The normalized TAT results of Table II are also shown in Figure 1 to more clearly show the impact of the different scheduling methods, with violating no power bound schedules also being indicated.

Figure 2 gives the TAT result of 2 f2126 benchmarks stacked together (with combined power bound) as the number of data-carrying TAM pins and TSVs are increased. The figure gives the number TAM pins allowed on the x-axis, while the number of TSVs allowed is given on separate plots. The figure shows that the decrease in TAT becomes less as more TAM pins are allowed for scheduling. This trend has been observed for single tests in [9], but to the author's knowledge this has never been observed for an entire SoC. However, the graph also shows an interesting relationship between the TAM pin bound and the TSV bound, most notably that they limit each other on the quality of the schedule. For instance, when only a single TSV is allowed for scheduling, allowing for more TAM pins will

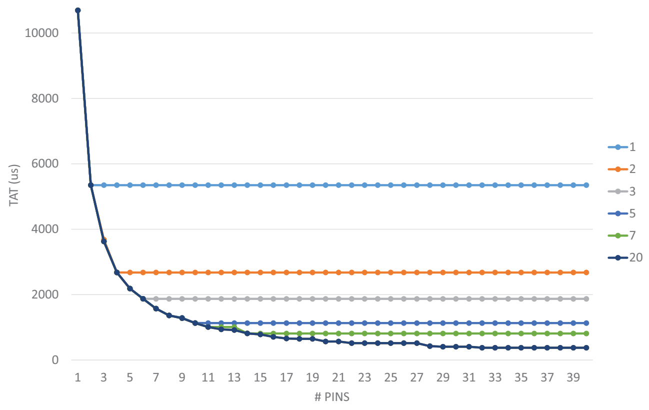

Fig. 2. TAT result of two stacked f2126 benchmarks as more pins and TSVs are allowed for scheduling.

cannot improve the schedule quality. The inverse is also true, since allowing extra TSVs gives no decrease in TAT when few TAM pins are available. One can conclude two things from this plot: first, although increasing the available TAM pins and TSVs can give a better schedule, the investment is worth much more for fewer pins and TSVs, and second, when scheduling tests for 3D-ICs, one must balance the available TAM pins and TSVs, as these hardware resources are costly and should not go to waste.

VI. EXTENSIONS FOR OTHER CONSTRAINTS

Although the formulation presented here has given a method for scheduling under several different constraints, other constraints may be desired depending on the scheduling environment. Some scheduling environments can have different constraints due to the hardware available for test scheduling or due to the nature of the tests to be applied. For this reason, a number of extensions to the formulation are given below to cope with other testing environments.

It is not uncommon for pre-defined hardware constraints to be given to the test scheduler, which can come from either test-specific hardware (e.g., BIST) or non-test-specific hardware (e.g., memory). If such constraints are required to be enforced, extra constants "Γ_{t_1,t_2}" can be provided to the scheduler.

$$\Gamma_{t_1,t_2} = \begin{cases} 1 & \text{if the test } t_1 \text{ is incompatible} \\ & \text{with the test } t_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1, t_2 \in T : \Omega(t_1, t_2) \leq 1 - \Gamma_{t_1,t_2}$$

Some schedules may require individual tests to be executed at one or more specific voltage/frequency values. The reason for this is that some faults are voltage/frequency dependent and can only be detected at such voltage/frequency values. Forcing the execution of a test t at a specific voltage/frequency value n by setting $\sum_{\forall c \in C_t} \psi(t, n, c) = 1$ for the required t and n. This modification may also require a change of the right-hand side of Equation (1) to reflect how many times a test must be executed.

Many designs have voltage/frequency hardware shared between tests, which implies that if two tests that share such hardware are executing at the same time they must also be executing at the same voltage/frequency. If this is the case for two tests t_1 and t_2, a new constraint given below can be enforced.

$$\exists t_1, t_2 \in T, \forall n \in N \sum_{\forall c \in C_t} \psi(t_1, n, c) \geq \sum_{\forall c \in C_t} \psi(t_2, n, c) - (1 - \Omega(t_1, t_2))$$

In some scheduling problems, the issue at hand is not necessarily to minimize the TAT of a device, but instead to reduce the overall cost of testing a device, since TAM pins and testing TSVs can be a large cost burden on a designer. Since the total TSVs ($Total_{TSV} = \sum_{\forall d \in D} V(d)$) and total number of TAM pins ($\forall t_1 \in T : Total_{TAM} = \sum_{\forall t_2 \in T} C(t_1, t_2)$) are known, the cost to minimize can be easily formulated given the cost of TAT (α), TAM pins (β), and testing TSVs (γ) is available to the scheduler.

$$cost = \alpha * t_{finish} + \beta * Total_{TAM} + \gamma * Total_{TSV}$$

VII. Conclusion

This paper has presented an SoC test scheduling formulation which incorporates several test scheduling constraints concurrently. By doing so, the formulation allows for the scheduling of tests under environments which previously required assumptions which lead to less than optimal test schedules. Incorporating several constraints at once will not only lead to smaller testing costs by producing shorter test schedules, but will also allow for more valid test schedules under modern design constraints.

Although the presented formulation is an advancement in its ability to incorporate modern testing constraints, future work can still focus on further reducing the TAT of schedules by more accurately modelling testing constraints. For instance, allowing voltage and frequency to be scaled independently and continuously, as opposed to forcing voltage and frequency to be discrete values, has the opportunity to allow for more overlaps of tests under power constraints. Also, it is desirable to more accurately model the power of tests, as opposed to presuming the power of a test be constant throughout its execution. By removing these and other presumptions from the test scheduling process, the TAT of test schedules can be further reduced, but this is left for future work.

References

[1] M. Taouil, S. Hamdioui, K. Beenakker, and E. J. Marinissen, "Test Impact on the Overall Die-to-Wafer 3D Stacked IC Cost," *Journal of Electronic Testing*, vol. 28, pp. 15–25, Dec. 2011.

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

[3] 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.

[4] B. Sen Gupta, U. Ingelsson, and E. Larsson, "Scheduling Tests for 3D Stacked Chips under Power Constraints," in *2011 Sixth IEEE International Symposium on Electronic Design, Test and Application*, pp. 72–77, IEEE, Jan. 2011.

[5] V. Sheshardi, V. D. Agrawal, and P. Agrawal, "Optimal Power-Constrained SoC Test Schedules With Customizable Clock Rates," in *IEEE International SOC Conference (SOCC)*, (San Jose, CA), pp. 271–276, Oct. 2012.

[6] S. K. Millican and K. K. Saluja, "Formulating Optimal Test Scheduling Problem with Dynamic Voltage and Frequency Scaling," in *22nd AsianTest Symposium (ATS)*, pp. 165–170, IEEE, Nov. 2013.

[7] V. Sheshardi, V. D. Agrawal, and P. Agrawal, "Optimum Test Schedule for SoC with Specified Clock Frequencies and Supply Voltages," in *26th International Conference on VLSI Design and International Conference on Embedded Systems*, (Pune, India), pp. 267 – 272, Jan. 2013.

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

[9] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," *IEEE Transactions on Computers*, vol. 52, pp. 1619–1631, Dec. 2003.

[10] E. J. Marinissen, C.-C. Chi, J. Verbree, and M. Konijnenburg, "3D DfT architecture for pre-bond and post-bond testing," in *2010 IEEE International 3D Systems Integration Conference (3DIC)*, pp. 1–8, IEEE, Nov. 2010.

[11] R. M. 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, pp. 175–185, June 1997.

[12] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji, "Test Schedule Optimization for Multicore SoCs: Handling Dynamic Voltage Scaling and Multiple Voltage Islands," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, pp. 1754–1766, Nov. 2012.

[13] E. Larsson and H. Fujiwara, "Power Constrained Preemptive TAM Scheduling," in *Proceedings of the 7th IEEE European Test Workshop*, pp. 119–126, IEEE Comput. Soc, 2002.

[14] 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 (ICCAD)*, pp. 59–66, IEEE, Nov. 2008.

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