An Effective Solution to Thermal-Aware Test Scheduling on Network-on-Chip Using Multiple Clock Rates

***Abstract*—As more cores are being deployed on a single chip, bus-based communication is suffering from bandwidth and scalability issues. As a result, the new approach is to use a network on chip (NoC) as the main communication system on a SoC. NoC provides the flexibility and scalability much needed in the era of multi-cores. NoC-based systems also provide the capability of multiple clocking that is widely used in many SoC nowadays. In this paper, a simulated annealing (SA) solution to thermal and power-ware test scheduling of cores in a NoC-based SoC using multiple clock rates is presented. Results on different benchmarks show the effectiveness of our technique.**

I. Introduction

Bus communication is no more able to keep up with the increasing complexity of system-on-chip in the multi-core era. The new paradigm nowadays is to deploy a network-on-chip (NoC) as the main communication system. An NoC provides an attractive solution to the problems brought forth by increasing complexity and size of system on chip. NoC greatly improves the scalability of SoCs, and provides higher power efficiency in complex SoCs compared to buses.

It is widely recognized that testing embedded cores is a major bottleneck. A major challenge in testing embedded cores is test scheduling which determines the order in which various cores are tested. Test scheduling for SoC, even for a simple SoC, is equivalent to the NP-complete m-processor open shop scheduling problem. An effective test scheduling approach must minimize the test time while addressing resource conflicts among cores arising from the use of shared Test Access Mechanisms (TAMs), on-chip BIST engines and power dissipation constraints. Obviously, the minimal test time would be achieved by maximizing the simultaneous test of all cores; however, design constraints may prevent this full parallelism. For example, power consumption is an important factor that may impact the test parallelism. However, power constraints do not necessarily achieve thermal safety of each core in the system. Thermal-constrained test scheduling is therefore essential in order to ensure that the maximum thermal safety temperature is not exceeded by any core.

For a system that contains a NoC as the communication platform, the access to cores is already available and thus the idea of reusing the NoC as the communication platform for testing becomes a natural solution. NoC contains the asset of input/output cores and thus test vectors and the responses can be communicated to and out of the cores under test. Such systems usually use a packet-based scheduling algorithm where the test vectors will be embedded inside packets. Many NoC have the flexibly of multiple clock rates. Not all the cores have to be tested using the same clock rate. A core with long testing time can be assigned a higher clock rate to reduce the test time. Increasing clock rates increases the power consumption of the system cores. What clock rate should be assigned to what core to minimize the overall test time and such that the safety power and thermal constraints are met is one of the main concerns of this paper.

In this paper, the problem of minimizing test scheduling in a system that contains a NoC using multiple clock rates is studied. Also power and thermal constraints are set for the safety and protection of the system under test. The main objective is to minimize the overall test time of the system. Proper allocation of input/output ports in the NoC to cores and the proper allocation of clock rates to different cores are essential in achieving this objective. We present a thermal-aware simulated annealing solution to this problem.

The remainder of the paper is organized as follows. Section II presents related work in this area. Section III presents a brief overview of NoC. Section IV deals with test scheduling using multiple clock rates. Section V presents our thermal-aware simulated annealing solution to the test scheduling problem in a NoC using multiple clock rates. Section VI summarizes the results. Finally, Section VII presents our conclusions.

II. Related Work

Vermeule et al. [1] introduced the general concept of NoC testing. First, the communication NoC infrastructure should be tested before the built-in core test. Testing communication resources are introduced in [2]. Only after the communication resources are tested, NoC can now be used as a TAM to test the on chip cores. Nahvi et al. [3] proposed network-oriented test architecture to utilize NoC for testability. Generally speaking, there are two main approaches to test scheduling algorithms for NoC-based SoC, namely, core-based and packet-based scheduling. In core-based test scheduling, the scheduler assigns a dedicated routing path for each core to transport test vectors and test responses. The basic idea is to determine the test order of each cores so that the overall test time is minimized under different constraints [4].

On the other hand, packet-based scheduling cores are tested using test packets. The basic idea is to determine the order of generation and transmission of test packets for cores. The use of packet switching architecture to test cores is proposed in [5] and [6]. Nahvi et al. [5] proposed the use of a packet switching

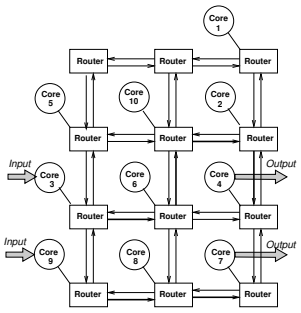

Fig. 1. NoC-based d695 with two routing paths scheduled.

communication-based TAM for system on chip. Cota et al. [7] proposed test scheduling based on a packet-switching protocol with power constraints. Embedded processors have been used for test sources and sinks to increase test parallelism [8]. In [9], [10], multiple test clocks are also used to reduce overall schedule time of NoC-based SoCs.

Liu et. al [4] presented a power-aware testing scheduling algorithm for NoC-based systems. Thermal-ware test scheduling has been studied by many researchers [11], [12], [13].

III. NoC Test Scheduling

The new paradigm in the new SoC testability is to use a NoC. NoC usually uses a 2D mesh topology with wormhole switching. In a mesh structure, each router is connected to the four neighboring routers. The communication channels are assumed to be 32-bit width and bidirectional. Switching is usually based on wormhole technique. Each packet is divided into flow control units (flits) where a flit is the smallest unit over which flow control is performed. The flit size is usually equal the channel width. NoC usually uses a message passing communication model where information is sent via packets. A packet is composed of a header, a payload, and a trailer. Test vectors are delivered through packets. Packets are routed from an input port to the core under test and then the results are routed to an output port.

There are two different approaches to go for the test core scheduling namely, preemptive and non-preemptive. In the preemptive approach, the core does not have to be tested till completion at once. Hence packets can be scheduled in an individual manner. Preemptive techniques can increase parallelism as paths are not dedicated to cores for long periods of times. However, preemption is not always an appealing solution especially in the presence of BIST tests. It is usually desirable that the test pipeline of a certain core is not interrupted. This is hard to achieve in preemptive pipelining as such pipeline has to be interrupted if the test vector or test response cannot be scheduled at this time. Such a problem can complicate the wrapper control as well as the overall test time of the schedule.

Our approach is based on a non-preemptive core-based approach where a path is dedicated to a core from the beginning of the test till the end. In such a case, the test pipeline of the core under test is not interrupted as this path is always available for test vectors and test responses. The dedicated routing path usually consists of an input port, an output port, and corresponding channels that transport test vectors to the core and transport the test responses from the core. Even though packet-based scheduling algorithm is more parallel in nature, the test time of its schedules are usually higher than those of a core-based scheduling technique. Figure 1 shows two routing path schedules for cores 8 and 10 for the d695 example benchmark using two input and two out ports. Each path will be dedicated to the corresponding core throughout the process of testing. The input/output ports in our example are implemented by using the functional ports on cores 3, 4, 7, and 9.

IV. Test Scheduling Using Multiple Clock Rates

In this paper, we propose the use of multiple clock rates for thermal management and test time minimization. One way to reduce the hot spots is to use a slower clock. Using a single slow clock rate is usually inefficient as it may result in a higher testing time for the cores. Hence a single slower clock adversely affects test time and increases cost. Although a slower clock can bring down the hot spots, it is not efficient in achieving thermal balance across the chip.

As a result, we propose to use multiple clock rates instead of just one slower clock. Slower clocks can be generated using a frequency divider. Generally speaking, a slow clock is assigned to hot spots to cool them down. The problem is not that simple. A slow clock rate usually means a higher test time. The problem is to find best clock rates for different spots and a test scheduling to minimize the test time of all the cores such that the core temperatures are below a threshold for thermal safety.

Huang et. al [10] showed that on NoC, some cores cannot utilize the entire width of the network channels. Those idle channels can be used to transport more test data and increase the clock rate for that specific core. The general idea is to use slower clock for high spot cores and faster clocks to cool cores so that the overall thermal safety constraints are met and the test time is minimized. To simplify the problem, we assume that there are $2n + 1$ different on-chip clock frequencies. For example, assume an on-chip clock rate of f and n equals 3 there are a total of 7 clock frequencies, namely, $f/4$, $f/3$, $f/2$, f, $2f$, $3f$, $4f$.

The problem now is to assign the on-chip clocking rates to different cores under test. The overall objective is to minimize the overall test time while not exceeding the maximum core temperature allowed for safety issues. Accurate core temperature is usually best calculated using thermal simulation. In order to obtain the temperature of a core during test, we use a computationally efficient thermal modeling tool called HotSpot [14]. HotSpot accounts for heat dissipation within each functional block as well as the heat flow among different blocks.

V. Thermal-Aware Simulated Annealing

In this section, we present a thermal and power aware simulated annealing (SA) solution to test scheduling on NoC

with multiple clock rates.

Simulated annealing [15] is a global stochastic method that is used to generate approximate solutions to very large combinatorial problems. The algorithm begins with an initial feasible configuration, and then a neighbor configuration is created by perturbing the current solution. If the cost of the neighboring solution is less than that of the current solution, the neighboring solution is accepted; otherwise, it is accepted or rejected with some probability. The probability of accepting inferior solutions is a function of a parameter, called the temperature T, and the change in cost between the neighboring solution and the current solution.

The set of parameters controlling the initial temperature, stopping criterion, temperature decrement between successive stages, and number of iterations for each temperature is called the *cooling schedule*. Typically, at the beginning of the algorithm, the temperature T is large and an inferior solution has a high probability of being accepted. During this period, the algorithm acts as a random search to find a promising region in the solution space. As the optimization progresses, the temperature decreases and there is a lower probability of accepting an inferior solution.

A. Solution Representation and Initial Solution

An example of a random feasible solution to our problem and the corresponding test schedule is presented in Figure 2-(a). This example solution shows a system of 4 cores and two I/O ports and two possible clock rates, f_1 and f_2. To ensure feasibility, the initial solution, in our case, is chosen such that all cores are assigned to the same I/O.

B. Neighborhood Transformation

The main operation of the simulated annealing is the neighborhood function. Starting from a current solution, the neighborhood function applies some operations to move into a new solution. Our neighbor function can perform one of the following operations.

- Change the position of one core.
- Swap the positions of two cores.
- Change the clock of one core.
- Change the I/O of one core.
- Swap the I/O of two cores.

Figure 2 shows an example of exchanging the I/O of cores 2 and 4 and the corresponding schedules for the solution before and after applying such an operation.

C. Cost Function and Cooling Schedule

Given a test schedule, the cost is the maximum end time for the tests of all the cores in our system. The cooling schedule is the set of parameters controlling the initial temperature, the stopping criterion, the temperature decrement between successive stages, and the number of iterations for each temperature. The cooling schedule was empirically determined as follows, T_{init} = 4000, the temperature reduction multiplier, α , is set to 0.99, the number of iterations, M, is set to 5 while the iteration multiplier, β, is set to be 1.5.

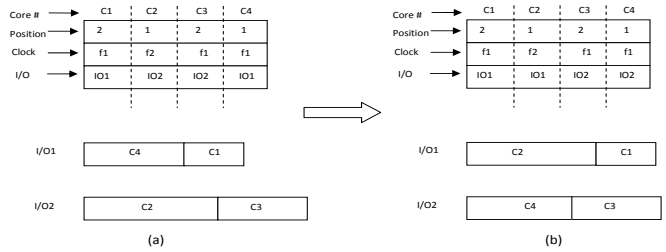

Fig. 2. (a) An example SA solution and its test schedule. (b) The new solution and its test schedule.

Thermal-Aware SA_Test_Scheduling()
N_{iter} = The number of Iteration.
T_{stop} = Stopping temperature .
K = Temperature reduction ratio.
S_{init} = Initial Solution.
$T_{thermal\_safety}$ = Maximum allowable temperature.
P_{max} = Maximum allowable power consumption at the same time.
Boolean *Power*, *Temperature*.
Build S_{init}.
$Cost(S_{init})$ = Test time for S_{init}.
Current solution $S_{curr} = S_{init}$.
While ($T_{curr} > T_{stop}$)
  For (i = 1 to N_{iter}) **do**
    Generate a neighboring solution S_{new} = *Neighbour* (S_{curr})
    Evaluate S_{new}.
    If (S_{new} is feasible) **then**
      If (*Power*)
        For each core i **do**
          Find all cores j such that.
          $t_j \leq t_i$ AND $t_j + l_j > t_j$ /∗Overlap .
          Calculate the sum of power consumption of such cores.
          Find maximum power consumption, P, of all overlapping cores.
          If $P > P_{max}$ **then** Violation.
      If (*Temperature*)
        Invoke *HOTSPOT(Power File, Floorplan File).*
        Find core i with maximum temperature T.
        If $T > T_{thermal\_safety}$ **then** Violation.
      If (No Violation) **then**
        Find test time $Cost(S_{New})$
        Compute Δ_{Cost} = $CostS_{new}$ - $CostS_{curr}$
        If (Δ_{Cost}) < 0) **then**
          $S_{curr} = S_{new}$.
        Else
          Generate random number R, $0 < R < 1$.
          If $R < exp(-\Delta_{Cost}/T_{curr})$ **then**
            $S_{curr} = S_{new}$.
  Update *SA parameters* .

Fig. 3. Thermal-Aware Simulated Annealing for Test Scheduling.

VI. Results

In this section, we present test scheduling results to four benchmarks from ITC'02 [16] namely, $d695$, $g1023$, $p22810$ and $p34392$. We created a hypothetical layout for each NoC system. As the power information of the core in these benchmarks is not available, approximation values based on the number of scan flip-flops of each core is used. Based on these approximated values, the power consumption of each core defined in Equation 1 is adjusted by replacing n_{ff} by the number of scan flip-flops and n_{gt} by an estimated number of gates for the module. In Equation 1, C_L is the load capacitance, T is the clock period, and σ is the switching activity factor. Since it is hard to approximate the power consumption by the routers we only consider power consumption by the cores. Clearly, this number is an approximation of the actual power

TABLE I
TEST TIMES USING SINGLE CLOCK AND MULTIPLE CLOCK RATES.

Benchmark	I/O	Single Clock	Multiple Clocks
d695	2/2	13227	10905
g1023	3/3	14794	7397
p22810	3/3	112793	92554
p34392	3/3	544579	275409

TABLE II
TEST TIMES USING POWER AND THERMAL CONSTRAINTS.

Benchmark	I/O	Power	Max Temp	Temperature
d695	2/2	18786	136	13227
g1023	3/3	16749	203	16473
p22810	3/3	142213	227	112552
p34392	3/3	781834	214	769284

consumption of the modules.

$$Power = C_L \cdot V_{dd}^2 \cdot \frac{1}{T} \cdot [(\sigma_{ff} + 1) \cdot nb_{ff} + \sigma_{gt} \cdot nb_{gt}] \qquad (1)$$

First, we tested our simulated annealing without any power or temperature constraints and assuming there is only one clock rate. Results are shown in the third column of Table I. The channel width is assumed to be 32. Then we repeated our experiments using multiple clock rates. We assume that there are three clock rates $f/2$, f, $2f$. The test times for channel width of 32 are shown in the fourth column in Table I. The results clearly show the importance of using multiple clock rates especially to decrease the test time for long test time cores. On the other hand, faster clocks mean higher power consumption and consequently higher temperature.

Second, we tested our technique based on stringent power constraint where we set the power limit at a certain point not to exceed 25% of total power consumption of all the cores. In Table II, we show the testing time (Column III) and the temperature of the hot spot in our different benchmarks using the 25% strict power constraint. Although the power constraint is very stringent, the hot spot temperature was still way over the acceptable thermal safe temperature of $127^{\circ}C$ (Column 3 of Table II). Decreasing the maximum allowable power further will cause a significant increase in test time and will render some benchmarks unschedulable.

It is clear that power consumption constraint is not sufficient to keep the whole chip safe as the temperature of some cores exceeds the highest allowable temperature even though the total power is still less than the maximum allowable power. In this part of our results, we test our SA with temperature constraint where we set T_{max} to $127^{\circ}C$. Slower clock rates usually result on lower temperature with the downside of longer test times. We first set up the parameters for thermal simulation for the Hotspot tool such as layers, thermal resistance, thickness of layers and materials, and the chip dimensions. The testing time of our thermal safe schedule are shown in Column V of Table II. Our thermal-aware SA solution was able not only to ensure thermal safety but also to decrease the testing time compared to power constraints solution. We are not able to compare to previous works as the corresponding layout, input/output ports, Hotspot parameters and power values are not available.

VII. CONCLUSION

In this paper, we presented a simulated annealing solution to thermal safety test scheduling using multiple clock rates. We showed that stringent power constraints are not sufficient for thermal safety. Results on different benchmarks show the importance of our technique especially as the number of cores built on a single chip is continuously increasing.

REFERENCES

[1] B. Vermeulen, J. Dielissen, K. Goossens, and C. Ciordas, "Bringing communication networks on a chip: Test and verification implications," *IEEE Communication Magazine*, vol. 41, pp. 74–81, 2003.

[2] P. P. Pande, G. D. Micheli, C. Grecu, A. Ivanov, , and R. Saleh, "Design, synthesis, and test of networks on chips," *IEEE Design and Test of Computers*, pp. 404–413, 2005.

[3] M. Nahvi and A. Ivanov, "Indirect test architecture for soc testing," *IEEE Trans. on CAD*, pp. 1128–1142, 2004.

[4] C. Liu, V. Iyengar, J. Shi, and E. Cota, "Power-aware test scheduling in network-on-chip using variable-rate on-chip clocking," in *Proceedings VTS, pp. 349–354*, 2005.

[5] M. Nahvi and A. Ivanov, "A packet switching communication-based test access mechanism for system chips," *IEEE European Test Workshop*, 2001.

[6] C. Aktouf, "A complete strategy for testing an on-chip multiprocessor architecture," *IEEE Design and Test of Computers*, vol. 19(1), pp. 18–28, 2002.

[7] E. Cota, L. Carro, F. Wagner, and M. Lubaszewski, "Power-aware noc reuse on the testing of core-based systems," in *Proceedings ITC, pp. 612–621*, 2003.

[8] A. M. Amory, E. Cota, F. Wagner, L. Carro, M. Lubaszewski, and F. G. Moraes, "Reducing test time with processor reuse in network-on-chip based system," in *Proceedings SBCCI, pp. 111–116*, 2004.

[9] J. Ahn and S. Kang, "Noc-based soc test scheduling using ant colony optimization," *ETRI Journal*, vol. 30, pp. 129–140, 2008.

[10] J. Shao et. al, "Multi-clock domain soc test scheduling based on ant colony optimization algorithm," in *Proc. Fourth International Conference on Internet Computing for Science and Engineering*, 2009.

[11] P. Rosinger, B. Al-Hashimi, and K. Chakrabarty, "Rapid generation of thermal-safe test schedules," in *Proc. Design, Automation and Test in Europe (DATE) Conf.*, 2005, pp. 840–845.

[12] E. Tafaj, P. Rosinger, B. Al-Hashimi, and K. Chakrabarty, "Improving thermal-safe test scheduling for corebased system-on-chip using shift frequency scaling," in *Proc. Int. Symp. DFT*, 2005, pp. 544–551.

[13] Chunsheng Liu and Vikram Iyengar, "Test scheduling with thermal optimization for network-on-chip systems using variabe-rate on-chip clocking," in *Proc. Design, Automation and Test in Europe (DATE) Conf.*, 2006.

[14] W. Huang, S. Ghosh, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: a compact thermal modeling methodology for early-state vlsi design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14(5), pp. 501–513, 2006.

[15] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," vol. 220, no. 4598, pp. 671–680, 1983.

[16] V. Iyengar E. J. Marinissen and K. Chakrabarty, "A set of benchmarks for modular testing of socs," in *Proceedings of IEEE International Test Conference (ITC'02), pp. 519–528*, 2002.