Thermal-Aware Test Scheduling for Core-based SoC in an Abort-on-First-Fail Test Environment

Zhiyuan He, Zebo Peng, and Petru Eles
Department of Computer and Information Science
Linköping University
Linköping SE-58 183, Sweden
{zhihe, zebpe, petel}@ida.liu.se

***Abstract*—Long test application time and high temperature have become two major issues of system-on-chip (SoC) test. In order to minimize test application times and avoid overheating during tests, we propose a thermal-aware test scheduling technique for core-based SoC in an abort-on-first-fail (AOFF) test environment. The AOFF environment assumes that the test process is terminated as soon as the first fault is detected, which is usually deployed in volume production test. To avoid high temperature, test sets are partitioned into test sub-sequences which are separated by cooling periods. The proposed test scheduling technique utilizes instantaneous thermal simulation results to guide the partitioning of test sets and to determine the lengths of cooling periods. Experimental results have shown that the proposed technique is efficient to minimize the expected test application time while keeping the temperatures of cores under test below the imposed temperature limit.**

Keywords: system-on-chip test; test scheduling; thermal-aware test; abort-on-first-fail

I. INTRODUCTION [1]

In recent years, a core-based system-on-chip (SoC) approach has been employed for the design of very large scale integrated circuit systems, making the time-to-market shorter and the design cost lower. The core-based SoCs have pre-designed and pre-verified cores (modules) integrated into a single silicon die, and the testing of the system is naturally organized on a modular basis. Due to the complexity of the system, testing the core-based SoCs needs large volume of test data, leading to a substantially long test application time (TAT). Scheduling tests in parallel can reduce the TAT, while putting demands on the deployed test access mechanism (TAM). In a volume production environment, an abort-on-first-fail (AOFF) test approach [12], [14], [11], [7] is usually employed, in which the test process is terminated as soon as the first fault is detected. This means that testing SoCs in the AOFF environment can have significantly reduced TATs.

Another critical challenge for the SoC test is the high temperature occurred during test. It has been reported that the power density and the temperature are dramatically increased in the latest generation of ICs using deep-submicron technologies [1], [6], [15], [22]. As testing consumes more power than applications running in normal functional modes [16], [20], a higher temperature on a core under test (CUT) can be expected. When tests are applied to neighborhood cores in parallel, the CUTs encounter even higher temperatures. Therefore, rigid temperature control during test is required to avoid possible damages to the CUTs. Although advanced cooling mechanisms can help to solve the overheating problem, they substantially increase the system costs and usually require large space volumes. Other techniques such as lower frequency and reduced speed do help to avoid unexpectedly high temperature during tests, but they result in long test application times and are not applicable to at-speed tests.

Recently, low power test techniques [4], [18] are proposed to reduce the power consumption during tests. Power-constrained test scheduling approaches [3], [2], [13], [7] are proposed to minimize the TAT with a limited power package. Although the power-aware test techniques are efficient to solve the high power consumption problem, they cannot completely avoid the overheating problem because of the complex thermal phenomenon in modern electronic chips [17]. Therefore, thermal-aware test techniques [17], [24], [8] using direct temperature information are proposed to minimize the test application times while keeping the temperatures of CUTs below imposed temperature limits.

In this paper, we propose a thermal-aware test scheduling technique for modular SoC tests in the AOFF test environment. To the best of our knowledge, this is the first work that combines the thermal-aware test scheduling and modular SoC test using the AOFF approach. To estimate the TATs in volume production tests, we calculate the expected test application time (ETAT) [7] with given defect probabilities of individual cores. The core defect probabilities can be derived from statistical analysis of the production process or generated based on inductive fault analysis. We employ a thermal simulator, ISAC [23], to obtain instantaneous temperature values of the cores. The

[1] This work has been partially supported by the Swedish Foundation for Strategic Research (SSF) under the Strategic Integrated Electronic Systems Research (STRINGENT) program.

core temperatures are used to determine how to divide test sets into shorter test sub-sequences separated by cooling periods. An algorithm is developed to select cores for test according to their defect probabilities and a finite-state machine (FSM) is used to guide the partitioning and interleaving of test sets. A heuristic is proposed to generate thermal-safe test schedules with minimized ETATs.

The rest of this paper is organized as follows. The next section presents the assumed basic test architecture. In Section III, the background and motivation are demonstrated. Section IV gives the problem formulation, and Section V illustrates the proposed thermal-aware test scheduling technique. Experimental results are presented in Section VI and the paper is concluded in Section VII.

II. Basic Test Architecture

We assume that the tester employed for a SoC test is either an automatic test equipment (ATE) or an embedded tester in the chip. The tester consists of two major components, a test controller and a memory. The memory stores a test schedule and the generated test patterns. The test controller transports the test data to and from the CUTs in accordance with the test schedule. A test bus is employed for the test data transportation between the tester and the CUTs. Each core is connected to the test bus through dedicated TAM wires. Through the test bus and TAM wires, test patterns are sent to the CUTs and test responses are sent back to the tester.

III. Background and Motivation

A. Expected Test Application Time

In a volume production environment where diagnosis is not needed, a chip will be discarded as soon as a fault is detected. Therefore, the AOFF approach is usually used in production tests, leading to shorter TATs.

In order to solve the test time minimization problem for SoC tests using the AOFF test approach, the estimation of the TAT in production tests is critically important. In an AOFF test environment, the test process is terminated as soon as a fault is detected in any of the CUTs. This means that a possible test termination moment (PTTM) is the moment that a test response has been sent to the tester and the test result for a single test pattern has been revealed. We consider the termination of a SoC test process at a certain PTTM as a random event that occurs with a certain probability. Thus, the TAT in an AOFF test environment becomes a random variable. The mathematical expectation of the TAT, referred to as the expected test application time (ETAT), is used to estimate the actual test application times in volume production tests.

In [7], a method to calculate the ETAT was proposed. Suppose x is a PTTM, let A_x be the random event that the test process is aborted at PTTM x, and let T be the random event that the test process is passed till the completion. Thus, the ETAT is given by

$$ETAT = \sum_{\forall x \in X}(t_x \times p[A_x]) + L \times p[T] \qquad (1)$$

where x is a PTTM, X is the set of all PTTMs, t_x is the test application time by the moment x, L is the test application time when the entire test process is completed, $p[A_x]$ is the probability of the random event A_x, and $p[T]$ is the probability of the random event T.

From Equation (1), we can see that the TAT in an AOFF test environment depends on two factors, one is the elapsed time when the test process is terminated, and the other is the probability of the test process being terminated at a certain PTTM. In this paper, we assume the defect probabilities of individual cores are derived from statistical analysis of the production process or generated based on inductive fault analysis. Using the core defect probabilities and the incremental fault coverage calculated from fault simulation results, we compute the probability of the test terminated at every PTTM as presented in [7].

B. Test Set Partitioning and Interleaving

The temperatures of CUTs increase rapidly under high testing power consumptions. The temperatures may exceed a safe threshold, resulting in damages to the CUTs or eventually leading to a thermal runaway. The high degree of test parallelism as well as the long test application time makes the thermal issue more severe in the case of SoC test.

In order to avoid overheating in the AOFF test environment, an individual test has to be stopped when the temperature of the core reaches a temperature limit, denoted with *TL*, beyond which the core may be damaged. On the other hand, a cooling period is needed before the test is continued at a lower temperature level. In this paper, we refer to the cooling as passive cooling, meaning that the core is deactivated and does not consume dynamic power. Thus, by partitioning a single test set into a number of test sub-sequences and inserting cooling periods between them, we can avoid overheating during the entire test process.

Introducing cooling periods between test sub-sequences increases the TAT of that single test, though it helps to solve the overheating problem. As a core does not require transportation of test data during its cooling period, the surplus bandwidth of the test bus can be allocated to other cores for their test data transportations and test applications. In this way, the single tests for different cores are interleaved and, as a consequence, the long TAT due to introducing cooling periods is shortened.

C. Lateral Thermal Influence

As the technology scales, the area size of a silicon die decreases, while the die thickness decreases much slower. The mismatch of the decreasing rate in geometrical size at the horizontal and vertical dimensions causes the lateral heat flow taking a higher proportion in the overall heat flow. This makes it important to consider the thermal influences between adjacent cores, especially when concurrent tests are employed.

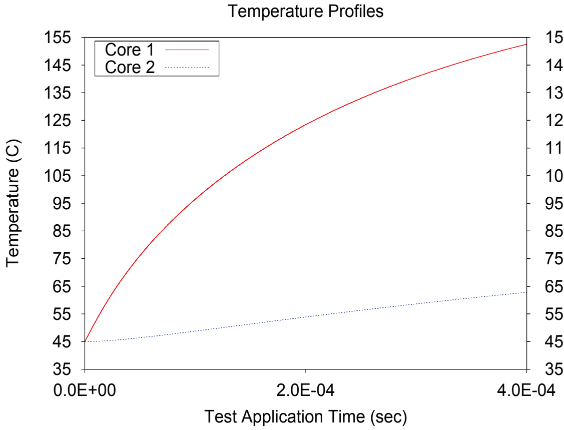

Figure 1. An example illustrating lateral thermal influence

Figure 1 depicts the result of thermal simulation performed for a SoC with a 200-micrometer die thickness. The SoC consists of two adjacent cores. In this experiment, core 1 is tested for a period of 400 microseconds while core 2 remains inactive. It can be seen that core 2 is passively heated by core 1 and the temperature of core 2 increases by 19 degrees. This experiment illustrates the scenario where the temperatures of inactive cores can be elevated by the active cores in the neighborhood. The degree of the temperature elevation on the inactive core depends on the floorplan, the number of hotter neighbors, and how long the tests last for the hotter neighbors. The result shows that the lateral thermal influence should be taken into account when generating a thermal-aware test schedule.

In [9], a thermal-simulation driven test scheduling technique was proposed for SoCs with lateral thermal influences. This technique aims to minimize the TAT under the temperature and test-bus bandwidth limits. However, directly applying the technique proposed in [9] to the AOFF tests is inefficient in reducing the ETAT (see experimental results in Section VI), since it does not consider the information regarding defect probabilities of cores. Therefore, in this paper, we propose a new technique for the thermal-aware test scheduling in the AOFF environment.

D. *Stop-Cooling Temperature*

Using test set partitioning and interleaving technique, the testing and cooling periods alternate for every core. The testing periods are interrupted at the temperature limit *TL*, while the cooling periods are stopped at a relatively low temperature level. In this paper, we referred to such a lower temperature level as the stop-cooling temperature and denote it with *CL*. The temperature curve of a core oscillates between *CL* and *TL*, and the gap has a large impact on the length of both the cooling periods and the test sub-sequences. Figure 2 illustrates a scenario where the test schedule for one of the cores in a SoC varies with respect to different *CL*s used for test scheduling.

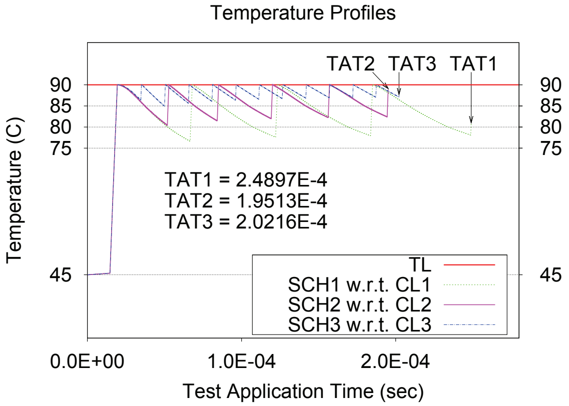

Figure 2. Alternative test schedules w.r.t. various CLs

In Figure 2, three different test schedules are depicted, which are denoted with *SCH1*, *SCH2*, and *SCH3*, respectively. The corresponding *CL*s and TATs are denoted with *CL1*, *CL2*, *CL3* and *TAT1*, *TAT2*, *TAT3*, respectively. Comparing the temperature profiles of *SCH2* and *SCH3*, we find out that *SCH3* uses a higher *CL* and has a longer TAT than *SCH2* does. The main reason why a higher *CL* can lead to a longer test schedule is the time overhead needed for the test controller to stop one test and start or resume another test [5], [7]. It can be seen that *SCH3* has shorter but more test sub-sequences than *SCH2* does, indicating that *SCH3* has a larger amount of time overhead.

On the other hand, when comparing the temperature profiles of *SCH1* and *SCH2*, we can see that *SCH1* uses a lower *CL* and also has a longer TAT than *SCH2* does. This means that a decreased *CL* may not lead to a shorter TAT, although the amount of time overhead is reduced due to a decrease in the number of test sub-sequences. This is because the temperature of a core decreases much more slowly at lower temperature levels, and therefore the cooling periods are much longer when a lower *CL* is used. If the increase in the length of cooling periods is larger than the decrease in the amount of time overhead, a longer TAT is expected. Thus, in order to generate efficient test schedules, we should use different *CL*s to explore various schemes of test set partitioning and interleaving.

In this paper, we aim to generate efficient and thermally safe test schedules for modular SoC tests in the AOFF environment. The problem is formulated as how to minimize the expected test application times under the temperature and the test-bus bandwidth limits. The exact problem formulation and the proposed solutions are demonstrated in the rest of the paper.

IV. PROBLEM FORMULATION

Suppose that a system-on-chip, denoted with *S*, consists of *n* cores, denoted with C_1, C_2, ... , C_n, respectively. Each of the cores has a defect probability, denoted with DP_1, DP_2, ... , DP_n, respectively. The cores are placed on the silicon die according to a floorplan, denoted with *FLP*, which also specifies the physical parameters of the silicon die and the package. In order to test core C_i ($1 \le i \le n$), l_i test patterns are generated, and the test set is denoted with TS_i. The test patterns/responses are transported through the test bus to/from core C_i, which requires a certain amount of test-bus bandwidth, denoted with W_i. The test bus is capable to concurrently transport test data for different cores under a bandwidth limit, denoted with *BL* ($BL \ge W_i$, $i = 1, 2, \dots, n$). A temperature limit, denoted with *TL*, is given. If the temperature of core C_i exceeds *TL*, the core may be damaged.

In order to avoid overheating a core under test, a test set needs to be partitioned into a number of shorter test sub-sequences and a cooling period needs to be inserted between two partitioned test sub-sequences. We address the problem as to minimize the expected test application time by generating an efficient test set partitioning scheme and test schedule for the SoC such that the following two constraints are satisfied: (1) the amount of test-bus bandwidth required for the concurrently applied tests is less than or equal to the bandwidth limit, and (2) the temperature of each core is kept below the temperature limit before the test process is terminated. The problem formulation is given in Figure 3.

V. Heuristics for Test Scheduling

A. Straight-Forward Approach

In [8], a thermal-aware test scheduling algorithm was proposed to minimize the TAT under temperature and test-bus bandwidth limits, assuming that thermal influences between cores are negligible. For convenience, we denote this algorithm with *ALG0*. Directly applying *ALG0* for SoCs with significant lateral thermal influences may not generate thermally safe test schedules, as the maximum temperature occurred during test may exceed the temperature limit. Figure 4 shows such a scenario where core temperatures exceed the temperature limit because of applying *ALG0* on a SoC with lateral thermal influences.

Input:
SoC floorplan *FLP* including physical parameters of die and package,
Set of defect probability for each core $\{DP_i \mid i = 1, 2, \dots, n\}$,
Set of test set for each core $\{TS_i \mid i = 1, 2, \dots, n\}$,
Set of required test-bus bandwidth for each test $\{W_i \mid i = 1, 2, \dots, n\}$,
Test-bus bandwidth limit *BL*,
Temperature limit *TL*.

Output:
Test schedule of the partitioned test sets with the minimized expected test application time (ETAT).

Subject to the following constraints:
1. At any time moment t before the test process is terminated, total amount of allocated test-bus bandwidth $BW(t)$ is less than or equal to bandwidth limit *BL*, i.e. $\forall t$, $BW(t) \le BL$ *where* $BW(t) ::= \Sigma_j BW_j(t)$;
2. At any time moment t before the test process is terminated, instantaneous temperature $TEM_i(t)$ of every core C_i is less than or equal to temperature limit *TL*, i.e. $\forall t$, $\forall i$, $TEM_i(t) \le TL$.

Figure 3. Problem formulation

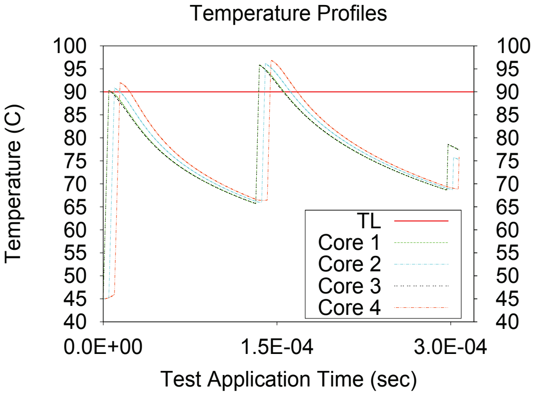

Figure 4. An example showing that ALG0 cannot guarantee thermal safety when lateral thermal influence is significant

In this paper, we first consider a straight-forward approach (SFA) that extends *ALG0* to the AOFF test environment by calculating the ETAT for the test schedules generated by *ALG0*. A simple solution to reduce the maximum temperature is to perform *ALG0* with a lower temperature limit. We denote the originally imposed temperature limit with TL_{orig}, the new temperature limit with TL_{new}, and the maximum temperature occurred in the thermal simulation result with T_{max}. The difference between the new temperature limit and the originally imposed temperature limit, denoted with d, is defined as

$$d = T_{max} - TL_{orig} \tag{2}$$

and the new temperature limit is given by

$$TL_{new} = TL_{orig} - d \tag{3}$$

Then, *ALG0* is invoked with TL_{new} and a new test schedule is generated. A thermal simulation is performed again to check if the new test schedule is thermally safe. This procedure is repeated until the first thermally safe test schedule is generated.

The test schedule generated in this way can be pessimistically long because the adjusted temperature limit may be lower than needed. In order to reduce the pessimism in terms of long ETAT, we use the same procedure to increase the imposed temperature limit until T_{max} is sufficiently close to but smaller than TL_{orig}. The flowchart of the SFA is depicted in Figure 5, where D ($D > 0$) denotes a given threshold for d, m denotes the number of iteration steps, and M denotes a given threshold for the total number of iteration steps.

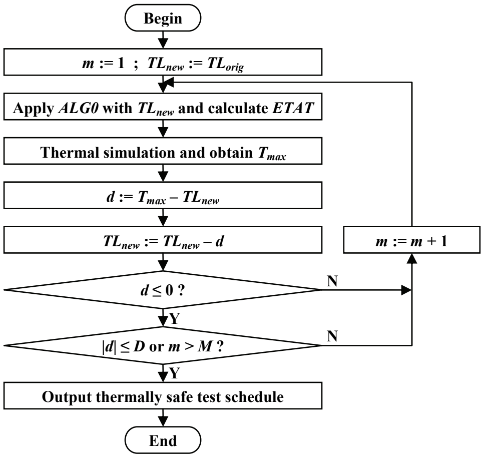
Begin
$m := 1$; $TL_{new} := TL_{orig}$
Apply $ALG0$ with TL_{new} and calculate $ETAT$
Thermal simulation and obtain T_{max}
$d := T_{max} - TL_{new}$
$TL_{new} := TL_{new} - d$
$m := m + 1$
$d \leq 0$?
N
Y
$|d| \leq D$ or $m > M$?
N
Y
Output thermally safe test schedule
End

Figure 5. Straight-forward approach (SFA)

B. Simulation-Driven Scheduling Approach

Although the SFA can generate thermally safe test schedules, the long ETAT of the generated test schedules as well as the long optimization times make it far from efficient. Therefore, we propose a simulation-driven scheduling approach (SDSA) which generates efficient test schedules with short ETAT while guaranteeing the thermal safety.

The SDSA employs instantaneous thermal simulation to guide the test set partitioning and interleaving. The employed thermal simulator is the ISAC system [23], which takes the floorplan of a chip and the power consumption profiles of individual cores as inputs, and calculates the temperature value of every core at every simulation cycle. During the temperature calculation, the lateral thermal influences among cores are taken into account.

We have developed a finite state machine model to guide the test set partitioning and interleaving during the thermal-simulation driven test scheduling process. A core has three states, namely *inactive*, *active*, and *finished*, corresponding to the cases that the core is not being tested, the core is being tested, and the test for the core is finished, respectively. When the test scheduling process starts, we assume that all cores are at the *inactive* state and their temperatures are equal to the ambient temperature. When a core is selected (see *ALG1* in Figure 6) for test and the required test-bus bandwidth is allocated for the test, a flag *start_test* is set to 1 and the state of the core moves from *inactive* to *active*. While test patterns are applied to the core, the temperature of the core, denoted with *TEM*, increases, and the state of the core remains *active* until the temperature reaches temperature limit *TL* or the test is finished. As soon as the test is finished, the state of the core moves from *active* to *finished*. Otherwise, when the core temperature reaches *TL*, the core state moves from *active* to *inactive* and remains unchanged until the core temperature decreases to stop-cooling temperature *CL*, from which the core state moves repeatedly between *active* and *inactive* until the test is finished. The test scheduling process terminates when all cores are at the *finished* state.

```
ALG1. ACTIVATE(Queue of inactive cores ready for test :: Q)
01  if (IsNotEmpty(Q)) then
02     Sort Q decreasingly according to
          (#_of_rem_test_patt × core_defect_prob / curr_tem);
03     while (GetRemainingBandwidth() > 0 & IsNotEmpty(Q)) loop
04        CurrentCore = GetFirstElement(Q);
05        ReqBwd = GetBandwidthRequirement(CurrentCore);
06        if (ReqBwd <= GetRemainingBandwidth()) then
07           Move the state of CurrentCore to active;
08           SubtractBandwidthRemainder(ReqBwd);
09           Remove(CurrentCore, Q);
10        else
11           break loop;
12        end if-then-else
13     end while
14  end if
```

Figure 6. Pseudo-code of heuristic activating cores for test

Using the FSM to guide the test set partitioning can guarantee thermal safety for the generated test schedules. However, the scheduling of test sub-sequences should also take into account the test-bus bandwidth limit and the ETAT. This is solved by a heuristic given in Figure 6. The heuristic, denoted with *ALG1*, allocates bandwidth to some of the cores ready to test and activate them for test.

ALG1 takes a queue of all inactive cores ready for test as an input. It allocates the required test-bus bandwidth to some of the cores and changes their states to *active*. The heuristic first sorts the queue decreasingly according to the number of remaining test patterns multiplied by the core defect probability divided by the current core temperature (Line 2). This means that a higher priority is given to a core which has a larger number of remaining test patterns, a higher defect probability, and a lower temperature. As such, the physical parameters (the sizes of cores as well as the distances between cores) have been taken into account, because the temperature values of the cores are given by the thermal simulator which addresses the lateral thermal influence issue. Then the heuristic allocates the required bandwidth to the cores according to their priorities until there is no sufficient bandwidth to allocate or all cores are activated for test. (Lines 3 through 13).

The overall strategy to generate thermally safe test schedules with minimized ETAT is illustrated in Figure 7. The test scheduling algorithm iteratively explores alternative solutions by using different stop-cooling temperatures. At every iteration step, a thermally safe test schedule is generated by invoking *ALG1* with a new stop-cooling temperature, denoted with CL_{new}. A counter, denoted with *k*, is used to count the number of consecutive iteration steps at which the reduction of ETAT is no larger than a given threshold, denoted with ε ($\varepsilon > 0$). If the ETAT of the newly generated test schedule is less than the minimal ETAT of the best solution obtained through

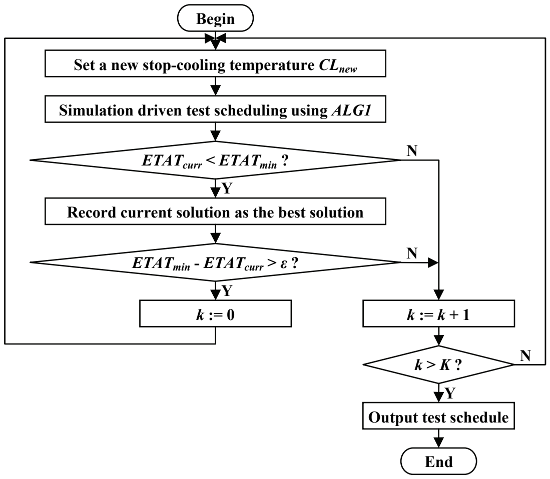
Begin
Set a new stop-cooling temperature CL_{new}
Simulation driven test scheduling using *ALG1*
$ETAT_{curr} < ETAT_{min}$?
N
Y
Record current solution as the best solution
$ETAT_{min} - ETAT_{curr} > \varepsilon$?
N
Y
$k := 0$
$k := k + 1$
$k > K$?
N
Y
Output test schedule
End

Figure 7. Overall solution strategy for the SDSA

previous iteration steps, the current solution is recorded as the best solution. Further, if the reduction of ETAT is greater than ε, counter *k* is reset to 0. In the cases that the current ETAT is larger than the minimal ETAT or the reduction of ETAT is less than ε, counter *k* is incremented by 1. This procedure repeats until *k* is larger than a given threshold, denoted with *K*. Thereafter the optimized test schedule is output and the test scheduling process terminates.

By checking the temperature value of each core at every simulation cycle, the test scheduling algorithm restricts the core temperature between *CL* and *TL*, after the core temperature is raised from the ambient temperature to *CL*. With respect to different *CL*s, alternative test schedules based on various test set partitioning schemes are explored. Figure 8 depicts ETATs with respect to different *CL*s for a SoC consisting of four cores. The optimal *CL* is 82.26°C and the corresponding minimal ETAT is 1.015×10^{-4} seconds.

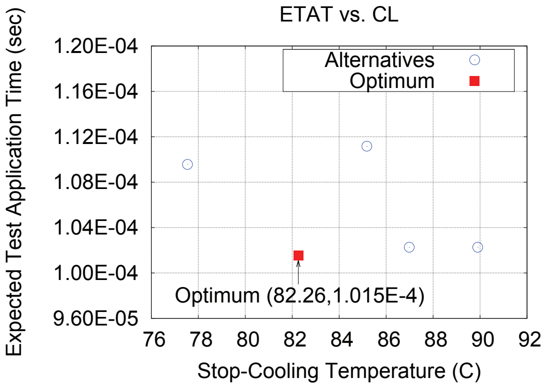

Figure 8. ETATs w.r.t. CLs

VI. Experimental Results

We have randomly selected designs from the ISCAS'89 benchmarks as the cores of SoCs used for our experiments. 7 different SoCs are employed for our experiments, with the number of cores varying from 6 to 42. The physical sizes of different cores are different, depending on the complexity of the core. The defect probabilities of individual cores are generated randomly such that the system defect probability equals 0.6. The power consumption profiles of the cores under test are obtained by using a cycle-accurate power estimation approach proposed in [19], which calculates the power consumption in watt according to the amount of switching activity. Taking the power consumption profiles of individual cores as an input, the thermal simulator, ISAC, calculates instantaneous temperature values of all individual cores at every cycle of the test process. The imposed temperature limit (*TL*) for our experiments is 90°C and the assumed scan frequency is 100MHz. The results of the thermal simulations performed on the generated test schedules have shown that the temperatures of all cores under test are less than the imposed temperature limit. For each SoC, we perform 3 experiments with respect to 3 different test-bus bandwidth limits (low, medium, and high). The average values of the experimental results are calculated and listed in the tables in this section.

In order to demonstrate that the proposed thermal-simulation driven test scheduling approach (SDSA) is efficient in minimizing the ETAT while guaranteeing the thermal safety, we have compared the SDSA with the straight-forward approach (SFA). The experimental results are shown in Table 1. The first column in the table lists the number of cores used in the designs. Columns 2 and 4 show the expected test application times (in seconds) of the test schedules generated by the SFA and the SDSA, respectively. Columns 3 and 5 list the CPU times (in seconds) needed to generate the test schedules using the corresponding approaches. Column 6 shows the ETAT reduction (in percentage) by using the SDSA versus the SFA. It can be seen that the ETAT is about 20% to 50% lower when using the SDSA than the SFA. The CPU times

Table 1. SDSA vs. SFA

# of Cores	SFA		SDSA		ETAT Reduction
	ETAT (s)	*CPU Time (s)*	*ETAT (s)*	*CPU Time (s)*	
6	10903.15 E-4	1218.67	8759.95 E-4	1165.33	19.66%
12	13586.17 E-4	1482.67	9916.61 E-4	1310.33	27.00%
18	13140.36 E-4	4612.67	9243.19 E-4	1770.33	29.66%
24	13423.10 E-4	4293.33	10022.23 E-4	8632.33	25.38%
30	15611.01 E-4	12651.00	10005.39 E-4	4740.67	35.98%
36	15791.91 E-4	32526.33	9969.47 E-4	3265.00	36.85%
42	18470.58 E-4	40022.67	9335.08 E-4	5847.33	49.54%

TABLE 2. SDSA VS. NSA

# of Cores	NSA		SDSA		ETAT Reduction
	ETAT (s)	***CPU Time (s)***	***ETAT (s)***	***CPU Time (s)***	
6	8807.05 E-4	1154.00	8759.95 E-4	1165.33	0.53%
12	10975.15 E-4	1206.67	9916.61 E-4	1310.33	9.32%
18	10413.96 E-4	1238.00	9243.19 E-4	1770.33	10.68%
24	12717.56 E-4	1437.00	10022.23 E-4	8632.33	20.81%
30	10992.45 E-4	1448.33	10005.39 E-4	4740.67	8.87%
36	11795.83 E-4	4749.67	9969.47 E-4	3265.00	15.43%
42	11084.48 E-4	1335.33	9335.08 E-4	5847.33	15.86%

are usually shorter when using the SDSA. This is because the SFA requires more invocations of thermal simulation than the SDSA does. For a SoC consisting of *n* cores, the SFA invokes *n* thermal simulations at each iteration step, as generating the initial partitioning scheme for every core needs to perform a thermal simulation. However, the SDSA only invokes one thermal simulation for all cores at each iteration step.

In order to demonstrate the necessity and efficiency of using the SDSA in an AOFF test environment, we compare the SDSA with a naive scheduling approach (NSA). The NSA is based on the approach proposed in [9], and aims to minimize the total test application time with thermal awareness. When obtaining a test schedule using the NSA, we calculate the ETAT of the obtained test schedule and compare it with the ETAT by using SDSA. The experimental results have shown that directly applying the NSA for test scheduling in an AOFF test environment results in longer ETATs than using the SDSA. The detailed experimental results are listed in Table 2.

VII. CONCLUSIONS

In this paper, we have proposed a thermal-aware test scheduling technique for modular systems-on-chip test. The objective of the proposed technique is to minimize the expected test application time in an abort-on-first-fail test environment while guaranteeing the thermal safety during test. The test scheduling algorithm employs a thermal simulation to partition and interleave test sets on-the-fly such that the expected test application time of the generated test schedule is minimized while satisfying the temperature limit and test-bus bandwidth limit. Experimental results have shown the efficiency of the proposed technique.

REFERENCES

[1] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, Vol. 19, No. 4, pp. 23-29, 1999.

[2] K. Chakrabarty, "Design of system-on-a-chip test access architectures under place-and-route and power constraints," *IEEE/ACM Design Automation Conf.*, 2000, pp. 4332-437.

[3] R. Chou, K. Saluja, and V. Agrawal, "Scheduling tests for VLSI systems under power constraints," *IEEE Trans. VLSI Systems*, 5(2):175-184, June 1997.

[4] P. Girard, C. Landrault; S. Pravossoudovitch, and D. Severac, "Reducing power consumption during test application by test vector ordering," *Int. Symp. Circuits and Systems*, 1998, pp. 296-299.

[5] S. K. Goel and E. J. Marinissen, "Control-aware test architecture design for modular SoC testing," *European Test Workshop*, 2003. pp. 57-62.

[6] S. Gunther, F. Binns, D. M. Carmen, and J. C. Hall, "Managing the impact of increasing microprocessor power consumption," *Intel Technology J.*, 2001.

[7] Z. He, Z. Peng, and P. Eles, "Power constrained and defect-probability driven SoC test scheduling with test set partitioning," *Design Automation and Test in Europe Conf.*, 2006, pp. 291-296.

[8] Z. He, Z. Peng, and P. Eles, "A heuristic for thermal-safe SoC test scheduling," *IEEE Int. Test Conf.*, 2007, pp. 1-10.

[9] Z. He, Z. Peng, and P. Eles, "Simulation-driven thermal-safe test time minimization for system-on-chip," *IEEE Asian Test Symp.*, 2008, pp. 283-288.

[10] W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," *IEEE/ACM Design Automation Conf.*, 2004. pp. 878-883.

[11] U. Ingelsson, S. Goel, E. Larsson, and E. J. Marinissen, "Test scheduling for modular SOCs in an abort-on-fail environment," *IEEE European Test Symp.*, 2005, pp. 8-13.

[12] W. Jiang, and B. Vinnakota, "Defect-oriented test scheduling," *IEEE Trans. VLSI Systems*, Vol. 9, No. 3, pp. 427-438, June 2001.

[13] E. Larsson and Z. Peng, "Power-aware test planning in the early system-on-chip design exploration process," *IEEE Trans. Computers*, Vol. 55, No. 2, pp. 227-239, Feb. 2006.

[14] E. Larsson, J. Pouget, and Z. Peng, "Defect-aware SOC test scheduling," *IEEE VLSI Test Symp.*, 2004, pp. 359-364.

[15] R. Mahajan, "Thermal management of CPUs: a perspective on trends, needs and opportunities," *Int. Workshop THERMal INvestigations of ICs and Systems*, 2002.

[16] B. Pouya and A. Crouch, "Optimization trade-offs for vector volume and test power," *Int. Test Conf.*, 2000, pp. 873-881.

[17] P. Rosinger, B. M. Al-Hashimi, and K. Chakrabarty, "Thermal-safe test scheduling for core-based system-on-chip integrated circuits," *IEEE Trans. CAD of ICs and Systems*, Vol. 25, No. 11, pp. 2502-2512, Nov. 2006.

[18] P. Rosinger, B. M. Al-Hashimi, and N. Nicolici, "Scan architecture with mutually exclusive scan segment activation for shift- and capture-power reduction," *IEEE Trans. CAD of ICs and Systems*, Vol. 23, No. 7, pp. 1142-1153, July 2004.

[19] S. Samii, E. Larsson, K. Chakrabarty, and Z. Peng, "Cycle-accurate test power modeling and its application to SoC test scheduling," *IEEE Int. Test Conf.*, 2006, pp. 1-10.

[20] C. Shi and R. Kapur, "How power-aware test improves reliability and yield," *EE Times*, Sep. 15, 2004. [Online] http://www.eetimes.com/showArticle.jhtml?articleID=47208594.

[21] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," *Int. Symp. Computer Architecture*, 2003, pp. 2-13.

[22] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: modeling and implementation," *ACM Trans. Architecture and Code Optimization*. Vol. 1, No. 1. pp. 94-125, Mar. 2004.

[23] Y. Yang, Z. P. Gu, C. Zhu, R. P. Dick, and L. Shang, "ISAC: Integrated Space and Time Adaptive Chip-Package Thermal Analysis," *IEEE Trans. CAD of ICs and Systems*. Vol. 26, No. 1, pp. 86-99, Jan. 2007.

[24] T. Yu, T. Yoneda, K. Chakrabarty, and H. Fujiwara, "Thermal-safe test access mechanism and wrapper co-optimization for system-on-chip," *IEEE Asian Test Symp.*, 2007, pp. 187-192.