Thermal-Aware SoC Test Scheduling with Test Set Partitioning and Interleaving

Zhiyuan He, Zebo Peng, Petru Eles
Embedded Systems Laboratory (ESLAB)
Linköping University
SE-581 83 Linköping, Sweden
{zhihe, zebpe, petel}@ida.liu.se

Paul Rosinger, Bashir M. Al-Hashimi
School of Electronics and Computer Science
University of Southampton
United Kingdom SO17 1BJ
{pmr, bmah}@ecs.soton.ac.uk

Abstract[1]

High temperature has become a major problem for system-on-chip testing. In order to reduce the test time while keeping the temperature of the chip under test within a safe range, a thermal-aware test scheduling technique is required. This paper presents an approach to minimize the test time and, at the same time, prevent the temperature of cores under test going over the given upper limit. We employ test set partitioning to divide test sets into shorter test sequences, and add cooling spans between test sequences, so that overheating can be avoided. Moreover, test sequences from different test sets are interleaved, in order to utilize the cooling spans and the test bus bandwidth for test data transportation, hence the total test time is reduced. The test scheduling problem is formulated as a combinatorial optimization problem, and we use constraint logic programming (CLP) to solve it in order to obtain the optimal solution. As the CLP approach needs relatively long time for execution, we have also developed a heuristic to generate the near-optimal test schedule with much shorter computation time. Experimental results have shown the efficiency of both the CLP and heuristic approach.

1. Introduction and related work

Integrated circuits have become more and more complex in order to meet the increasing application requirements, consequently raising the production cost. Recently, a core based system-on-chip (SoC) design methodology has been proposed to reduce the design complexity by integrating pre-designed and pre-verified intellectual property (IP) cores. Although the design cost of such core-based systems has been reduced, the cost of testing increases instead, since a larger quantity of test data is required and therefore a longer test time is expected [1]. In order to reduce the test time, a lot of research efforts have been devoted to increasing the test concurrency by developing advanced test architectures and test scheduling algorithms [2, 3, 4, 5, 6, 7, 8, 9]. It is known that the power consumption of a chip during test is much higher than in normal operation mode, because of the increased switching activities in the circuits [10, 11]. As a result, thermal-safe testing has become more and more important.

The thermal problem becomes more severe since deep submicron technology is used. The power density of the chip dramatically increases since more transistors are integrated into the same area. A direct consequence of the increasing power density is the higher junction temperature that poses several problems [12, 13, 14, 15]. High junction temperature decreases the carrier mobility of electrons and therefore reduces the driving current of CMOS transistors, which consequently degrades the circuit performance. The reliability and lifespan of the circuits are also decreased at high temperatures.

In this paper, we aim to minimize the test time by generating efficient test schedules, while taking into account the thermal issue in order to avoid overheating the chip. In the case that the core temperature goes beyond an upper limit before the test completion, a test set will be partitioned into shorter test sequences and a cooling span is introduced between two consecutive test sequences. Because of the test set partitioning and the cooling phase, the test time is substantially increased. Thus we allow the interleaving

[1] This work has been partially supported by the Swedish Foundation for Strategic Research (SSF) under the Strategic Integrated Electronic Systems Research (STRINGENT) program, and partially supported by the EPSRC, U.K., under grant GR/S95770 and GR/S05557

of test sequences that belong to different test sets, so that test time can be reduced. Based on the test set partitioning and interleaving, we propose a test scheduling technique to minimize the test time.

Several optimization techniques have been proposed to solve the constrained SoC test scheduling problem. The optimization objectives can be the test time, memory size, area size, energy/power consumption, number of pins, or the number of wires in the test access mechanism (TAM), etc. The constraint(s) can be one or multiple factors such as the memory size constraint in a problem of minimizing the test power consumption. Usually such constrained optimization problems can be formulated as an existing optimization problem, such as the bin packing problem or rectangular packing problem [16, 17]. Our previous work [9] solved a test time minimization problem with a power constraint, which was formulated as a non-classical two-dimensional rectangular packing problem.

Recently, thermal-aware test scheduling has attracted a lot of research interests, since this issue has become more and more critical for testing high-performance systems. In [26], Liu et al. propose a technique to evenly distribute the generated heat across the chip during tests, so that high temperature can be avoided. In [27] Rosinger et al. propose algorithms based on clique partitioning to generate thermal-safe test schedules with minimized test time. They take into account the adjacency of cores and utilize this information to drive the test scheduling and reduce the temperature stress between cores. In this paper, we utilize test set partitioning and interleaving to generate thermal-safe test schedules, and provide an approach to find the test schedule with the minimum test time.

The rest of this paper is organized as follows. The next section presents the assumed test architecture. In Section 3, the temperature simulation for system-on-chip cores is introduced. In Section 4 the test set partitioning and interleaving approach is demonstrated, and a motivational example is given to illustrate the thermal-aware test scheduling problem. Section 5 gives the problem formulation. In Section 6, the constraint logic programming model used in our work is described, and a heuristic is proposed in Section 7. Experimental results are shown in Section 8 and the paper is concluded in Section 9.

2. Basic test architecture

We have assumed a test architecture using a test bus to transport test data between the cores under test and an embedded tester or an external automated test equipment (ATE). Each core on the chip is connected to the test bus through a number of dedicated TAM wires. The tester has a local memory that stores a generated test schedule and all the test patterns for all cores. During the test, the test controller sends the test patterns to the test bus, at particular time moments, according to the test schedule. The cores under test then get the test patterns transported through the test bus and the test patterns for each activated core are applied. Test responses are also transported through the test bus from the core to the tester for analysis.

Figure 1 gives an example of the assumed test architecture for a system with five cores. In this example, an embedded tester consisting of a test controller and a tester memory is integrated on the chip. Test patterns are stored in an on-chip memory and are applied through a test bus, such as a dedicated TAM or the advanced microcontroller bus architecture (AMBA) [18]. Note that the embedded tester in the example can be replaced by an ATE.

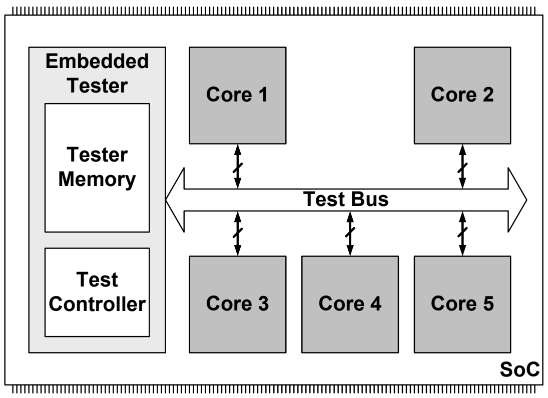


Figure 1. Basic test architecture

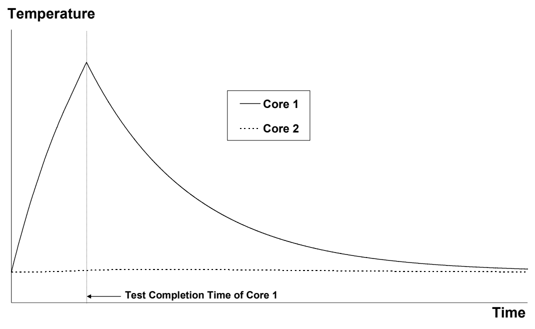
3. SoC temperature simulation

In order to obtain the temperature of a core during tests, we have employed an architectural-level temperature simulator, Hotspot [19,20], to simulate the thermal behavior of the chip under test. HotSpot uses a compact thermal model [21] for an assumed circuit packaging configuration which consists of the following layers from top to bottom, the heat sink, the heat spreader, thermal interface material, the silicon bulk, interconnect layers, I/O pads, the ceramic substrate, and joint balls. There are three major heat flow paths in the chip package [20,21]. In the vertical direction to upper layers, heat is generated on the silicon bulk and transferred through the thermal interface material, heat spreader, and heat sink to the ambient. In the vertical direction to lower layers, heat is conducted from the silicon bulk through the interconnect layers, I/O pads, ceramic substrate and joint balls to the printed-circuit board. In the lateral direction, heat is conducted between blocks at the same layer.

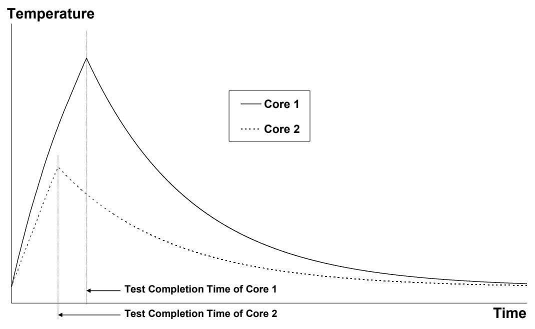
Temperature simulation in HotSpot is based on the duality between thermal and electrical quantities [15] and the proposed compact thermal model. It divides large blocks into smaller units at different layers and builds a network of thermal resistances and capacitances for all units. Taking the chip floorplan and the average power consumption of each functional block unit as inputs, HotSpot calculates the instantaneous temperature for each unit by deducing the RC network into lumped RC values and solving the first-order differential formula with the 4-th order Runge-Kutta method.

We utilized HotSpot to simulate instantaneous temperatures for the cores being tested. As mentioned above, the heat can be transferred vertically and/or laterally. Although both of the two heat flow paths conduct heat from the active core to its adjacent unit, they play different roles. The vertical heat transfer mainly takes away the heat to the ambient, while the lateral heat path conducts the heat from one core to another. Consequently, there are two different scenarios on the temperature distribution across the chip. When the vertical heat transfer dominates, the temperature of a core has very little impact on its adjacent cores. On the other hand, if the lateral heat transfer is not neglectable, the temperature of a core can have an impact on its adjacent cores. Whether this is the case depends on the package construction, materials at different layers, and also the physical dimension of the units.

We assume in this paper that the same packaging configuration as modeled in HotSpot is used for the SoC designs, and the contact area size of a core in the vertical direction is much larger than that in the lateral direction. Thus the vertical heat transfer dominates the whole heat exchanges in the system and the temperature influence between cores is neglectable. This is a typical situation which is the characteristic in the most widely used circuits. In Figure 2, the temperature profiles of two adjacent cores under test are depicted from the temperature simulation results. Figure 2(a) illustrates the case that Core 1 is tested while Core 2 is not. It can be observed that the temperature of Core 2 remains nearly constant even when the temperature of Core 1 increases rapidly while it is tested. Figure 2(b) shows the situation that both Core 1 and Core 2 are tested. By comparing with the temperature curve of Core 1 in Figures 2(a) and 2(b), we can see that the temperature of Core 2 does not significantly influence the temperature of Core 1, even when Core 2 is tested in parallel.



(a). The case that only Core 1 is tested



(b). The case that both cores are tested

Figure 2. Temperature profiles of two adjacent cores under test

4. Test set partitioning and interleaving

As mentioned in the introduction, the temperature of a core being tested increases as the test time elapse until it reaches a steady state temperature. The steady state temperature is usually much higher than an upper allowed limit, beyond which the core may be damaged. Therefore, when the temperature of the core reaches the upper allowed limit, the test has to be stopped for a period in order to let the core be cooled down. When the required cooling has been achieved, the test is restarted. Thus the entire test set is partitioned into several test sequences so that the temperature upper limit is not violated. The example in Figure 3 illustrates a situation in which a test set TS_i is partitioned into four test sequences, namely TS_{i1}, TS_{i2}, TS_{i3}, and TS_{i4} , which successfully avoids violating the upper temperature limit.

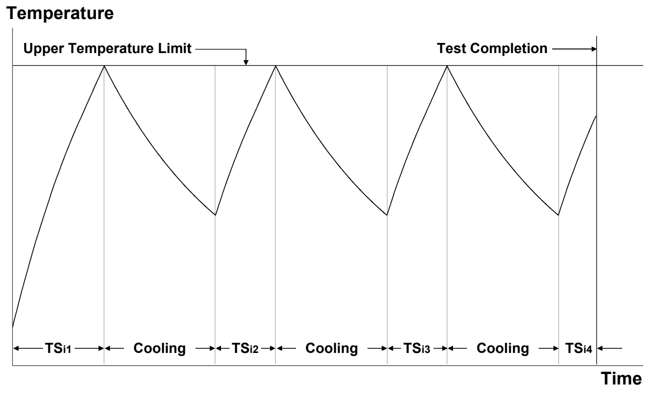


Figure 3. An example of test set partitioning

It is useful to generate test partitions of equal length and to keep the cooling spans of identical length as well. This significantly reduces the complexity of the test controller. In Figure 3, test sequences TS_{i2} and TS_{i3} have equal length and all cooling spans between two consecutive test sequences are identical. Note that the first test sequence, TS_{i1}, is longer than the normal length, since the core starts with the initial temperature which is much lower than the operation temperature during the test. And the last test sequence, TS_{i4} in this example, is usually shorter, since the entire test set has been nearly completed.

It is obvious that after partitioning a test set into several test sequences, the test bus bandwidth is not efficiently utilized, since the bandwidth of the test bus allocated to this test is not utilized during the cooling spans. A direct consequence of this is the increased test time. In order to avoid low utilization of the test bus and long test times, we use the bus bandwidth allocated to core C_i , during its cooling spans, to transport test data for another core C_j ($j \neq i$)and test core C_j . Thus typically several tests are interleaved and the total test time will be much shortened, without necessarily increasing the total amount of consumed bus bandwidth. Figure 4 gives an example where two partitioned tests are interleaved.

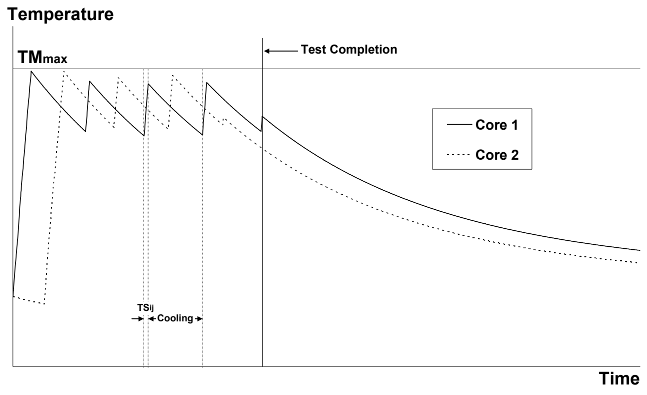
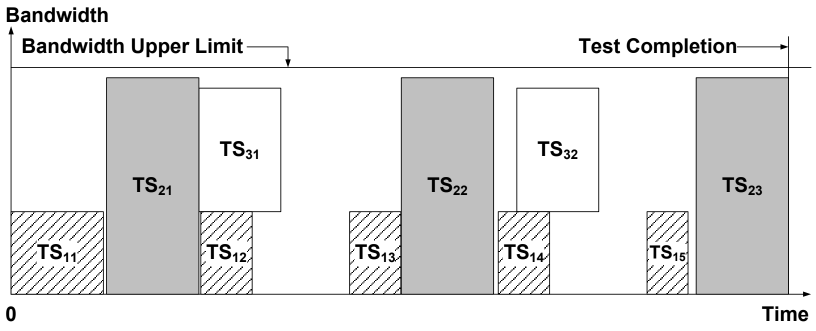


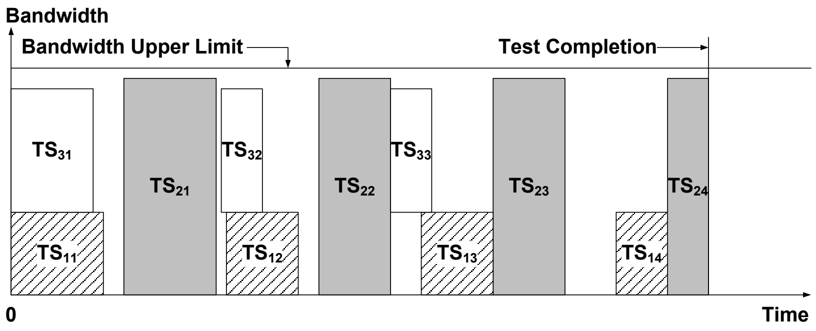
Figure 4. An example of test set interleaving

Partitioning a test set into several test sequences can introduce time overheads for the partitioned tests due to the switching of different tests that are interleaved [22]. Therefore, increasing the number of partitions can lead to longer test time. On the other hand, less number of test partitions does not necessarily reduce the test time, especially when the interference between the different test are considered. Therefore, elaborately selecting the number of partitions for every test is very important to solve the problem of minimizing the total test time. When the number of partitions for each test has been decided, the rest of the problem is how to schedule all the test sequences such that the test time is minimized under the limitation of the total available bus bandwidth. Note that our test scheduling problem is similar to a rectangular packing problem, while the partitioning and interleaving of test sets and the introducing of cooling spans between partitioned test sequences makes our problem even more complex.

Figure 5 gives a motivational example that explains the significance of our optimization problem. In Figure 5(a), a feasible test schedule is generated for three test sets, TS_1, TS_2, and TS_3. The three test sets are partitioned into 5, 3, and 2 test sequences, respectively. In Figure 5(b), all the three test sets are repartitioned, with the number of partitions 4, 4, and 3, respectively. A new test schedule is then generated and the test time is reduced. This example shows the possibility to find a better solution among alternative partitioning schemes and test schedules. Note that by selecting different number of partitions, the cooling spans between test sequences are also changed. For example, the cooling spans between the partitioned test sequences of the test set TS_1 in Figure 5(a) is longer than that in Figure 5(b), since the test sequences in Figure 5(a) have longer length.



(a) A feasible test schedule



(b) An alternative test schedule with a shorter test time

Figure 5. A motivational example that illustrates our optimization problem

5. Problem formulation

Suppose that a system S, consisting of n cores $C_1, C_2, \dots, C_n$, has the test architecture illustrated in Figure 1. Each core in the system is connected to the test bus with a constant number of TAM wires W_i. The total test bus bandwidth is B ($\forall W_i \leq B$). The system has the maximum tolerable temperature TM_{max} which should not be violated for any cores at any time. Each core C_i ($i = 1, 2, \dots, n$) is assigned a test set TS_i which consists of l_i ($l_i > 0$) test patterns. Moreover, a test set can be partitioned into a number of test sequences. Suppose that test set TS_i is partitioned into m_i ($1 \le m_i \le l_i$) test sequences. With TS_{ij} ($j = 1, 2, ..., m_i$) we denote the j-th test sequence of test set TS_i. Note that when $m_i = 1$, the test set has only one partition and the term "test sequence" is referred to the entire test set in this case, if not mentioned otherwise.

In order to reduce the required memory for storing the start time and length of each partition, and also to reduce the complexity of the test controller, we assume that the test partitions in the same test set have identical time durations except the first one[2] and the last one. For the same reason, the cooling span between two consecutive test sequences from the same test set should be of identical length as well.

Each partitioning scheme has three parameters, the number of partitions m_i, the time duration of the first test sequence l_{i1}, and the cooling span d_i between two consecutive test sequences. Each test starts at time t_i which is equal to the start time of the first partition in the same test set.

$$t_i = t_{i1}$$

The number of partitions and the test start time is decided during the optimization. The start time t_{ij} and finishing time e_{ij} of the test sequence TS_{ij} can be calculated as follows. Note that o_i is the time overhead.

$$t_{ij} = t_{i,j-1} + l_{i,j-1} + d_i + o_i \quad (2 \le j \le m_i\, ,\ 1 \le i \le n)\, ,$$
$$e_{ij} = t_{ij} + l_{ij} \quad (1 \le j \le m_i\, ,\ 1 \le i \le n)$$

The last test sequence in each test set is special since its finishing time is the end of the whole test set. Thus the finishing time e_i of the test set TS_i is

$$e_i = e_{i,m_i}$$

and the total test application time TTT for testing all cores is the maximum of all single test finishing times.

$$TTT = \max_{1 \le i \le n}\left(e_{i,m_i}\right)$$

The total test application time TTT is the cost function of our optimization problem, and our objective is to find the optimal solution $\{(m_i^*\, , t_i^*) \mid i = 1, 2, ..., n\}$ such that the total test application time is minimized, subject to the following two constraints:

1. the total amount of test bus bandwidth used by the concurrent test sequences at any time moment is less than the total allowable bandwidth, that is

$$\sum_{k=1}^{p_x} W_k \le B$$

where p_x is the number of concurrent test sequences at any time moment x.

2. the temperature of each core $TM_{i,x}$ at any time moment x should be less than the temperature upper limit TM_{max} , i.e.

$$TM_{i,x} \le TM_{max}$$

We assume that a test TS_i is started when the core has the ambient temperature TM_{amb} , and the test set has to be partitioned into a number of test sequences if the entire test is so long that the temperature of the core goes over the TM_{max} before completion. The length of each partition and the total number of test sequences also depends on the cooling span d_i between two consecutive test sequences. This is because a longer cooling span leads to a lower temperature for the succeeding test sequence to start at, and therefore the succeeding test sequence can be longer which leads to a smaller number of test sequences. It is important to find a possible interval of the number of partitions for each test set, since our optimization algorithm explores alternative partitioning schemes between the minimum and maximum values in the interval: $I_i = [I_{i,min}\, , I_{i,max}]$ ($i = 1, 2, ..., n$)

We used the temperature simulation to find the interval I_i of the number of partitions for each test set TS_i. It is obvious that a relatively long time is needed in order to cool down a core to the ambient temperature, as the temperature decreases slowly at a lower temperature level (see the dashed curve in Figure 6). Thus we let a core to be cooled down until the slope of the cooling curve reaches (-1), and then we start the next test sequence. We define the number of partitions obtained by using this approach as the minimum value of the exploration interval I_i, denoted with $I_{i,min}$, which is applicable for our problem. In Figure 6, the example illustrates a test set TS_i partitioned into four test sequences (TS_{i1} to TS_{i4}), which gives the minimum number of partitions. Through experiments, we have found that the actual number of partitions for a test set in the

[2] We assume the first test partition starts from the ambient temperature and is not terminated until the core temperature reaches the allowed upper limit.

optimal solution is close the minimum value $I_{i,min}$. Thus we consider the upper limit of the interval I_i as $I_{i,max} = K + I_{i,min}$, where K is a constant fixed by the designer. Thereafter, the exploration interval $I_i = [I_{i,min}, I_{i,max}]$ ($i = 1, 2, \ldots, n$) for core C_i is taken as an input to the optimization algorithm.

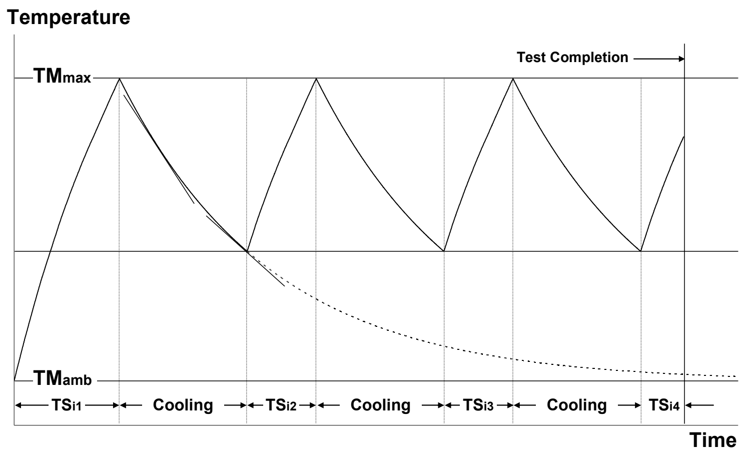


Figure 6. An example of test set partitioning

6. Constraint logic programming model

We have used constraint logic programming (CLP) to model the test set partitioning and scheduling problem and find the optimal solution.

CLP provides supports to define relationships or constraints among entities and let programmers focus on formulating the problems in their application domains [23]. CLP usually uses logic programming language to describe the constraints and finds feasible solutions for the specified problems. Some CLP tools also provide solvers to find the optimal solution using branch and bound or exhaustive search. We use CHIP [24] in our work.

In our work, there are two sets of decision variables, one is the number of partitions m_i for core C_i and the other is the start time t_i of the test for the core C_i. For a particular number of partitions, a test partitioning scheme is generated for the corresponding core. The two decision variables are instantiated during the optimization, and test schedules that satisfy the constraints are explored. In the end, the solver finds the optimal solution which provides the minimal total test application time.

We formulate the constraints for our problem as:

- Test sequences belonging to the same test set have to be applied in a particular order, i.e. the start time of a test sequence is equal to the finish time of the previous test sequence belonging to the same test set plus the cooling span.
$$t_{i,j+1} = e_{ij} + d_i + o_i$$
- The finishing time of a test sequence is equal to its start time plus the duration of the test sequence.
$$e_{ij} = t_{ij} + l_{ij}$$
- The finishing time of a test is equal to its start time plus all the test sequence duration and the all the cooling span between any two consecutive test sequences.
$$e_i = t_i + \sum_{j=1}^{m_i} l_i + d_i \times (m_i - 1)$$
- The total amount of bandwidth consumed by all the test sequences applied in parallel should be less than the test bus bandwidth limit.
$$\sum W_k \leq B$$

7. A heuristic approach

By using the exact approach with CLP, we can obtain the optimal solution for our test set partitioning and scheduling problem. However, the computation times are relatively long, especially for large designs (see experimental results). Therefore, we proposed a heuristic to generate test schedules which are of slightly longer test application times but need much less computation times. The heuristic chooses the partitioning scheme which has the minimum number of partitions for each design and schedule every test set to the earliest possible time. Based on the selected partitioning scheme, the heuristic fixes the order of all the test sets to be considered for scheduling by giving higher priorities to the test sets with larger sizes. The size of a test set is defined as the product of the time duration of the entire test set and the dissipated test bus bandwidth. Each test set is then scheduled to the earliest possible time moment, and all the partitioned test sequences belonging to the same test set remains their regular length and keeps identical cooling period between its predecessor and successor. The pseudo-code of the heuristic is depicted in Figure 7.

```
for (each test set $TS_i$) do
        choose the partitioning scheme with the minimum number of partitions;
        calculate the size of the entire test set based on the selected partitioning scheme, where
                size($TS_i$) = length($TS_i$) * bandwidth($TS_i$);
end for
Sort $TS = [TS_i \mid i = 1, 2, \dots, n]$ decreasingly by size($TS_i$);
for (each test set $TS_i$ in $TS$) do
        Schedule $TS_i$ to the earliest possible time moment;
end for
```

Figure 7. Pseudo-code of the proposed heuristic

8. Experimental results

We have used the ISCAS'89 benchmarks as cores for the SoC designs to our experiments. Table 1 shows the experimental results for five different SoC designs. The number of cores composing each generated SoC is listed in the first column of Table 1. For each SoC design, test patterns are generated for all cores in the design, and the switching activities are calculated for each test pattern. We used the approach in [25] to obtain the power consumption values, taking the switching activities of test patterns as inputs. HotSpot is used to find the total number of partitioning schemes for each SoC design, which are listed in the second column of Table 1. The imposed temperature limit is 90°C.

We used the developed CLP formulation to generate the optimal test schedule by selecting the number of partitions and the start time for each test. The third column of Table 1 is the problem size of each design, which is the number of partitioning schemes multiplied by the number of cores. The total test time of the optimal solution for each design is shown in the fourth column and the optimization time is listed in the fifth column.

When the test schedule for a design has been generated, we run the HotSpot temperature simulator for the generated test schedules to check if the temperatures of the cores were went over the upper limit. The simulation results confirm that the temperatures of cores are below the upper limit.

Table 1. Experimental results for 5 different designs using the CLP approach

Number of Cores	Total Number of Partitioning Schemes	Problem Size	Total Test Times (# of clock cycles)	CPU Times (ms)
4	7	28	2775	2141
12	8	96	8306	35359
24	20	480	9789	47500
36	20	720	10017	120219
48	20	960	10941	881766

We also did experiments to see how the optimization result is impacted by the given total number of partitioning schemes. In Table 2, four different number of partitioning schemes have been given to the optimization algorithm for the same design consisting of 6 cores. The optimal solution is the same in the three cases where the total number of partitioning schemes is 7, 10, and 15, respectively. In the case that the number of partitioning schemes is only 5, the total test time of the obtained solution is larger than those in the other three cases. This experiment shows that, for this design, the best solution does not correspond to the partitioning scheme found among the five alternative partitioning schemes, as indicated in the first line in Table 2. If we introduce 2 additional alternative partitioning schemes (see line 2 in Table 2), a better solution is found. However, more additional alternatives, up to 15, do not lead to better solutions.

Table 2. Experimental results for one design with different number of partitioning schemes

Number of Cores	Total Number of Partitioning Schemes	Problem Size	Total Test Times (# of clock cycles)	CPU Times (ms)
6	5	30	9574	10156
	7	42	9570	26031
	10	60	9570	31875
	15	90	9570	39797

We also ran the proposed heuristic to generate test schedules for the same SoC designs used in the experiments with the CLP approach. The experimental results are listed in Table 3. Comparing the results in Table 3 and Table 1, we can see that the test schedules generated by the heuristic is in average 14% longer than those by the exact approach, however the execution times are much shorter.

Table 3. Experimental results for 5 different designs using the proposed heuristic

Number of Cores	Total Number of Partitioning Schemes	Problem Size	Total Test Times (# of clock cycles)	CPU Times (ms)
4	7	28	3384	1
12	8	96	9682	15
24	20	480	10657	15
36	20	720	11524	31
48	20	960	11694	47

9. Conclusions

In this paper, we have presented an exact approach to minimize the total test time for core-based systems which have a temperature upper limit and a bus bandwidth limit. Based on the proposed test set partitioning and interleaving technique, we have used constraint logic programming to solve the optimization problem and obtained the optimal solution. Nevertheless, the optimization times for large designs are long. Therefore, a heuristic approach have been proposed to generate near-optimal solutions with much less computation time.

References

[1] B. T. Murray, and J. P. Hayes. "Testing ICs: Getting to the core of the problem". *IEEE Transactions on Computer*, Vol. 29, pp. 32-39, Nov. 1996.

[2] Y. Zorian, E. J. Marinissen, and S. Dey. "Testing Embedded Core-Based System Chips". *IEEE International Test Conference (ITC)*, 1998, pp. 130-143.

[3] Y. Huang, W.-T. Cheng, C.-C. Tsai, N. Mukherjee, O. Samman, Y. Zaidan, and S. M. Reddy. "Resource Allocation and Test Scheduling for Concurrent Test of Core-based SOC Design". *IEEE Asian Test Symposium (ATS)*, 2001, pp. 265-270.

[4] E. Larsson, and Z. Peng. “An Integrated Framework for the Design and Optimization of SOC Test Solutions”. *Journal of Electronic Testing; Theory and Applications (JETTA)*, Vol. 18, No. 4/5, pp. 385-400, 2002.

[5] J. Aerts, and E. J. Marinissen, “Scan Chain Design for Test Time Reduction in Core-Based ICs”, *International Test Conference (ITC)*, 1998, pp. 448-457.

[6] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, “Test Access Mechanism Optimization, Test Scheduling, and Test Data Volume Reduction for System-on-Chip”, *IEEE Transactions on Computer*, Vol. 52, No. 12, Dec. 2003.

[7] P. Varma, and B. Bhatia, “A Structured Test Re-use Methodology for Core-based System Chips”, *International Test Conference (ITC)*, 1998, pp. 294-302.

[8] R. Chou, K. Saluja, and V. Agrawal. “Scheduling tests for VLSI systems under power constraints”. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 5(2):175-184, June 1997.

[9] 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 Conference (DATE)*, 2006, pp. 291-296.

[10] B. Pouya and A. Crouch. “Optimization trade-offs for vector volume and test power”. *International Test Conference (ITC)*, 2000, pp. 873-881.

[11] C. Shi and R. Kapur. “How power aware test improves reliability and yield”. *EETimes*, Sep. 15 2004. http://www.eetimes.com/news/design/features/showArticle.jhtml?articleId=47208594&kc=4235.

[12] S. Borkar. “Design challenges of technology scaling”. *IEEE Micro*, Vol. 19, Issue 4, pp. 23-29, 1999.

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

[14] R. Mahajan. “Thermal management of CPUs: A perspective on trends, needs and opportunities”. Keynote presentation at the *8th Int’l Workshop on THERMal INvestigations of ICs and Systems*. 2002.

[15] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. “Temperature-aware microarchitecture: Modeling and implementation”. *ACM Transactions on Architecture and Code Optimization (TACO)*. Vol. 1, Issue 1. pp. 94-125, Mar. 2004.

[16] B. S. Baker, E. G. Coffman Jr., and R. L. Rivest. “Orthogonal Packings in Two Dimensions”. *SIAM J. of Computing*, Vol. 9, Issue 4, pp. 846-855, 1980.

[17] N. Lesh, J. Marks, A. McMahon, and M. Mitzenmacher. “Exhaustive Approaches to 2D Rectangular Perfect Packings”, *Elsevier Information Processing Letters*, Vol. 90, Issue 1, pp. 7-14, 2004.

[18] D. Flynn. “AMBA: Enabling Reusable On-Chip Designs”. *IEEE Micro*, Vol.17, No.4, 1997,pp. 20-27.

[19] W. Huang, S. Ghosh, K. Sankaranarayanan, K. Skadron, and M. R. Stan. “HotSpot: Thermal Modeling for CMOS VLSI Systems.” *IEEE Transactions on Component Packaging and Manufacturing Technology*. 2005. (to appear).

[20] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. “Temperature-Aware Microarchitecture.” *Int’l Symposium on Computer Architecture*, 2003, pp. 2-13.

[21] W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy. “Compact thermal modeling for temperature-aware design”. *Design Automation Conference (DAC)*, 2004. pp. 878-883.

[22] S. K. Goel, and E. J. Marinissen. “Control-aware test architecture design for modular SOC testing”. *European Test Workshop (ETW)*, 2003. pp. 57-62.

[23] J. Jaffar, and J.-L. Lassez. “Constraint Logic Programming”. *14th ACM Symposium on Principles of Programming Languages (POPL)*, 1987, pp. 111-119.

[24] P.Van Hentenryck, “The CLP language CHIP: constraint solving and applications”. *Compcon Spring '91. Digest of Papers*, 1991, pp. 382 -387.

[25] S. Samii, E. Larsson, K. Chakrabarty, and Z. Peng. “Cycle-Accurate Test Power Modeling and its Application to SoC Test Scheduling”. *IEEE International Test Conference (ITC)*, Oct. 2006. (to appear).

[26] C. Liu, K. Veeraraghavant, and V. Iyengar. “Thermal-aware test scheduling and hot spot temperature minimization for core-based systems”. *IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, 2005, pp. 552-560.

[27] P. M. Rosinger, and B. M. Al-Hashimi. “Thermal-Safe Test Scheduling for Core-Based System-on-Chip Integrated Circuits”. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. (Accepted for future publication).