2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems

Thermal-Safe Schedule Generation For System-on-Chip Testing

Rajit Karmakar and Santanu Chattopadhyay
Dept. of Electronics & Electrical Comm. Engineering
Indian Institute of Technology Kharagpur, India, Kharagpur, 721302
Email: {rajit,santanu}@ece.iitkgp.ernet.in

***Abstract*—This paper presents a thermal safe test scheduling strategy for System-on-Chip (SoC). While most of the existing strategies rely on some approximate thermal models to avoid the time consuming online thermal simulations, the present work proposes to use a superposition principle-based thermal model, which can estimate the temperature of the cores quite accurately, yet fast, without invoking thermal simulation inside the schedule generation process. The thermal model, along with a window-based peak power model, has been incorporated into a Particle Swarm Optimization (PSO) based meta search technique to generate the test schedules. In contrast to the existing works, the introduction of new SoC benchmarks with detailed information regarding power and floorplan enables us to observe exact thermal behaviors of the cores. Experimental results on these newly proposed benchmarks show the superiority of our thermal model over the existing ones.**

Keywords*—System-on-Chip; Test scheduling; Superposition principle based thermal model; Particle Swarm Optimization; Bin packing.

I. Introduction

With the increasing demand for high performance and low-power chips, present day's semiconductor industry is heading towards smaller feature sizes and reduced chip area. Device dimensions are reducing drastically, while the system designs are becoming more and more complex. The problems related to power and thermal issues are becoming more prominent in the System-on-Chip (SoC) designs. Testing of such a complex chip has become a major challenge for the test engineers. The test mode power is often 30 times higher than the functional mode. Not only the high power consumption, but also the high peak temperature during testing is causing serious threat to the chip. Due to the non-uniformity in the spatial power distribution, the temperature may not be equal throughout the chip. High power density of a particular core may create localized heating, called hotspots [1]. These hotspots may lead to decrease in the reliability of the circuit and even permanent damage of the chip, due to thermal runaway [2]. A test engineer has to pay special attention towards the power and thermal safety, at the time of the development of test infrastructure of the SoC.

On the other hand, shrinking product development cycle requires to reduce the test time of the SoC. This can be achieved via a proper scheduling of the tests for the cores. Development of test infrastructure and schedule of a SoC under resource, power and thermal constraints can be described as follows. A test engineer has to (i) partition the available test resources and allocate to the cores and (ii) decide upon the core ordering in the test schedule, with an objective to reduce the overall Test Application Time (TAT). Moreover, at any point of time in the schedule, the total power consumed by all the cores tested in parallel, must not exceed a certain pre-defined system level power limit and the peak temperature of any core must not violate the maximum allowable temperature limit.

To ensure the thermal-safety during testing, the temperature of the cores, in the scheduling interval needs to be computed, which requires online thermal simulation (i.e. at the time of schedule generation process). However, one major drawback of the thermal simulators like HotSpot [3] is their execution time, which restricts us to invoke thermal simulators inside any meta-search technique, that are often used to find the optimal test schedules for the SoCs, with large number of embedded cores. An alternative solution is to incorporate a thermal model, which can predict the temperature of the cores, without integrated thermal simulations. Several such approaches [4]–[10] have been proposed in the literature. The RC model based approach presented in [9], has tried to maximize the heat dissipation through the lateral neighbourhood of the active cores, in a test session. However, the concept of thermal ground of the idle cores and negligible heat transfer between the neighbouring cores does not hold, as we have reported later in this paper, the temperature of a core largely depends on the neighbouring cores, tested in parallel. Moreover, one common problem with all these thermal models is, due to lots of assumptions about the important parameters like power, floorplan etc, these thermal models may not always predict the temperature accurately. The exact thermal behaviour of a chip requires the exact power profiles of the cores as well as the accurate area and floorplan information of the chip. In the absence of all these information of commonly used ITC'02 benchmarks, most of the work presumes some approximate values for these parameters, which introduce inaccuracy in the thermal behavior of the chip.

Thermal simulator like Hotspot follows linear RC thermal model [8]. The linearity of the thermal model can be exploited using the superposition principle. The work presented in [8], tried to exploit the linearity of the Hotspot tool and used a superposition principle based thermal model. As the CPU execution time is the main bottleneck of the thermal simulations during scheduling, the authors have tried to avoid invoking the thermal simulator in the scheduling process. Instead, they have used the HotSpot [3] tool to create offline thermal profiles of the cores. These thermal profiles are used for the scheduling purpose. This type of thermal model is fast. However, while working with this thermal model [8], we have noticed that, it may result in thermal violations. This, we believe, because of neglecting the pre-schedule temperature increase of cores due to the leakage power and also inefficient modelling of the heating and cooling effects of the cores, which we have discussed elaborately in Section II.

To alleviate the inaccuracy of the thermal model of [8], in this paper, we have used a more elaborate thermal model, based on superposition principle. It uses relatively more detailed and accurate thermal information to get an efficient, yet fast

978-1-4673-8700-2/16 $31.00 © 2016 IEEE
DOI 10.1109/VLSID.2016.47

thermal-safe test schedule. Further, the work [8] do not perform Test Access Mechanism (TAM) wire allocation to the cores. Our work integrates TAM resource allocation with thermal-aware test schedule generation. The integration of this fast and accurate thermal model also aid us in designing a Particle Swarm Optimization (PSO) [11] based meta-search procedure to evolve better test schedule, exploring a large search space of solutions. To observe the exact thermal behavior of the cores, we have introduced a new set of SoCs, considering the ISCAS'85, ISCAS'89 and ITC'99 benchmarks as cores, with detail information regarding test vectors, area, floorplan, dynamic and leakage powers. Experimental results on the newly formed SoCs show that our model estimates temperature better than [8], hence assures thermal safety of the test schedule more efficiently.

Rest of the paper is organised as follows. Section II presents the proposed superposition principle based thermal model. The basic requirements to develop test infrastructure and the test scheduling strategy is described in Section III. Section IV presents a new set of SoC benchmarks and also the experimental results on the proposed benchmarks. Finally Section V draws the conclusion of the paper.

II. Superposition Principle Based Thermal Model

In this section, we have proposed a superposition principle based thermal model. It takes the help of the linearity of the thermal model of HotSpot and uses offline HotSpot [3] thermal simulations for each core in different possible conditions and creates thermal databases for all those conditions. These database information are used for the scheduling purpose. It helps us to generate a thermal safe test schedule much faster than the techniques that use online Hotspot simulations. To get the exact thermal behavior of a core, it is very much important to observe the different conditions, which can change the temperature of a core. Figure 1 summarises different possible conditions of temperature dependency between cores.

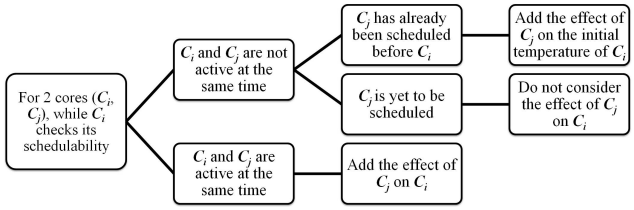

Figure 1. Different possibilities of temperature effect of core C_j on core C_i

The temperature of a core C_i can increase due to the following activities.

- Before C_i starts its testing, its initial temperature increases due to its leakage power consumption (Figure 2(a)).
- During the period of testing of the core C_i, its temperature increases (Figure 2(b)).
- If any numbers of other cores $C_j(1 \leq j \leq N)(j \neq i)$ are tested in parallel with C_i, due to the lateral spreading of heat from C_j, the temperature of C_i increases (Figure 2(b)).
- Initial temperature of C_i increases due to the effect of all the cores which have already been scheduled before C_i starts its testing (Figure 2(c)).

Figure 2 shows some simulation results on SoC $k10$ (details of the SoC has been mentioned in Section IV-A) to depict the exact thermal behavior of core 1 in different conditions. Figure 2(a) shows the increase in the initial temperature of core 1 due to the leakage power consumption before it starts its testing in SoC $k10$. Figure 2(b) shows the extra heating effect on core 1, if it is tested in parallel with core 2. Figure 2(c) shows the effect of the already scheduled cores on the initial temperature of a core chosen to be scheduled next. When core 1 is idle and core 2 is being tested, because of lateral heat spreading from core 2, the initial temperature of core 1 increases by 10°C. If core 1 starts its test just after core 2 has finished, the initial temperature of core 1 should be 10°C more than its normal initial temperature. However, if core 1 starts its test after a certain time interval, in that time gap, its initial temperature reduces towards its normal initial temperature. Depending upon the scheduling position of any core, the thermal model should be able to handle all these conditions and generate a test schedule that does not violate the thermal constraint. The objective of the superposition principle based thermal model is to sum up the individual effects of other cores on the temperature of a particular core, based on its ordering in the test schedule. The operation of our superposition principle based thermal model can be described by the following steps.

Step I: For each core $C_i(1 \leq I \leq N)$, calculate the increase in the initial temperature, due to leakage power consumption. Create a thermal database of it.

Step II: For each core $C_i(1 \leq I \leq N)$, calculate its temperature rise, when it is tested alone. Then, calculate the extra increase in the temperature of C_i, due to the core $C_j(1 \leq j \leq N)(I \neq j)$, when C_j tests in parallel with C_i. This extra increase in the temperature of C_i is calculated for each C_j separately. Create a thermal database of it. Superposition principle is applied to calculate the actual temperature increase of core C_i, considering the cores, tested in parallel with C_i.

Step III: For each core $C_i(1 \leq i \leq N)$, calculate its initial temperature increase, due to the core $C_j(1 \leq j \leq N)(i \neq j)$, which is scheduled prior to C_i. Let us assume, the initial temperature increase is T_1. Calculate the cooling rate for the combination of C_i and C_j, from the slope of the temperature decrease curve, as mentioned in Figure 2(c). Although the temperature decrease portion of the curve is not exactly linear, still a linear prediction of this portion of the curve can be a good approximation to calculate the simplified cooling rate. If the time gap is Δt and the temperature decrease in Δt time is ΔT, then, cooling rate can be calculated as

$$Cooling\ rate = \frac{\Delta T}{\Delta t} \tag{1}$$

The net increase in the initial temperature of C_i is $(T_1 - \Delta T)$. This is carried out for each C_j, corresponding to each C_i. Two thermal databases store all these information. Superposition principle is applied to calculate the actual increase in the temperature of C_i, considering all such C_j.

The superposition principle based thermal model, proposed in [8], neglects the scenario of the pre-schedule increase in the initial temperature of a core, because of leakage power consumption. Moreover, to take care of the heating effect of the pre-scheduled cores and to introduce a cool down period after a test completion, the work [8] proposes to make the

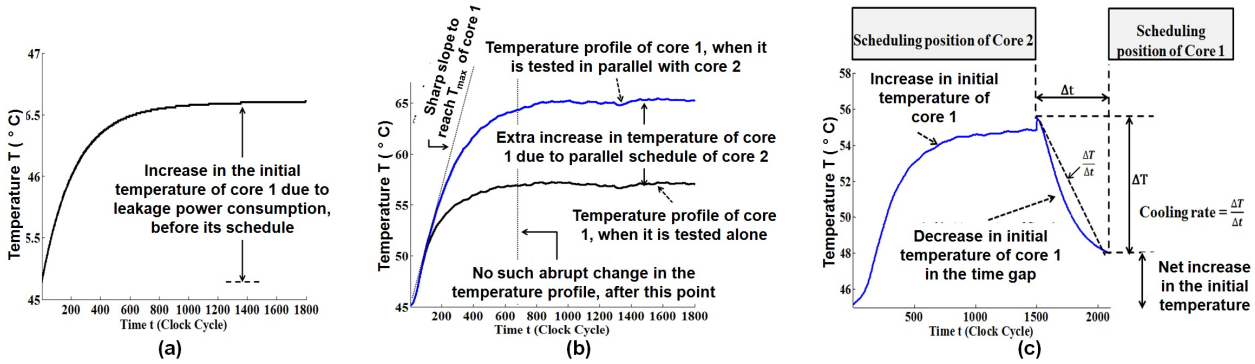

Figure 2. (a) Pre-scheduling increase of the initial temperature of core 1 of SoC $k10$ due to leakage power, (b) Temperature profile of core 1 during its scheduling, (c) Temperature profile of core 1 as an effect of core 2 schedules before core 1

power profiles of the finished cores to zero, for the rest of the schedule duration. However, the thermal simulation of each core is carried out offline and its scheduling position is unknown, until all the cores have been scheduled. Apparently, it is not clear from [8], that, how long one should carry on the offline thermal simulation for each core. As the waiting time of a core before it starts its testing is unknown and can only be determined by the scheduling process, the total test time of all the cores cannot be predicted beforehand. This makes the thermal model more complex. On the other hand, our thermal model handles the pre-schedule heating effects of a core more efficiently, hence estimates the temperature more accurately.

III. Test Infrastructure Development and Test Scheduling

Problem Formulation

Suppose a SoC with N cores $C_1, C_2 \ldots C_N$ is to be tested with a maximum of W_{max} TAM resources, a maximum power limit P_{max}, and a maximum temperature limit T_{max}. The test scheduling problem is to allocate TAM resources and test times to the cores so that, the total test application time (TAT) is minimized, while the power consumption during testing remains under P_{max} and the maximum temperature of any core does not cross the temperature upper bound of T_{max}.

A. *Test Infrastructure Development*

The basic requirements of test infrastructure development of a SoC under resource, power and thermal constraints are to consider a power model to estimate the test mode power of the cores and a thermal model to estimate the temperature of the cores during testing. Power profile of each core is required to check power validation at each point of the schedule, while selecting test parallelism between the cores. Calculation of the temperature of the cores also requires the power profile information of the cores, while a thermal model is used to avoid the time consuming online thermal simulations.

- ***Power Profile Generation***:

We have used a window-based peak power model [12] to estimate the test mode power of the cores. For this purpose, we have first calculated the cycle-accurate power profile of each core. Cycle-accurate power profile of a core can be estimated using the information of the transition counts in the wrapper chains in every clock cycle and the dynamic power consumption of the core. Average transition (ATR) is calculated for the total test time of the core. Dynamic power (DP) consumption of a core can be obtained using Synopsys Design Vision Report Power Compiler tool [13] as mentioned in the Table I in Section IV-A. If the transition in i^{th} cycle is TR_i, the cycle-accurate power of the i^{th} cycle (CAP_i) can be estimated as

$$CAP_i = \frac{TR_i}{ATR} \times DP \quad (2)$$

From this cycle-accurate power profile, we calculate the window-based peak power profile for each core by partitioning the total test time of the core into some smaller sized time windows and considering a single peak power in that interval to represent the power value for that interval. A suitable window-size results in an efficient balance between the scheduling complexity of the cycle-accurate power model [14] and the power-overestimation of the global peak power model [15]. These window-based power profiles of all the cores are used in the next steps of power validity and estimation of thermal profiles.

- ***Thermal database creation***:

After generating the power profile of each core, we create offline thermal databases from these power profiles and the floorplan of the chip, using offline Hotspot thermal simulations. Four such databases are created, considering different thermal conditions between the cores. $Thermal\_Database\_1(TD1)$ stores the values of the initial temperature rise, due to the leakage power consumption of each core. $Thermal\_Database\_2(TD2)$ stores the temperature values of self testing as well as parallel testing. The increase in the initial temperature of a core, because of the previously scheduled cores, are stored using two thermal databases, $Thermal\_Database\_3(TD3)$ and $Thermal\_Database\_4(TD4)$. The procedures of thermal database generations are mentioned next in $Procedure\_TD1$, $Procedure\_TD2$, $Procedure\_TD3$ and $Procedure\_TD4$.

- ***Procedure_TD1 (Thermal_Database_1 (TD1))***:
  1) For all the cores $C_i(1 \leq i \leq N)$, create power profile $P_{leakage_i}$, by considering leakage power of C_i and zero power value of other cores.
  2) Calculate the maximum temperature, $T_{leakage_i}$ from the transient and steady state responses of C_i and update $TD1$.

- ***Procedure_TD2 (Thermal_Database_2 (TD2))***:

1) For all the cores $C_i(1 \leq i \leq N)$, create power profile P_{self_i}, by considering dynamic power of C_i for the clock cycles to test C_i and leakage power values of other cores.
2) Calculate the maximum temperature, T_{self_i} from the transient and steady state responses of C_i and update $TD2$.
3) For each core $C_j(1 \leq j \leq N)(j \neq i)$, create power profile $P_{parallel_{ij}}$, by considering the dynamic power of C_i and C_j for the clock cycles required to test C_i and leakage powers for all other cores $C_k(k \neq j \neq i)$.
4) Calculate the maximum temperature, $T_{parallel_{ij}}$, from the transient and steady state responses of C_i.
5) $T_{extra_{ij}} = T_{parallel_{ij}} - T_{self_i}$ and update $T_{extra_{ij}}$ in $TD2$.

- ***Procedure_TD3 (Thermal_Database_3 (TD3))***:
  1) For all the cores $C_i(1 \leq i \leq N)$, corresponding to the core $C_j(1 \leq j \leq N)(j \neq i)$, create pre-schedule power profile $P_{preschedule1_{ij}}$, by considering dynamic power of C_j and leakage powers for all other cores $C_k(k \neq j)$, including C_i for the clock cycles required to test C_j.
  2) Calculate the maximum temperature, $T_{preschedule1_{ij}}$ of C_i, from the transient and steady state responses, update $TD3$.

- ***Procedure_TD4 (Thermal_Database_4 (TD4))***:
  1) For all the cores $C_i(1 \leq i \leq N)$, corresponding to the core $C_j(1 \leq j \leq N)(j \neq i)$, create pre-schedule power profile $P_{preschedule2_{ij}}$, by considering dynamic power of C_j for the clock cycles required to test C_j, leakage powers of C_j for next 2000 number of clock cycles and leakage power for all other cores $C_k(k \neq j)$, including C_i for the entire time period.
  2) Calculate the final temperature, $T_{preschedule2_{ij}}$ of C_i, from the transient and steady state responses of C_i, update $TD4$.

In $Procedure\_TD1$, $T_{leakage_i}$ is the maximum increase in the initial temperature of C_i, due to leakage power consumption. In $Procedure\_TD2$, T_{self_i} is the temperature increase of core C_i, when no other cores are tested in parallel with C_i. $T_{parallel_{ij}}$ is the maximum increase in temperature of C_i, when it is tested in parallel with C_j. $T_{extra_{ij}}$ is the extra increase in the temperature of C_i, as an effect of parallel testing of C_j with C_i. In $Procedure\_TD3$, $T_{preschedule1_{ij}}$ calculates the maximum increase in the initial temperature of C_i, as an effect of the earlier scheduling of C_j. In $Procedure\_TD4$, we allow C_i to cool down towards its normal initial temperature for 2000 clock cycles. $T_{preschedule2_{ij}}$ is the initial temperature of C_i after 2000 clock cycles. The cooling rate of C_i for the combination of C_i and C_j can be calculated as

$$Cooling\ Rate(CR_{ij}) = \frac{(T_{preschedule1_{ij}} - T_{preschedule2_{ij}})}{2000} \quad (3)$$

We can calculate the actual reduction in the initial temperature of C_i, by multiplying CR_{ij} with the time gap (TG) between the end time of C_j and start time of C_i. The net increase in the initial temperature of C_i, because of the earlier scheduling of C_j, can be calculated as

$$Net\ Initial\ Increase = T_{preschedule1_{ij}} - (CR_{ij} \times TG) \quad (4)$$

B. *Particle Swarm Optimization Guided Test Scheduling*

In this step, we generate a valid power and thermal-safe test schedule. Rectangular bin packing approach is used to partition and allocate test resources to the cores and to decide the core order in the test schedule. Each core $C_i(1 \leq i \leq N)$ is represented by a set of wrapper configurations R_i. The test resource requirement of core C_i with j^{th} wrapper configuration can be represented by a rectangle whose height and width represent allocated TAM width (w_{ij}) and the corresponding test time ($T(w_{ij})$) respectively. To get a schedule for the full SoC, the rectangles are to be packed into a bin of fixed height (W_{max}), so that the TAT (width of the bin) is minimized. As the bin packing problem is NP-Hard [16], we have utilized a Particle Swarm Optimization based meta search technique to solve the scheduling problem. In PSO, each particle corresponds to a solution to the optimization problem being solved. Any PSO formulation involves choosing a proper representation of the particles, their fitness calculation, and defining an evolution policy. Let, the number of cores in the SoC be N and the maximum number of rectangles for any core is M. Let $B = \lceil \log_2 M \rceil$. A particle consists of $N \times B$ number of bits. First B bits identify the test rectangle selected for the first core, second B bits for the second core, and so on. Figure 3 shows a simple particle with N = 4 and B = 4. In this case, test rectangles 9, 2, 8 and 13 are selected for cores 1, 2, 3 and 4 respectively. Particles evolve over the generations guided by self ($lbest$) and group-intelligence ($gbest$), and also via their inertia. A $replace$ operator attempts to align a particle with its $pbest$ and the $gbest$ particles, with some probability. For bit position i of a particle, the bit is replaced by the corresponding bit of $pbest$ and $gbest$ particles with probability α and β respectively to evolve a new particle. Fitness of a particle is equal to the total test time (TAT) of the SoC after scheduling the test rectangles using the algorithm $Schedule\_Rectangles$.

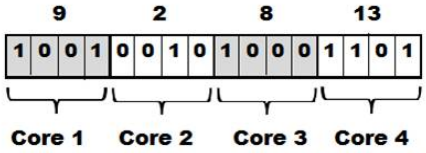

Figure 3. Sample particle structure

The $Schedule\_Rectangles$ algorithm takes as input the rectangle set corresponding to the particle, the maximum TAM width W_{max}, the maximum power limit P_{max} and the maximum allowable temperature T_{max}. It performs a scheduling of the rectangles, honouring the constraints that at no instant of time, the total TAM width requirement exceeds W_{max}, and the instantaneous power value does not exceed P_{max}. Also the maximum temperature of individual cores does not exceed T_{max}. A $Power\_Violation\_Checker(PVC)$ checks whether there is any power violation during scheduling. If at any particular point in the schedule, a core does not respect PVC, it does not get the permission to get scheduled at that point. Similarly, a $Thermal\_Violation\_Checker(TVC)$ checks whether any core is violating thermal budget or not. TVC checks the scheduling position of a core and uses the superposition principle to add different thermal values from $TD1$, $TD2$, $TD3$ and $TD4$, corresponding to that core to calculate the exact temperature of the core. BP, ATW and PT keep track of the scheduling points, corresponding resource availability at those points and the power values at BPs respectively, while TT notes the temperature of each core. As the till unscheduled cores get scheduled, the list BP, ATW, PT and TT also get updated. The bin packing procedure also needs to prioritize the next unscheduled rectangle to be selected for packing (scheduling). For this purpose the rectangles are sorted on their area values (TAM width(w) $\times$ test time(T)) in a descending order. The break-point list BP is scanned from the minimum to the maximum value. When the rectangles corresponding to

all the cores have been scheduled, the maximum end time of the testing of all the cores gives the total test application time. The algorithm to produce the schedule is presented next.

input : $LIST_C$, W_{max}, P_{max}, T_{max}
output: A schedule of all the cores respecting all the constraints with minimum TAT
begin
  while (all cores are not scheduled) **do**
    $bp_k \leftarrow$ Break point with minimum time value;
    $atw_k \leftarrow$ Available tam resource at bp_k;
    while $(atw_k > 0)$ **do**
      Set $Area_{max}$ = 0;
      forall the unscheduled cores C_i **do**
        if $w_{ij} \leq atw_k$ **then**
          if P_{ij} respects PVC **then**
            if C_i and all earlier scheduled cores respect TVC **then**
              Calculate $Area_{ij} = w_{ij} \times T(w_{ij})$;
          if $Area_{ij} > Area_{max}$ **then**
            Set $Area_{max} = Area_{ij}$;
            Set $C_{imax} = C_i$;
      Select C_{imax} for scheduling at bp_k;
      Update BP, ATW, PT, TT;
      $Start_{C_i} = bp_k$; $End_{C_i} = bp_k + T(w_{ij})$;
      Mark C_i as scheduled;
    Remove bp_k and atw_k from BP and ATW respectively;

Algorithm 1: ***Schedule_Rectangles***

IV. Experimental Results and Discussions

A. Experimental Setup

Fewer details of the traditional ITC'02 SoC benchmarks have motivated us to form new SoCs with detail information to observe the exact thermal behaviour of them. We have developed two SoCs, named $k10$ and $k25$ having 10 and 25 cores respectively, using ITC'99, ISCAS 89 and ISCAS 85 circuits. From the benchmark suites mentioned above, a number of cores have been designed as follows.

- Each circuit description in $Verilog$ format (.v) has been taken as input and mapped to the Faraday 90nm standard cell library [17]. The circuits are synthesized using the $Synopsys\ Design\ Vision$ Compiler [13] to generate gate-level netlist from the $Verilog$ design description.
- All the flip-flops are replaced by scan flip-flops using the $Synopsys\ DFT$ Compiler for the testability purpose.
- Multiple scan chains are inserted to reduce the TAT.
- Dynamic and leakage powers are extracted from the $Synopsys\ Design\ Vision\ Report\ Power$ tool [13].
- Area of the cores are calculated using the $Cadence\ Encounter$ tool [18].
- Exact floorplan of the SoCs are calculated using the Floorplanning tool $Blobb$ [19], [20] and $Plottool$. The empty spaces in the floorplan are packed with the zero power boxes to make it compatible with the HotSpot tool. The packing of the empty spaces with zero power boxes are done using a C program.
- Test patterns are generated using the Synopsys $TetraMax\ ATPG$ tool [21].

Table I reports the value of the different parameters of different circuits, that we have obtained using the above mentioned tools. The components of two newly formed SoCs $k10$ and $k25$ are also mentioned in this table.

B. Experimental Results

1) ***Thermal-Aware Test Scheduling****:* In this section, we present the results of our experimentation under resources, power and thermal constraints, for different SoCs. Superposition principle based thermal model, along with the window-based peak power model, has been considered to guide PSO based power and thermal-aware test scheduling strategy. Table II shows the results on two newly formed SoCs $k10$ and $k25$. We have shown the simulation results for W_{max} = 64 and 56, for both the SoCs and varied the P_{max} and T_{max} values. It may be observed that, if we increase the P_{max} and T_{max} values, TAT (in clock cycles) gets reduced. Also, for a particular P_{max} value, if we increase T_{max} value, TAT gradually reduces. This happens as with the relaxation of power and thermal constraints, TAT can be minimized further. It may be noted that in some cases increase in the T_{max} value does not really help to reduce the TAT value further. It is because of the fact that the SoC has already reached its saturation TAT in that temperature range. Variation in the temperature does not have any impact on the TAT, in that temperature range. However, direct comparison of our work with other related thermal-aware test scheduling works is not possible, as they assume their own power profiles and different floorplans.

2) ***Comparison Between Thermal Models****:* In this part, we focus on the comparison between our proposed thermal model and the thermal model proposed in [8]. Our main objective in

Table I. Information of the Cores of Newly Formed SoCs

Circuit name	DP (mW)	LP (uW)	No. Test Vector	No. Scan Chain	Max Scan length	No of copies in SoC	
						$k10$	$k25$
$s38584$	9.15	315.57	146	32	45	2	2
$s38417$	4.60	311.45	100	32	55	2	0
$s35932$	10.27	282.44	64	32	54	0	2
$s15850$	3.05	128.62	131	16	34	1	0
$s13207$	1.91	116.48	273	16	41	0	1
$s9234$	1.23	71.08	147	4	54	1	2
$s5378$	0.72	35.09	124	4	46	0	2
$b22$	4.35	218.53	1501	46	15	0	1
$b21$	2.55	143.38	711	16	31	1	2
$b17$	3.67	346.39	1477	46	31	0	3
$b15$	1.18	120.60	457	16	29	2	2
$b14$	1.32	68.12	737	4	62	1	1
$c7552$	3.59	38.45	219	0	0	0	3
$c5315$	2.29	27.89	123	0	0	0	2
$c1908$	0.95	7.52	119	0	0	0	1
$c499$	0.01	1.12	52	0	0	0	1

Table II. Variation in Test Application Time (TAT) for Different SoCs with the Variation of P_{max} and T_{max}

$k10$	T_{max} (°C)	P_{max} = 1.3 Watt TAT	P_{max} = 1.5 Watt TAT	P_{max} = 2 Watt TAT	P_{max} = 5 Watt TAT
W_{max} = 64	72	43154	42494	42365	42365
	80	41424	41388	41347	40924
	90	39349	37875	37875	37875
	100	39349	35880	34810	34459
	110	39349	35732	34268	34021
W_{max} = 56	72	50285	50285	50285	49010
	80	43530	42424	42424	42424
	90	43530	39672	39672	39672
	100	42300	39672	39672	39672
	110	42300	39672	38936	38936
$k25$	T_{max} (°C)	P_{max} = 2.7 Watt TAT	P_{max} = 3 Watt TAT	P_{max} = 3.5 Watt TAT	P_{max} = 4.5 Watt TAT
W_{max} = 64	82	171142	169876	169869	169684
	90	170578	169562	169146	169101
	100	169513	169101	168982	168778
	110	168698	168602	168602	168602
W_{max} = 56	82	196567	196567	193613	192163
	90	195710	193202	192494	192050
	100	192494	191894	191276	191276
	110	191814	191644	191276	191276

comparison between the two thermal models is to find the minimum temperature budget at which the thermal models can produce test schedules and then checking the thermal validation of the obtained schedules, using the Hotspot thermal simulations. To check the quality of the two thermal models, we have tried to fit the thermal model proposed in [8] in our TAM constrained test scheduling strategy and then compare with our proposed thermal model.

Table III presents the minimum allowable temperature values to get a valid test schedule (T_{max}), for both the thermal models and their corresponding maximum temperature values obtained from the post-schedule Hotspot simulations (T_{out}). It may be noted that, using the thermal model proposed in [8], we are able to generate test schedules at $70°$C and $78°$C for the SoCs $k10$ and $k25$ respectively. However, while using our proposed thermal model, we can get test schedules at minimum temperature levels of $72°$C and $82°$C, for these two SoCs respectively. Our thermal model is unable to produce test schedules for any thermal budget lower than that. For example, when fed with $70°$C as T_{max} value, our tool does not produce any schedule for $k10$. It could work successfully only with a limit of $72°$C. The actual thermal simulations show this limit to be a bit pessimistic as the Hotspot results range between $70.66°$C and $71.92°$C. This, we believe, has occurred due to the simplified time overlapping model that has been used in our model. In our model, if there is a small overlap in the scheduled test times of two cores, it has been taken as a complete overlap of their test durations, which is probably not accurate for most of the cases. However, this pessimistic assumption could result in thermally valid test schedules.

Table III. COMPARISON OF VALIDATION OF TEST SCHEDULE OF OUR THERMAL MODEL WITH THE THERMAL MODEL PRESENTED IN [8]

SoC name	W_{max}		Thermal Model [8]			Our Thermal Model		
		P_{max} (Watt)	T_{max} ($°$C)	T_{out} ($°$C)	Violation	T_{max} ($°$C)	T_{out} ($°$C)	Violation
$k10$	32	1.3	70	70.74	**Yes**	72	70.74	No
		1.5	70	71.56	**Yes**	72	70.74	No
		2	70	70.74	**Yes**	72	70.74	No
		5	70	71.3	**Yes**	72	71.81	No
	48	1.3	70	70.79	**Yes**	72	70.66	No
		1.5	70	70.8	**Yes**	72	70.66	No
		2	70	70.86	**Yes**	72	70.66	No
		5	70	70.86	**Yes**	72	70.66	No
	64	1.3	70	71.2	**Yes**	72	70.66	No
		1.5	70	70.91	**Yes**	72	70.91	No
		2	70	71.32	**Yes**	72	71.1	No
		5	70	70.91	**Yes**	72	70.66	No
$k25$	32	2.7	78	77.94	No	82	78.27	No
		3	78	79.05	**Yes**	82	78.13	No
		3.5	78	79.88	**Yes**	82	79.2	No
		4.5	78	78.09	**Yes**	82	77.5	No
	48	2.7	78	78.14	**Yes**	82	78.72	No
		3	78	78.14	**Yes**	82	77.15	No
		3.5	78	77.8	No	82	77.52	No
		4.5	78	77.84	No	82	77.77	No
	64	2.7	78	76.77	No	82	77.12	No
		3	78	76.42	No	82	77.94	No
		3.5	78	76.72	No	82	77.34	No
		4.5	78	76.95	No	82	77.18	No

Although, the thermal model presented in [8] produces valid test schedules with a lower temperature limit than our thermal model, it does not always ensure thermal safe test schedule. It violates thermal limits in most of the cases. On the other hand, our thermal model violates the thermal limit for none of the cases. This is because our proposed thermal model takes care of the pre-scheduled heating effects of the cores more efficiently by considering both the temperature increase due to the leakage power and doing proper estimation of the heating and cooling effects of the already scheduled cores with more accuracy. This shows the efficiency of our thermal model.

V. CONCLUSION

In this paper, a new superposition principle based fast thermal model, which is capable of estimating temperature with accuracy, has been proposed. The thermal model along with a window-based peak power model, has been incorporated into a PSO based formulation to select the test rectangles for the cores and scheduling them using a 3-D bin-packing approach. A new set of SoC benchmarks with detail information has been used to get exact thermal effects of the cores at the time of generation of thermal-aware test schedule using the proposed method. The proposed approach can generate efficient test schedules without violating power and thermal constraints.

REFERENCES

[1] M. Cho and D. Z. Pan, "Peakaso: peak-temperature aware scan-vector optimization," in *IEEE VTS, 2006.*

[2] P. Girard, "Survey of low-power testing of vlsi circuits," *IEEE Design & test of computers*, vol. 19, no. 3, pp. 82–92, 2002.

[3] M. R. Stan, K. Skadron, M. Barcella, W. Huang, K. Sankaranarayanan, and S. Velusamy, "Hotspot: a dynamic compact thermal model at the processor-architecture level." *Microelectronics Journal*, vol. 34, no. 12, pp. 1153–1165, 2003.

[4] T. Yu, T. Yoneda, K. Chakrabarty, and H. Fujiwara, "Thermal-safe test access mechanism and wrapper co-optimization for system-on-chip," in *IEEE ATS., 2007. 16th*, pp. 187–192.

[5] D. Bild, S. Misra, T. Chantemy, P. Kumar, R. Dick, X. Huy, L. Shangz, and A. Choudhary, "Temperature-aware test scheduling for multiprocessor systems-on-chip," in *ICCAD 2008.*, pp. 59–66.

[6] T. Yu, T. Yoneda, K. Chakrabarty, and H. Fujiwara, "Test infrastructure design for core-based system-on-chip under cycle-accurate thermal constraints," in *ASP-DAC 2009.*, pp. 793–798.

[7] Z. He, Z. Peng, and P. Eles, "Thermal-aware test scheduling for core-based soc in an abort-on-first-fail test environment," in *Digital System Design, Architectures, Methods and Tools, 2009. 12th Euromicro Conference on*, pp. 239–246.

[8] C. Yao, K. Saluja, and P. Ramanathan, "Power and thermal constrained test scheduling under deep submicron technologies," *IEEE TCAD*, vol. 30, no. 2, pp. 317–322, 2011.

[9] P. Rosinger, B. Al-Hashimi, and K. Chakrabarty, "Thermal-safe test scheduling for core-based system-on-chip integrated circuits," *IEEE TCAD*, vol. 25, no. 11, pp. 2502–2512, 2006.

[10] Z. He, Z. Peng, and P. Eles, "A heuristic for thermal-safe soc test scheduling," in *ITC 2007.*, pp. 1–10.

[11] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, Proceedings., IEEE Int. Conf. on*, 1995, pp. 1942–1948 vol.4.

[12] C. S. Karmakar, Rajit, "Window-based peak power model and particle swarm optimization guided 3-dimensional bin packing for soc test scheduling," *Integration, the VLSI Journal*, vol. 50, pp. 61–73, 2015.

[13] *Design Vision "User Guide", Version 2002.05*, Synopsys Inc., 2002.

[14] S. Samii, M. Selkala, E. Larsson, K. Chakrabarty, and Z. Peng, "Cycle-accurate test power modeling and its application to soc test architecture design and scheduling," *IEEE TCAD*, vol. 27, no. 5, pp. 973–977, 2008.

[15] R. Chou, K. Saluja, and V. Agrawal, "Scheduling tests for vlsi systems under power constraints," *IEEE TVLSI*, vol. 5, no. 2, pp. 175–185, 1997.

[16] h. K. M. E. Iyengar, V., "On using rectangle packing for soc wrapper/tam co-optimization," in *VTS 2002*, pp. 253–258.

[17] "Faraday." January 2011. [Online]. Available: http://www.faraday.com.tw/AIP/ips/90library.html

[18] *Encounter. ,"Encounter user guide," year = 2008,*, Cadence Inc.

[19] H. H. Chan and I. L. Markov, "Practical slicing and non-slicing block-packing without simulated annealing," in *GLS-VLSI*, 2004, pp. 282–287.

[20] [Online]. Available: http://vlsicad.eecs.umich.edu/BK/BloBB/

[21] *TetraMax ATPG Guide*, Synopsys Inc., 2006.