Temperature-Aware Voltage Islands Architecting in System-on-Chip Design

W.-L. Hung, G. M. Link, Yuan Xie, N. Vijaykrishnan, N. Dhanwada†, and J. Conner
Department of CSE, The Pennsylvania State University, University Park, PA 16802
†IBM EDA Laboratory, Hopewell Junction, NY 12533
{whung,link,yuanxie,vijay,jconner}@cse.psu.edu,†nagu@us.ibm.com

Abstract

As technology scales, power consumption and thermal effects have become challenges for system-on-chip designers. The rising on-chip temperatures can have negative impacts on SoC performance, power, and reliability. In view of this, we present a hybrid optimization approach which aims at temperature reduction and hot spot elimination. We demonstrate that considerable improvement in the thermal distribution of a design can be achieved through careful voltage island partitioning, voltage level assignment, and voltage island floorplanning. The experimental results on MCNC benchmarks show significant improvement on the thermal profiles. To the best of our knowledge, this is the first work to explore the thermal impacts of voltage islands.

1. Introduction

Aggressive scaling of process technologies has enabled designers to pack more functionality onto a single die. The higher levels of integration due to scaling, along with the advent of highly complex re-usable IP (Intellectual Property) blocks has spurred an increase in core-based SoC (System-on-Chip) design techniques. However, the increased level of integration within a single die imposes rigid constraints on the power consumption budget. Among various approaches proposed to reduce the power consumption, the use of multiple voltage islands [1] is one of the most attractive approaches for core-based SOC designs.

The use of multiple voltage islands exploits the concept of using lower supply voltages for parts of the design that are not in the critical path to reduce both dynamic and leakage power. In order to alleviate the cost of supplying multiple voltages to different parts of the chip and the cost of level conversion when communicating across different voltage levels, the voltage island approach clusters a group of cores operating at the same voltage and provides one single voltage level for all modules inside this island.

Even though the voltage island technique can help mitigate power problems, accordingly, it complicates the chip design process in terms of power routing, floorplanning, and timing closure with the additional overheads of level converters with respect to area and delay. Thus, how to effectively group the compatible cores together with the same supply voltage without disturbing other design metrics such as wire length and critical path timing is a crucial issue.

Figure 1 shows an example of a SoC design with two voltage islands. Each module has a list of operating voltages from which it can choose. For example, module b_6 can operate at 1.2, 1.3 or 1.4 volt. The chip level supply voltage is 1.4 volt, so there is no need of level converter for modules b_1, b_2, and b_3. Level converters, however, are needed for islands A and B in order to communicate with components in other islands or cores operating at chip-level supply voltage. Typically, the modules may be soft or hard, free to rotate, and also there may exist some constraints associated with the SoC design, such as boundary constraints, range constraints, and/or performance constraints.

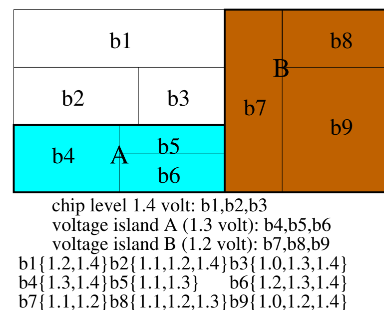

Figure 1. An example of two voltage islands.

To minimize the total power consumption of a voltage island-based SoC, the most intuitive way is to assign each core with the lowest voltage from its supply voltage list. However, it may not be feasible in practice. For example, in Figure 1, if we architect the voltage island by combining module b_3 and b_9 and assigning both of them to a single voltage island, it will result in large waste of dead space and may increase the wire length during floorplanning process. Moreover, there will be totally four voltage islands in this configuration, resulting in need of larger numbers of level converters compared to the partition in Figure 1, which has

Proceedings of the 2005 International Conference on Computer Design (ICCD'05)
0-7695-2451-6/05 $20.00 © 2005 IEEE
IEEE COMPUTER SOCIETY

only two voltage islands. Furthermore, it is possible that some timing critical cores might have timing violation when operated with certain voltage supplies.

While incorporating the concept of voltage islands can assist at reducing the impact of the escalating power problem in SoC design, thermal problems are not yet well studied. Process scaling and aggressive performance improvements have caused a dramatic power consumption increase. Power density directly translates into heat; as a result, the temperature in modern high-performance VLSI circuits increases dramatically due to smaller feature size, higher packing density, and rising power consumption. These high temperatures can have significant impacts on circuit behavior. First, MOS current drive capability decreases approximately 4% for every 10^oC temperature increase, and interconnect (Elmore) delay increases approximately 5% for every 10^oC increase [5]. Second, leakage power can be orders of magnitude greater at high temperature [3]. Last, but not least, reliability is strongly related to temperature [9]. A first order model for the impact of temperature on reliability is the $Arrhenius$ equation: $MTF = MTF_o exp(-E_a/k_b * T)$, where T is operating temperature.

The use of voltage islands [1, 14, 17] has been shown to be very effective in reducing active and static power consumption. Hu $et\ al.$ [8] explored the problem of voltage island partitioning and floorplanning with an objective of lowering the total power across a chip while trying to maintain a valid core-based supply voltage partition. However, none of the existing works on voltage island assignment have focused on thermal issues. Floorplanning algorithms that only consider chip area and power consumption are not sufficient for such problems because the thermal profile depends on not only the power density but also on the physical size and location of each functional core [5].

Based on the facts above, the additional imposed constraints make thermal-aware voltage island partitioning and floorplanning a unique and interesting problem. In this paper, we present a hybrid optimization approach which targets peak temperature reduction and elimination of hot spots. We demonstrate that by carefully partitioning the cores, assigning voltage levels, and allocating the islands, the thermal distribution of the design can be improved. Our approach is the first to explore the thermal issue of voltage islands and attempt to reduce the peak temperature and to even out the temperatures under this chip configuration.

The rest of the paper is organized as follows. Section 2 reviews previous work related to this research. Section 3 formulates the problem and Section 4 explains how to estimate temperature. Section 5 presents our thermal-aware voltage island partitioning and floorplanning framework based on a hybrid optimization approach. Section 6 presents and discusses experimental results. We conclude this paper in the last section.

2. Related work

In order to facilitate the thermal-aware floorplanning process, a compact thermal model is needed to provide the temperature profile. Numerical computing methods (such as finite difference method FDM [5] and FEM) are most accurate but computational intensive, while the simplified close-form formula [11] is the fastest but inaccurate. Skadron $et\ al.$ proposed a thermal model in [10], which is based on lumped thermal resistances and thermal capacitances. It is more efficient since the temperature is tracked at the granularity of functional block level.

There are several existing works on thermal-aware placement for standard cell ASIC designs. For example, Chu and Wong proposed using a matrix synthesis problem (MSP) to model the thermal placement problem. Three algorithms were proposed to even out the heat dissipation of the circuit [4]. Tsai and Kang [5] proposed a compact FDM-based temperature model to derive temperature, based on which the standard cell placement and macro cell placement were tackled by a simulated annealing based thermal-driven placement algorithm. Thermal placement can also be refined by partitioning; for example, Chen $et\ al.$ proposed a partition-driven thermal placement model [6] for standard cells, making use of multigrid-like approach to facilitate the inclusion of temperature constraints on the placement. Goplen and Sapatnekar explored the thermal placement for standard cells with force directed approach in [7] by formulating temperature as another force. Hung $et\ al.$ [16] proposed the thermal-aware floorplanning by using genetic algorithms, but wire length factor is not included in their cost function evaluation. However, these thermal-aware standard cell placement techniques cannot be applied directly to the targeted problem because the cores in voltage island domain are typically of a non-uniform size; in addition, the voltage island partitioning problem also prevents us from using traditional thermal-aware placers.

3. Problem formulation

We assume that the SoC design contains a set of IP modules. Let $B = b_1, b_2, ..., b_m$ be the set of rectangular IP modules embedded on the SoC and each of which has width w_i, height h_i, and area a_i associated with it. Different IP modules might have different supply voltage levels, so the related supporting supply voltages and its corresponding average power consumptions are also provided. A voltage island partition and floorplan is a mapping for each block b_i into distinct voltage islands such that no IP modules fall into two islands. The goal of thermal-aware voltage island partitioning and floorplanning is to minimize both peak and average temperatures across the SoC system and other design metrics, such as area, wire length, and power budget, which can also be imposed in the evaluation process.

IEEE COMPUTER SOCIETY

4. Temperature estimation

Skadron $et\ al.$ [10] proposed a thermal modeling tool called HotSpot, which employs the principle of thermal-electrical duality to allow for a computationally efficient model of thermal effects at the block level. HotSpot provides a simple compact model, where the heat dissipation within each functional core and the heat flow among cores is accounted for. The basic idea is that, if we define the transfer thermal resistance R_{ij} of functional module m_i with respect to module m_j as the temperature rise at module m_i due to one unit of power dissipated at module m_j. Thus, R_{ij} can be written as

$$R_{ij} = \triangle T_{ij} / \triangle P_j \qquad (1)$$

Using the similar method, we can get a transfer thermal resistance matrix for the entire device. The determination of resulting temperatures is similar in nature to the determination of the voltages in a resistance network, where heat sources are represented as current sources. The actual operations are performed in matrix format, where the full thermal resistance matrix is multiplied by a vector containing the power dissipation of each IP module. This allows the temperature of the SoC to be tracked on a per-module level.

While the HotSpot tool was originally intended to be a fast means of modeling temperature, as the number of IP modules grows, the runtime increases dramatically. While this may be tolerable for a single floorplan, when analyzing hundreds or millions of possible floorplans as in an annealing procedure, the runtime penalty can easily exceed the runtime of the actual floorplanning algorithm itself. To combat this performance overhead, the thermal model in HotSpot was re-implemented using a number of performance optimizations. In particular, all operations have been optimized to occur in a single large memory bank that requires only one allocation and deallocation. The operations in this bank are located in positions that optimize cache locality, and are aligned with the address space of the system. Rather than performing expensive matrix inversion and multiplication operations, our optimized thermal modeling tool employs the LAPACK package [15] to determine the final temperatures, resulting in significant additional speedup. Finally, the thermal model was profiled at the assembly level to determine problematic sections of the code, such as data stalls incurred due to dependencies in loops. These sections were then hand-optimized with techniques such as loop unrolling to minimize these problems.

5. Voltage island partitioning and floorplanning algorithm

The voltage island partition and floorplan framework is composed of two parts: a genetic algorithm (GA) based voltage island partitioning algorithm and a simulated annealing (SA) based floorplanning algorithm. GA's [13] are

```
Algorithm
Begin
    Given modules' information and the power values;
    GA is applied to generate the initial population;
    While (the termination criteria is not met)
        For each chromosome in the population
            For each voltage island encoded in chromosome
                *Use SA to floorplan the current voltage
                    island partition;
            EndFor
        EndFor
        **Use SA to floorplan all voltage islands at the
            chip level;
        GA fitness function evaluation;
        Apply three operators(selection, crossover, and
            mutation) to produce new chromosomes;
    EndWhile
End
```

Figure 2. Outline of the voltage island partitioningg and floorplanning algorithm.

a class of search and optimization methods that mimic the evolutionary principles in natural selection and have been used actively in recent VLSI optimization problems. In GA's, the voltage partition solution is encoded into an integer string called a chromosome. Instead of working with a single solution as in the simulated annealing algorithm, the GA search process begins with a random set of chromosomes called initial population. Each chromosome is assigned a fitness score that is directly related to the objective function of the optimization problem. The population of chromosomes is modified to a new generation by applying three operators similar to natural selection operators - selection, crossover and mutation.

As shown in Figure 2, the initial modules' information and their relative power values are given as the inputs to the proposed algorithm. The GA first generates a population of random voltage partitions, and then each valid voltage partition contained in the chromosome is evaluated. That is, each voltage island involves another floorplanning step. The simulated annealing based floorplanner is activated to iteratively improve the quality of the floorplanning solution, which is done by the solution perturbation. The cost function used by a SA-based floorplanner can be area minimization, balancing thermal profile of a chip, or both. More details about balancing the thermal profiles will be given in the following section. For the goal of area minimization, it will not only help reduce the dead space, but also implicitly reduce the wire length of connections between modules due to the resulting tight area compaction. After floorplanning each voltage island, a chip-level floorplanning is performed with a weighted cost of the optimized objectives and the constraints. The cost function can be written as

$$cost = \alpha * C_{area} + (1 - \alpha) * C_{wire} + \beta * C_{temp} \qquad (2)$$

where C_{area} and C_{wire} represent the chip area and wire length, while C_{temp} indicates the chip temperature. The fitness function is thus assigned to be the reciprocal of this cost function (i.e. fitness = 1/cost). Other factors in the de-

IEEE COMPUTER SOCIETY

sign process can be easily incorporated into our approach by adding more design variables and adjusting the relative weights. Note that the voltage partition should always be $valid$ and $compatible$, which is taken care of in GA before we pass the modules' information to the SA-based floorplanner. After assigning the fitness value to the current island partitions represented by a chromosome, the new chromosomes are those which likely have higher fitness than the old ones. That is, the voltage island partitions having lower area and total wire length. Thus, this process is oriented toward the optimal solution.

In order to generate a new chromosome (voltage island partition), three operations are applied and are listed in the following:

(1) Selection, which is done by stochastic $roulette-wheel$ approach;
(2) Crossover, which is done by single point crossover and is illustrated in Figure 3;
(3) Mutation, which is accomplished by the swap operation.

Usually, the crossover rate is higher than mutation rate, so that the evolution will not become a random search algorithm.

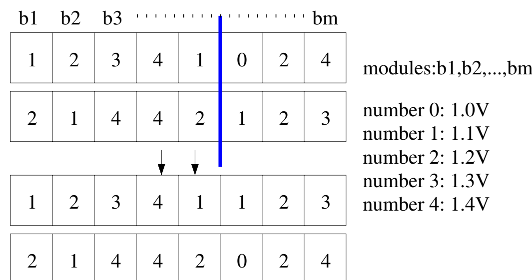

Figure 3. Example of single point crossover in voltage island partitioning.

5.1. Slicing tree floorplan model

The slicing tree structure [2] is used as the floorplan model and is incorporated with the simulated annealing algorithm [12]. Note that other floorplanners can be adopted in our approach. The four operations used in perturbing a slicing floorplan are listed below:

(1) Rotation, which rotates a module;
(2) Leaf node swap, which swaps two modules;
(3) Flip, which flips a slicing operator;
(4) Sub-tree swap, which swaps two sub-trees in a slicing tree.

A slicing floorplan is a rectangular area that is sliced recursively by a horizontal or vertical slicing line (operator) into a set of rectangular regions to accommodate a set of functional modules. A slicing floorplan can also be represented by $Polish$ expression. Figure 4 shows an example of slicing tree, the polish expression, and its corresponding floorplan. The slicing tree is a binary tree constructed in a bottom-up fashion, where each internal node in the binary tree contains the size and coordinate of the enclosing rectangle and those of two children. This tree structure gives the advantage of fast extracting the coordinates of the dead spaces, used in the calculation of the chip temperature profile.

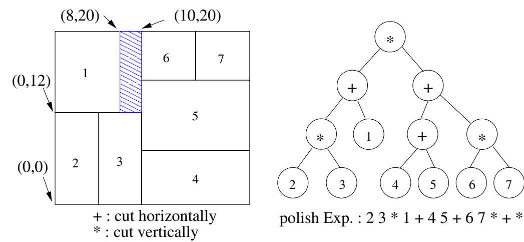

Figure 4. Slicing tree representation and its corresponding floorplan.

The procedure of dead space coordinate calculation is as follows: After constructing the slicing tree, the internal node contains the width and height of the enclosing rectangle where its left and right children are two modules combined together either by vertical or horizontal slicing operator. For example, modules 2 and 3 in Figure 4 are combined first by a vertical slicing operator and then this combined one is horizontal sliced to module 1. Since the width of module 1 is smaller than that of the combined one, an unavoidable dead space is generated. The bottom-left coordinate of this dead space (marked by dark color) is obtained by comparing the width of module 1 and that of combined one and is thus 8 for coordinate x, while y coordinate is 12. To ensure accurate thermal modeling, these dead spaces are passed as inputs to the thermal model, allowing heat to flow across and through them as it would in an actual chip.

5.2. Thermal-aware floorplanning

Traditional simulated annealing based floorplanner only considers area and wire length minimizations (i.e. only considers the first two terms in equation (2)). If two hottest blocks are adjacent to each other, the temperature is higher than the case when they are placed far away from each other. In order to take into account the temperature impact in the floorplanning process, a thermal-aware SA floorplanner should use the same cost function as equation (2).

Our thermal-aware floorplanning can be conducted using two different approaches: the first one is called chip-level approach (which only takes temperature into consideration at chip level), and the second one is called two-level approach (which considers temperature at both chip level and voltage island level). The main objective of both approaches is to distribute temperatures across a chip evenly and minimize the hot spot temperatures by using the thermal-aware floorplanning.

The chip-level approach uses a traditional SA floorplanner (minimizing area and wire length) in the voltage island

level floorplanning (line $*$ of Figure 2). The thermal optimization is performed at chip-level where the voltage islands are placed simultaneously using a thermal-aware SA floorplanner (line $**$ of Figure 2).

The two-level approach performs thermal optimization at both chip level and voltage island level. In this approach, both voltage island level floorplanning (line $*$ of Figure 2) and chip level floorplanning (line $**$ of Figure 2) are performed using a thermal-aware SA floorplanner. Compared to the chip-level approach (in which the voltage island level floorplanning is guided by traditional area and wire length metrics), this approach incurs longer run-time, because it needs to call temperature estimation tool at the voltage island floorplanning level, while the other approach only invokes temperature estimation at chip-level floorplanning. However, this approach may achieve better thermal reduction because temperature is considered at a finer granularity. Since an enormous number of configurations will be evaluated during the thermal aware SA-based floorplanning process, the fast retrieval of temperature profiles is necessary, which is the reason why we speed up the HotSpot computation (See section 4).

6. Experimental results

We implemented the proposed thermal-aware voltage island partitioning and floorplanning algorithm in C++. The thermal model and tool are based on the enhanced HotSpot source code. The experiment was run on a dual Intel Xeon (3.2 GHz, 2GB RAM) machine running Linux. We applied our algorithm to conduct experiments on a set of MCNC benchmarks. For voltage island setting, there are a total of five voltage levels available which range from 1.0V to 1.4V, with the chip-level supply voltage assigned as 1.4V. For instance, b_2 can be operated at 1.1V, 1.2V, 1.4V, while b_4 can be operated at 1.3V, 1.4V. We also assume the power consumption of the level converter is negligible. Although the number of voltage islands can be constrained in our framework, we let the algorithm to handle the voltage level assignment in an attempt to achieve overall temperature minimization. We use an approach similar to the one in [5] to assign the average power density for each IP module in the range of $2.2*10^4$ (W/m^2) and $2.4*10^6$ (W/m^2). The net length is estimated by using the general half-perimeter bounding box model.

6.1. Experiment on chip-level

Table 1 summarizes the experiments on three different approaches, area+wire (AW), power+area+wire (PAW), and temperature+area+wire (TAW) optimizations. Time is reported in minutes. The waste area columns below each optimization indicate the ratio of total dead space to the cell area. As can be seen from the table, by including the thermal effect into the traditional design metrics, area and wire length, the peak and average temperatures are effectively reduced. The average maximal and average temperature reductions from applying TAW, compared with AW, can be up to 26°C and 9°C, respectively, while with PAW are 11°C and 1.9^{o}C. The PAW approach used more voltage islands than the other two approaches in seeking a low power solution. Although power is also an important factor affecting the temperature distribution, without the physical location information of each module, the PAW optimization can not capture the thermal coupling effects as its counterpart, TAW optimization. Figure 5 shows the floorplanning result of ami49 partitioned into three voltage islands.

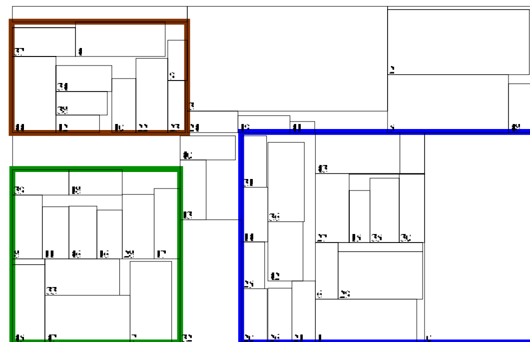

Figure 5. Partitioning and floorplanning result of ami49 under TAW optimization.

6.2. Experiment on two-level thermal-aware optimization

Table 2 shows the result of experiments when considering thermal effect in both chip level and island level. Comparing the best results in Table 1, the two-level approach can reduce the peak and average temperatures further by up to 5°C and 6°C, respectively. The floorplanning with thermal profile inside the voltage island is helpful in separating the hot IP modules away from each other.

Figures 6 and 7 show the distributions of temperatures across a chip. From Figure 7, we see the number of IP modules with temperatures in ranges of 80$\sim$85^{o}C and 85$\sim$90^{o}C are reduced compared to those in Figure 6, while the number of IP modules in the range of 75$\sim$80°C increases. The peak temperature also dropped from 96°C to 92°C. This redistribution of temperatures is beneficial in reducing both the peak and average temperatures with the reason of decreasing effects of hot core coupling. However, the area and wire length also increase accordingly, albeit slightly, due to the finer level of thermal-aware floorplanning. Notably, the runtime penalty for including thermal optimization is small, with an average performance penalty of only 17 percent over the benchmarks tested. Even for larger benchmarks, such as ami49, obtaining temperature values requires only 10ms. Without the optimized thermal tool, the performance penalty is 107%. As such, the inclusion of temperature opti-

IEEE COMPUTER SOCIETY

Table 1. Comparison of temperatures under different design metrics.

MCNC Benchmarks			Area and wire (**AW**)						Power-aware with area and wire (**PAW**)						Thermal-aware with area and wire (**TAW**)					
Circuit	# cells	Cell area	maxT	avgT	wire	waste area	# vi	time	maxT	avgT	wire	waste area	# vi	time	maxT	avgT	wire	waste area	# vi	time
$apte$	9	45.56	110.46	71.79	499	4.1%	3	11	87.95	68.18	500	10.5%	3	12	80.67	66.93	580	4.6%	3	14
$xerox$	10	19.35	116.67	84.58	450	9.1%	3	15	97.09	79.01	511	12.7%	3	16	86.65	78.47	510	14.7%	2	18
hp	11	8.83	139.80	113.74	168	8.9%	3	17	118.43	97.81	170	15.5%	4	19	98.77	95.41	203	15.1%	3	22
$ami33$	33	1.15	112.62	111.10	87	13.8%	4	35	103.37	102.02	87	17.3%	4	38	99.58	97.50	95	19.3%	4	38
$ami49$	49	35.44	113.02	87.79	1545	16.6%	4	49	112.46	85.12	1606	24.2%	4	51	96.03	84.50	1569	19.1%	3	51

mization at the floorplanning stage is a low-overhead means of improving resulting designs.

Table 2. Results of two-level thermal-aware approach.

Two-level thermal-aware with area and wire (**TTAW**)						
Circuit	maxT	avgT	wire	waste area	# vi	time
$apte$	77.29	60.97	803	5%	3	20
$xerox$	81.49	75.17	874	14.7%	2	27
hp	97.79	93.59	274	19.7%	4	29
$ami33$	97.01	95.21	101	22.1%	4	46
$ami49$	91.67	82.18	2112	20.3%	3	59

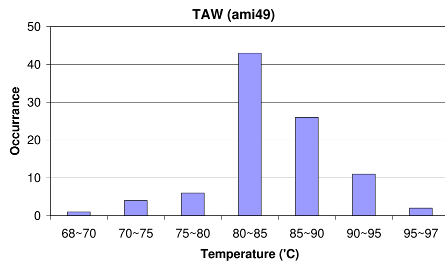

Figure 6. On-chip temperature distribution for ami49 with TAW optimization.

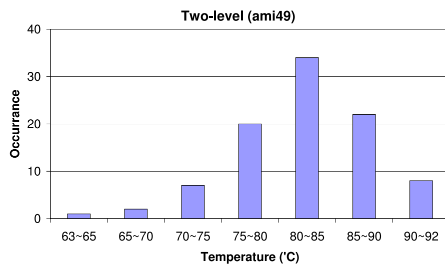

Figure 7. On-chip temperature distribution for ami49 with two-level optimization.

7. Conclusion

In this paper, we present a thermal-aware approach to address the voltage island partition and floorplan problem used in a SoC. Our algorithm is based on a hybrid optimization approach, GA-based partitioning and SA-based floorplanning, which is effective in lowering the hot spot and average temperatures; thus achieves a thermally balanced island-based SoC design.

8. Acknowledgments

This works was supported in part by a MARCO/DARPA GSRC Award, PDG/SoC, NSF Awards CAREER 0130143, 0093085, and 0202007.

References

[1] D. E. Lackey P. S. Zuchowski, and T. R. Bednar. Managing power and performance for System-on-Chip designs using voltage islands. In *ICCAD*, 2002.

[2] D. F. Wong and C. L. Liu. New Algorithm for Floorplan Design. In *DAC*, 1986.

[3] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand. Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. In *Proc. of the IEEE, 91(2)*, pages 305–237, Feb 2003.

[4] C.N. Chu and D.F. Wong. Mtrix Synthesis Approach to Thermal Placement. In *ISPD*, 1997.

[5] Ching-Han Tsai and Sung-Mo Kang. Cell-Level Placement for Improving Substrate Thermal Distribution. In *IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pages 253–266, 2000.

[6] Guoqiang Chen and Sachin Sapatnekar. Partition-Driven Standard Cell Thermal Placement. In *ISPD*, 2003.

[7] Brent Goplen and Sachin Sapatnekar. Efficient Thermal Placement of Standard Cells in 3D ICs using a Force Directed Approach. In *ICCAD*, 2003.

[8] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu. Architecting Voltage Islands in Core-based System-on-a-Chip Designs. In *ISLPED*, 2004.

[9] W. Huang, M R. Stan, K. Skadron, and K. Sankaranarayanan. Cmpact Thermal Modeling for Temperature-Aware Design. In *DAC*, 2004.

[10] K. Skadron, T. Abdelzaher, and M. R. Stan. Control-Theoretic Techniques and Thermal-RC Modeling for Accurate and Localized Dynamic Thermal Management. In *HPCA*, 2002.

[11] T. Y. Chiang, S. J. Souri, C. O. Chui, and K. C. Saraswat. Thermal Analysis of Heterogeneous 3D ICs with Various Integration Scenarious. In *Tech. Digest IEDM*, pages 681–684, 2001.

[12] S. Kirkpatrick, D. D. Gelatt, and M. P. Vecchi. Otimizations by Simulated Annealing. In *Science*, vol.220, no.4598, May 1983, pp.671-680.

[13] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, NewYork, 1989.

[14] J.-A. Carballo, J. L. Burms, S.-M. Yoo, I. Vo, and V. R. Norman A Semi-Custom Voltage-Island Techinque and Its Application to High-Speed Serial Links. In *ISLPED*, 2003.

[15] The LAPACK project. LAPACK - Linear Algebra PACKage. http://www.netlib.org/lapack

[16] W.-L. Hung, Y. Xie, N. Vijaykrishnan, C. Addo-Quaye, T. Theocharides, and M. J. Irwin Thermal-Aware Floorplanning Using Genetic Algorithms. In *ISQED*, 2005.

[17] Power Plan Gold, http://www.ggtcorp.com

IEEE COMPUTER SOCIETY