Multiobjective Microarchitectural Floorplanning for 2-D and 3-D ICs

Michael Healy, *Student Member, IEEE,* Mario Vittes, *Student Member, IEEE,* Mongkol Ekpanyapong, *Student Member, IEEE,* Chinnakrishnan S. Ballapuram, Sung Kyu Lim, *Senior Member, IEEE,* Hsien-Hsin S. Lee, *Member, IEEE,* and Gabriel H. Loh, *Member, IEEE*

***Abstract*—This paper presents the first multiobjective microarchitectural floorplanning algorithm for high-performance processors implemented in two-dimensional (2-D) and three-dimensional (3-D) ICs. The floorplanner takes a microarchitectural netlist and determines the dimension as well as the placement of the functional modules into single- or multiple-device layers while simultaneously achieving high performance and thermal reliability. The traditional design objectives such as area and wirelength are also considered. The 3-D floorplanning algorithm considers the following 3-D-specific issues: vertical overlap optimization and bonding-aware layer partitioning. The hybrid floorplanning approach combines linear programming and simulated annealing, which is shown to be very effective in obtaining high-quality solutions in a short runtime under multiobjective goals. This paper provides comprehensive experimental results on making tradeoffs among performance, thermal, area, and wirelength for both 2-D and 3-D ICs.**

***Index Terms*—Microarchitectural floorplanning, performance optimization, thermal distribution, three-dimensional integrated circuits (3-D ICs).**

I. Introduction

FUTURE processors implemented in deep submicrometer technologies will spend more time in communicating data operands or exchanging control information than actually performing useful computation. Meanwhile, the impacts of power and thermal densities on these deep submicrometer devices and interconnects continue to increase, thereby raising the cost for cooling solutions, eroding performance gains, and threatening the overall circuit reliability. Microarchitectural floorplanning has recently drawn significant interest from both the computer architecture and the electronic design automation communities [1]–[5]. The main motivation is to tackle the ever-worsening wire delay problem of high-performance processors [6], [7] with a collaborative effort between microarchitecture and physical computer-aided design.

The three-dimensional (3-D) IC is an emerging technology that vertically stacks multiple dies with a die-to-die interconnect, as illustrated in Fig. 1. The die-to-die via pitch is very small and provides the possibility of arranging digital functional unit blocks across multiple dies at a very fine level of granularity. This results in a decrease in the overall wire length, which translates into less wire delay and less power. Thus, 3-D ICs can address the wire delay problem effectively by replacing the long and slow global interconnects with short and fast vertical routes. Advances in 3-D integration and packaging are undoubtedly gaining momentum and have become of critical interest to the semiconductor community. These 3-D IC and package manufacturing technologies are rapidly being adopted by several leading companies for commercial applications.

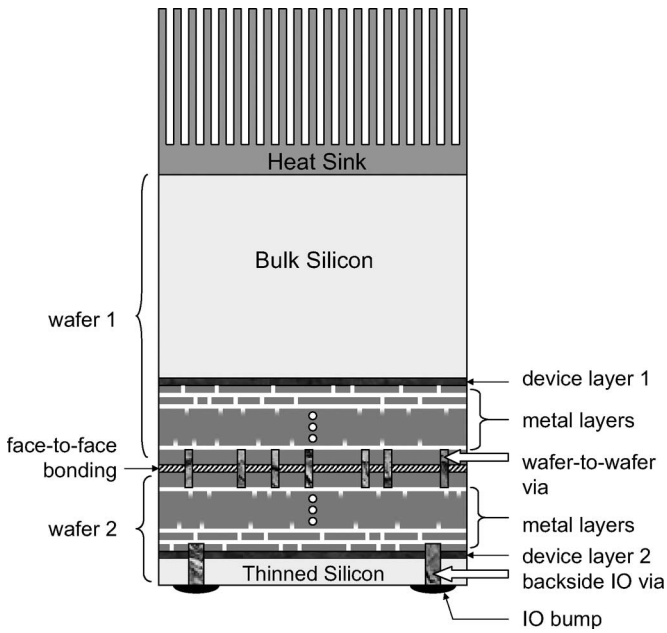

Fig. 1. Two-die 3-D IC with F2F bonding.

The location of individual microarchitectural modules plays a significant role on many important metrics. First, floorplanning has a huge impact on the performance of a given microarchitecture [measured by instructions per cycle (IPC)] as the global interconnects between modules are likely to be pipelined in order to meet high target clock frequencies. This may increase or decrease the access latency on all intermodule interconnects. Second, the thermal and leakage profile is highly correlated with the floorplan. This is because the temperature of each microarchitectural module is dependent not only on the heat generation rate of each individual module but also on the

Manuscript received December 11, 2005; revised March 24, 2006 and May 6, 2006. This work was supported in part by MARCO GSRC/C2S2 and in part by a National Science Foundation CAREER Award under Grant CCF-0546382. This paper was recommended by Associate Editor S. Sapatnekar.

M. Healy, M. Ekpanyapong, C. S. Ballapuram, S. K. Lim, and H.-H. S. Lee are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: limsk@ece.gatech.edu).

M. Vittes is with Intel Corporation, Santa Clara, CA 95052 USA.

G. H. Loh is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA.

Digital Object Identifier 10.1109/TCAD.2006.883925

heat coupling between its neighboring modules. Moreover, the leakage power of each transistor is exponentially proportional to the temperature. Third, floorplanning affects the dynamic power consumption of the buses and clock distribution network. The total number of flip-flops (FFs) inserted on global interconnects changes the dynamic power consumed by the clock distribution network. However, the performance and thermal objectives are conflicting with each other since the shorter distance among hot modules improves the performance while exacerbating the thermal issue. To address the different design constraints of different domains, we need a goal-directed automated floorplanner that allows users to weigh their own design requirements and make effective design tradeoffs. The contributions of this paper are as follows.

1) This paper proposes the first multiobjective floorplanner for deep submicrometer processors at the "microarchitectural level." In addition, microarchitectural floorplanning for 3-D ICs has never been investigated before to the best of our knowledge. Our two-dimensional (2-D)/3-D floorplanners simultaneously consider performance, thermal reliability, footprint area, and interconnect length objectives, providing various tradeoff points.
2) Our microarchitectural thermal modeling considers the thermal and leakage interdependence for effective thermal runaway avoidance. Our microarchitectural power analysis, integrated with our thermal analyzer, models the dynamic and leakage power consumed by functional modules, global interconnects, and the clock distribution network for higher modeling accuracy.
3) This paper provides in-depth discussions along with effective solutions for the following important 3-D-specific problem: vertical overlap optimization and bonding-style-aware layer partitioning. We show how the vertical overlap among modules in 3-D floorplanning affects performance, thermal, and area objectives. In addition, we discuss how layer partitioning is done under different interdie via requirements existing in face-to-face (F2F), face-to-back (F2B), and back-to-back (B2B) bonding in 3-D stacked ICs.
4) Our floorplanning optimizer consists of two steps, namely: 1) initial solution construction via linear programming (LP) and 2) stochastic refinement via simulated annealing (SA). This hybrid approach proves to be very effective in obtaining high-quality solutions in a short runtime.

The remainder of this paper is organized as follows. Section II discusses existing works. Section III presents our architecture model as well as the thermal and leakage simulators. Section IV presents our multiobjective 2-D floorplanner. Section V discusses the 3-D extension of our 2-D floorplanner. Experimental results are shown in Section VI, and we conclude in Section VII.

II. Related Work

Recent studies have focused on traditional 2-D microarchitectural floorplanning for performance optimization but not thermal concerns [1]–[5]. For example, Nookala *et al.* [5] use a statistical design of experiments to approximate the effect on IPC of various wire lengths and then use this approximation during SA to improve performance. Several microarchitecture research works on thermal [8]–[10] and leakage power [11]–[15] provide runtime management of the functional modules but do not perform floorplanning. In [15], the most recently published, they present a system level leakage power model and discuss dynamic management to reduce the thermal problem, as well as discussing thermal runaway and showing that a dynamic management scheme must include consideration of leakage power to be effective. Most existing floorplanning and placement work on thermal [16]–[22] target circuit designs and not on microarchitectural designs. For example, Cong *et al.* [22] present a 3-D temperature-driven floorplanner based on transitive closure graph and a novel bucket structure to represent module overlap. They use various thermal analyzers to trade off runtime with accuracy and overall performance. In addition, recently developed physical design tools for 3-D ICs [21]–[35] target gate-level netlists, are inefficient, and not suitable for evaluating different microarchitecture options during the early design stage. Thus, this paper is the first to simultaneously consider performance, thermal, and leakage for the automated floorplanning of an entire processor microarchitecture with full simulation of the results of floorplanning.

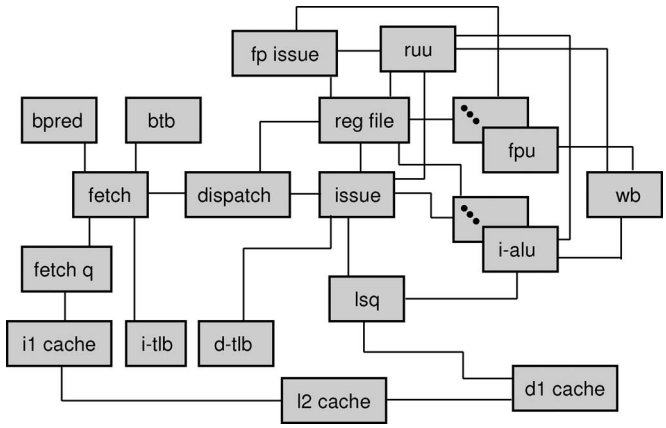

Fig. 2. Processor microarchitecture model.

III. Simulation Infrastructure

A. Microarchitectural Model

The microarchitecture used in our experiment is illustrated in Fig. 2. Each block represents a microarchitectural module used by our floorplanner. In order to model performance more faithfully for deep submicrometer processors, we isolate and model each wire as a separate "resource" that consumes power and has a delay in proportion to its length. Note that architectural simulators that ignore intermodule communication latencies will no longer be useful for evaluating high-frequency processors designed with deep submicrometer technologies due to wire delays, floorplan constraints, and thermal concerns. Essentially, the intermodule latency is a function of the distance and the number of FFs between modules and must be taken into account in both performance evaluation and floorplanning. For this reason, we use the distances generated by the floorplanner to

determine the latency-related parameters such as pipeline depth and communication/forwarding latencies for our performance simulation.

The microarchitectural configuration used in our study[1] is summarized as follows: The machine width is eight. We use a 1024-entry gshare branch predictor, a 512-entry register update unit (RUU) [36] that combines the functionality of a reservation station and a reorder buffer, 16-KB instruction and data L1 caches, a 256-KB unified L2 cache and no L3 cache, 128-entry instruction and data TLBs, eight ALUs, four FPUs, and a 64-entry load store queue.

B. Dynamic Power Modeling

While collecting the intermodule traffic, we also generate the power consumption profile for each microarchitectural module cumulatively for every hundred thousand cycles. The rationale for such sampling is that the temperature is very unlikely to elevate abruptly within a processor's operation period of a few hundred thousand cycles. Note that these detailed traffic activity and dynamic power profiles are only collected once at the very beginning of the entire design flow. The thermal analyzer then uses these power statistics to provide the thermal profile. The floorplanner generates a new floorplan for the given thermal profile and module netlist.

We assume that the intramodule dynamic power consumption remains the same for different floorplans as the module activity factors primarily depend on the program behavior rather than the relative positions. Since the new floorplan may lead to different interconnect lengths between modules, our tool recomputes all of the intermodule interconnect power based on the new lengths and adds it to the dynamic per-module power collected earlier.

The number of FFs inserted on the wires for an extremely high clock frequency can create a larger load on the clock distribution network. This combined with the increasing percentage of the power budget that the clock distribution network consumes necessitates modeling the clock power at a finer granularity. Toward this, we use the accurate clock power model from [37]. This model considers clock distribution network power for memory structure precharge arrays, distribution wiring and drivers, pipeline FFs, and the phase-locked loop.

C. Leakage Power Modeling

Leakage power is modeled in a separate process within our design flow. The model based on [38] considers different bias conditions, although it only estimates subthreshold leakage power. For array-like structures, such as caches and TLBs, the number of bits (or SRAM cells) stored is multiplied by the amount of leakage current per bit and by the supply voltage to calculate the total leakage power for the structure. To calibrate our model, we also calculate the subthreshold leakage currents using the method in eCACTI [39]. Our model closely matches the leakage power estimated from eCACTI. For logic structures, we assume CMOS gates where half of the transistors are leaking at any given time. The number of transistors in these structures is estimated using the area values from GENESYS [40].

The following equation shows the relation between the subthreshold leakage current $I_{\rm sub}$ and a given temperature θ:

$$I_{\rm sub} = k \cdot W \cdot e^{-V_{\rm th}/nV_\theta}(1 - e^{-V_{\rm dd}/V_\theta})$$

where k and n are experimentally derived, W is the gate width, $V_{\rm th}$ is the threshold voltage, and $V_{\rm dd}$ is the supply voltage. V_θ is the thermal voltage that increases linearly as the temperature elevates. Due to the temperature dependence on the subthreshold leakage current, we first use our model to estimate the leakage power based on an initial temperature. The results are then fed to our thermal analyzer so that it will estimate the temperature and the leakage power more accurately. This is done within the thermal analyzer by modeling their interdependence. First, a baseline temperature is calculated with a static leakage estimation, then the leakage power based on those temperatures is calculated, then a new temperature based on the previous iteration's leakage power, and so on, until convergence or thermal runaway is detected. We follow the criteria [41] for detecting the scenarios of thermal runaway: 1) the maximum module temperature $T_{\max}$ is increasing and 2) the increment of power is larger than the increment of the package's heat removal ability. The package's heat removal ability is defined as $(T_{\max} - T_a)/R_t$, where T_a and R_t are ambient temperature and thermal resistance of the package, respectively.

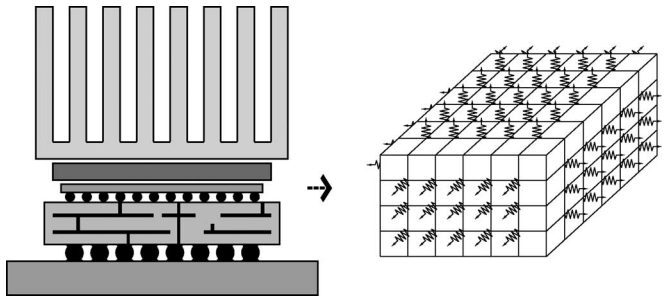

Fig. 3. 3-D grid of a chip for thermal modeling.

D. Thermal Modeling

The linearized differential equation $(k \cdot \nabla^2 T + P = 0)$ for steady-state heat flow was the basis of our thermal model, as described in [16]. In the equation, k is the thermal conductivity, T is the temperature, and P is the power density of heat sources. The chip is divided into a 3-D grid, as shown in Fig. 3, to apply a finite-difference approximation to the differential equation. We rewrite the thermal equation into the matrix form $\mathbf{R} \cdot \overrightarrow{P} = \overrightarrow{T}$, where $\mathbf{R}$ is the thermal resistance matrix ($\mathbf{R}_{i,j}$ is the thermal resistance between node i and node j), $\overrightarrow{P}$ is the power profile vector ($\overrightarrow{P}_i$ is the power dissipation of node i), and $\overrightarrow{T}$ is the temperature profile vector ($\overrightarrow{T}_i$ is the temperature of node i). Thus, the temperature of all the active nodes can now be calculated from the power profile using a single matrix–vector multiplication. The clock power is distributed evenly across the modules according to their areas. The bus

[1]Our algorithm is general enough to take in many different configurations. For the sake of expediency, one configuration was chosen for experimentation.

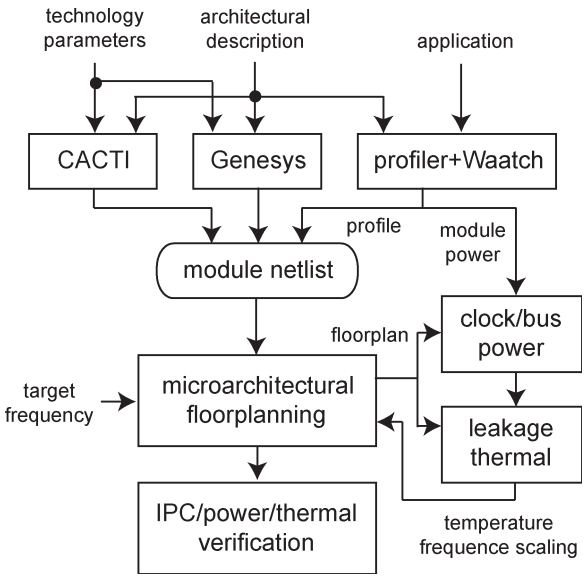

Fig. 4. Overview of our microarchitectural floorplanning.

power for each net is added to the total power of the source block. Then, the leakage power and the temperature of each module are calculated iteratively using our model until they converge or thermal runaway is detected.[2]

In order to facilitate fast but reasonably accurate temperature calculation, we use a nonuniform 3-D thermal resistor mesh, where grid lines are defined at the center of each microarchitectural module. These grid lines are defined for the x and y directions and extend through the z direction to form planes. The intersection of grid lines in the x and y directions defines the thermal nodes of the resistor mesh. Each thermal node models a rectangular prism of silicon that may dissipate power if it covers some portion of a block. The total power of each block is distributed according to and among the $x-y$ area of the nodes that block covers.

E. Integrated Design Flow

Our design flow incorporates the dynamic power, leakage power, performance, and thermal analysis discussed earlier into our floorplanner. An overview of this design flow is illustrated in Fig. 4. First, we use technology parameters and an architectural description to estimate the area and delay of the microarchitectural modules using the following analytical tools: CACTI [42] and GENESYS [40]. Then, a cycle-accurate simulation using SimpleScalar [43] combined with Wattch [44] is done in order to collect and extract the amount of traffic between modules and estimate the dynamic power consumption for each benchmark. From these tools, we extract a profile-weighed module netlist and power consumption information and feed all of these data into our multiobjective floorplanner. We also integrated the clock power estimation from [37] and the leakage estimation from [38] as described above with our thermal analyzer.

[2]The average number of iterations needed was found to be approximately seven for the largest number of layers. A smaller number of layers requires fewer iterations.

Performance and Thermal-aware Floorplanning
while (there exists a partition with multiple modules)
Choose a partition j to be divided;
Call thermal/leakage analysis;
for (number_of_repetitions)
Insert a cutline and compute center of gravity;
Solve LP with inserted cutline;
Pick the best cutline from the set of repetitions;
Update centers of gravity and bounding boxes;
return x_i, y_i, w_i, h_i, z_{ij} for all modules;

Fig. 5. Description of our floorplanning algorithm. We perform a top-down recursive bipartitioning and solve LP-based floorplanning at each iteration.

Our floorplanner consists of two steps, namely: 1) initial solution construction via LP and 2) stochastic refinement via SA. We recursively bipartition the floorplan area until each module is confined in its own partition. Each bipartitioning solution is optimized by an LP-based approach, where performance and thermal objectives are simultaneously considered under the leakage power constraint. We then call our thermal/leakage analyzer upon each bipartitioning to update the thermal and leakage profile. The interdependence between leakage power and temperature creates the possibility of thermal runaway [15], in which temperature and leakage are caught in a positive feedback loop and both continue to exacerbate. If the floorplanner decides that thermal runaway is unavoidable given the current clock frequency, then it scales the frequency down until it succeeds in avoiding runaway. Once the recursive bipartitioning is finished, we further optimize the current solution during our SA-based refinement. We perform low-temperature annealing to fine tune the LP-based solution, where a thermal/leakage analyzer is again used to guide our optimization. When the final solution is obtained, we use SimpleScalar, Wattch, and our thermal/leakage analyzer to evaluate the final solution for IPC, power, and thermal metrics.

IV. 2-D Microarchitectural Floorplanning

Given a set of microarchitectural modules and a netlist that specifies the connectivity among these modules, our multiobjective 2-D microarchitectural floorplanner tries to determine the width and height of each module and to place it into a single chip such that: 1) there is no overlap among modules; 2) a user-specified clock frequency constraint is satisfied; and 3) thermal runaway does not occur under the constraint. Our objective is to provide a floorplan that effectively maximizes the performance of a processor while simultaneously minimizing the footprint area of the floorplan and maximum module temperature for better thermal reliability. We discuss our LP-based floorplan construction and SA-based refinement in this section.

A. LP-Based 2-D Floorplanning

Fig. 5 shows our slicing floorplanning algorithm. The basic idea behind our algorithm is to perform recursive bipartitioning until each partition contains a single module, as shown in Fig. 6. In our approach, the slicing operation determines the overall relative location among the modules, while an LP fine tunes the

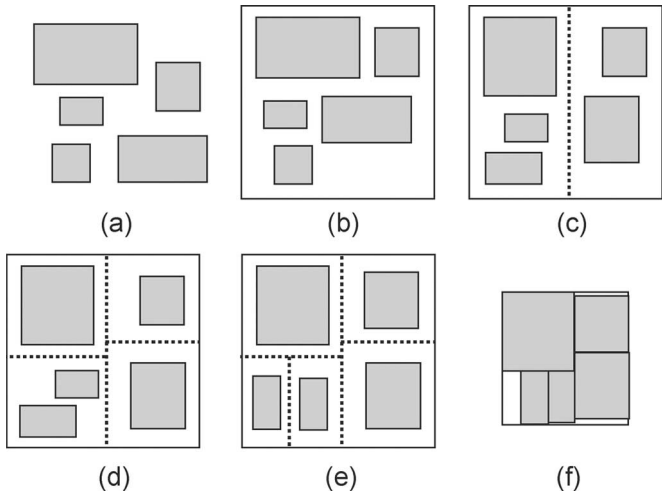

Fig. 6. Illustration of our 2-D microarchitectural floorplanning. (b)–(e) LP-based slicing floorplan. (f) Nonslicing floorplan refinement.

location and determines the dimension of the modules. After we choose a partition to be divided, we perform thermal/leakage analysis to obtain module temperature. The first iteration of the recursive bipartitioning contains no temperature objective because there is no way to obtain block temperatures without a floorplan. All subsequent iterations use temperatures calculated from the previous iteration's block positions. We then use our LP-based floorplanning to simultaneously optimize the performance and thermal distribution under the target frequency, leakage, center of gravity constraints (to remove overlap among the modules), and boundary constraints. An "iteration" in our algorithm combines a single bipartitioning and a subsequent LP-based floorplanning of all modules. Thus, we perform $k-1$ iterations if there are k modules in the netlist. Note that each iteration can be repeated multiple times to obtain different cutlines. This is because there exist multiple solutions that satisfy the boundary and center of gravity constraints during each bipartitioning. Thus, we perform each bipartitioning several times and pick the best solution in terms of performance and thermal profile.

The following variables are used for our LP-based floorplanning formulation:

N	set of all modules in the netlist;
E	set of all nets in the netlist;
x_i, y_i	location of module i;
w_i, h_i	half width and half height of module i;
a_i, g_i	area and delay of module i;
$w_m(i)$, $w_x(i)$	minimum/maximum width of module i;
$\lambda_{i,j}$	normalized profile weight on wire (i,j);
$z_{i,j}$	number of FFs on wire (i,j) after insertion;
$X_{i,j}$	$=\lvert x_i - x_j\rvert$ and $Y_{i,j} = \lvert y_i - y_j\rvert$;
$T_{i,j}$	normalized product of the temperature of modules i and j;
A	aspect ratio of the chip;
X_x	maximum x_i;
Y_x	maximum y_i;
C	target cycle time;
d_r	unit length delay of repeated interconnects.

Our LP floorplanner determines the values for the following decision variables: x_i, y_i, w_i, h_i, and z_{ij}. The following are the variables used for bipartitioning:

$B(u)$	set of all modules at iteration u;
$M_j(u)$	set of all modules in partition j at iteration u;
$S_{j,k}(u)$	set of modules assigned to subpartition k ($k \in \{1,2\}$ for bipartitioning) in partition j at iteration u;
$(\bar{x}_{jk}, \bar{y}_{jk})$	center of subpartition k contained in partition j;
r_j, v_j, t_j, b_j	the right, left, top, and bottom boundaries of partition j.

Our LP formulation is used to perform floorplanning at iteration u of the main algorithm shown in Fig. 5. Our LP-based slicing floorplanning is formulated as follows:

Minimize

$$\sum_{(i,j)\in E} (\alpha \cdot \lambda_{ij} \cdot z_{ij} + \beta \cdot (1 - T_{ij})(X_{ij} + Y_{ij}) + \gamma \cdot X_x) \tag{1}$$

subject to

$$z_{ij} \geq \frac{g_i + d_r(X_{ij} + Y_{ij})}{C}, \qquad (i,j) \in E \tag{2}$$

$$X_{ij} \geq x_i - x_j \text{ and } X_{ij} \geq x_j - x_i, \qquad (i,j) \in E \tag{3}$$

$$Y_{ij} \geq y_i - y_j \text{ and } Y_{ij} \geq y_j - y_i, \qquad (i,j) \in E \tag{4}$$

$$z_{ij} \geq 0, \qquad (i,j) \in E \tag{5}$$

$$w_m(i) \leq w_i \leq w_x(i), \qquad i \in N \tag{6}$$

$$x_i, y_i \geq 0, \qquad i \in N \tag{7}$$

$$X_x \geq x_i \text{ and } A \cdot X_x \geq y_i, \qquad i \in N. \tag{8}$$

Boundary constraints

$$x_i + w_i \leq r_j, \qquad i \in M_j(u), \quad j \in B(u) \tag{9}$$

$$x_i - w_i \geq v_j, \qquad i \in M_j(u), \quad j \in B(u) \tag{10}$$

$$y_i + m_i w_i + k_i \leq t_j, \qquad i \in M_j(u), \quad j \in B(u) \tag{11}$$

$$y_i - m_i w_i - k_i \geq b_j, \qquad i \in M_j(u), \quad j \in B(u). \tag{12}$$

Center of gravity constrains: for $k \in \{1,2\}$, $j \in B(u)$

$$\sum_{i\in S_{jk}(u)} a_i x_i = \sum_{i\in S_{jk}(u)} a_i \times \bar{x}_{jk} \tag{13}$$

$$\sum_{i\in S_{jk}(u)} a_i y_i = \sum_{i\in S_{jk}(u)} a_i \times \bar{y}_{jk}. \tag{14}$$

Our objective function shown in (1) contains three terms, namely: 1) profile-weighed wirelength $(= \lambda_{ij} \cdot z_{ij})$; 2) thermal-weighed wirelength $(= (1 - T_{ij})(X_{ij} + Y_{ij}))$; and 3) footprint area $(= X_x)$, where λ_{ij} is the profiled activity factor of the wire between modules i and j.[3] The minimization

[3]Since we add performance and thermal-related weights to the pure wirelength, we do not explicitly consider nonweighed pure wirelength objective. However, we report the wirelength metric in all of our experiments to show the impact of this multiobjective on wirelength.

of the first term improves IPC, while the minimization of the second term stretches the distance of two modules, thereby reducing thermal coupling. $(1 - T_{ij})(X_{ij} + Y_{ij})$ was chosen as the temperature-dependent portion of the cost function because it satisfies several properties, i.e., it is linear with respect to distance between module i and module j, it considers the temperatures of both module i and module j, and it grows smaller when considering hot blocks and larger when considering cool blocks. Because the cost function is being minimized in the LP, it is necessary to only consider minimizing the distance between cool blocks and not maximizing the distance between hot blocks, as would be preferable. Since minimizing $X_x \cdot Y_x$ (= floorplan area) is nonlinear, we only minimize X_x since the constraint (8) enforces $A \cdot X_x$ to be greater than all y values. Note that α, β, and γ are user-defined parameters for weighing the performance, thermal, and area objectives. In case $\alpha = 0$, our floorplanner optimizes thermal + area only. In case $\beta = 0$, our floorplanner optimizes the performance + area objective only. Lastly, the conventional area/wirelength-driven floorplanner uses the new objective function

$$\gamma \cdot X_x + \delta \cdot \sum_{(i,j) \in E} (X_{ij} + Y_{ij}). \tag{15}$$

We provide an extensive comparison among these four different floorplanning objectives (simultaneous performance + thermal + area, performance + area, thermal + area, and area + wirelength) in Section VI-C.[4]

Constraint (2) is obtained from the definition of latency. If there is no FF on a wire (i, j), the delay of this wire is calculated as $d(i, j) = d_r(X_{ij} + Y_{ij})$. Then, $g_i + d(i, j)$ represents the latency of module i accessing module j, where $d(i, j)$ denotes the delay between i and j. Since C denotes the clock period constraint, $(g_i + d(i, j))/C$ denotes the minimum number of FFs required on (i, j) in order to satisfy C. Absolute values on the x and y distances are given in (3) and (4). Constraint (5) requires that the number of FFs on each edge is nonnegative. The block boundary constraints (9)–(12) require that all modules in the block be enclosed by these block boundaries. The center of gravity constraints (13) and (14) requires that the module area-weighed mean (= center of gravity) among all modules in each subblock corresponds to the center of the subblock.

B. Stochastic Refinement

The standard LP relaxation of the floorplanning problem introduces several nonoptimalities. The recursive bipartitioning process also yields only slicing floorplans. In order to address these issues, we implemented an SA-based refinement engine for our floorplanner. This allows us to search around the local space and find a local minimum without being constrained by linearity. We use three intralayer moves during the SA refinement, namely: 1) swapping in positive sequence; 2) swapping in both positive and negative sequences; and 3) rotation. We derive a sequence pair from the LP floorplanning result and perform low-temperature annealing with them. We use the "gridding" scheme described in [45] to derive the corresponding sequence pair representation from the slicing floorplan. Specifically, we draw the positive and negative loci for each module and order these loci to obtain the sequence pair. Next, we compute the initial annealing temperature by setting the probability of accepting bad moves to a low value. This reduces the runtime required for the annealing process significantly and focuses on results that are near the LP-based result, which is assumed to be fairly close to optimal. During our annealing, we use the cost function

$$\text{cost} = \alpha \cdot \text{per}\_f\text{\_wire} + \beta \cdot \max\text{\_temp} + \gamma \cdot \text{area}$$

where per f_wire is the profile-weighed wirelength, and $\max$_temp is the maximum module temperature. We use the same weighting constants α and β used in (1) between the performance and thermal objectives. It is important, however, to note that our temperature is "not" the weighed distance between two hot blocks but the "actual" temperature we obtain from our thermal analyzer. Thus, our thermal analysis is the runtime bottleneck during our refinement since we need to perform the analysis for potentially many candidate solutions during the annealing process. The consideration of performance is done in both SA and LP approaches by inclusion of the profile-weighed wirelength in the cost function.

Assuming that the thermal conductivity of functional modules is similar (they are mostly silicon), swapping the location of modules would not change the thermal resistance matrix $\mathbf{R}$. This means that matrix $\mathbf{R}$ only needs to be computed once in the beginning. To calculate the temperature profile of a new floorplan, the power vector $\overrightarrow{P}$ needs to be updated and then multiplied by $\mathbf{R}$. Alternatively, a change in power profile $\Delta\overrightarrow{P}$ can be defined. Multiplying $\mathbf{R}$ and $\Delta\overrightarrow{P}$ will give a change in temperature vector $\Delta\overrightarrow{T}$. Adding $\Delta\overrightarrow{T}$ to the old temperature vector will give the new temperature profile. Swapping two blocks usually has a small effect on the power profile, so $\Delta\overrightarrow{P}$ is usually sparse. This reduces the number of multiplications required by the second method at the expense of doing extra additions and subtractions. This approach may not give us the most accurate temperature numbers but does provide high fidelity to distinguish good solutions from bad ones. Our related experiments shown in Section VI-F support this claim. Lastly, the leakage and clock power updates are done faster since it basically involves evaluating a set of equations based on the new module locations and temperature values.

V. Extension to 3-D Floorplanning

The extension to 3-D floorplanning requires a new approach in floorplanning as well as updates on the architectural simulation for performance, power, and thermal evaluation. Our 3-D floorplanning algorithm considers the issues that are specific to 3-D: vertical overlap optimization and bonding-aware layer partitioning. We solve this problem using our LP-based 3-D slicing floorplanning plus stochastic nonslicing floorplan refinement.

[4]Note that the area objective is used in all of these variations. The area objective has a positive impact on performance and wirelength objectives and a negative impact on thermal objective.

A. 3-D Extension of Architectural Simulation

In order to support the performance, power, and thermal simulation for 3-D microarchitecture floorplanning, we extend the simulation engines discussed in Section III as follows.

1) Performance: The IPC computation for 3-D is not too different from the 2-D case except that the access latency on each interconnect is calculated based on a 3-D floorplan that involves delay in the z dimension.
2) Dynamic power: We again assume that the module power is independent of floorplanning. However, bus and clock power are heavily dependent on floorplanning and given the reduction of interconnect lengths in a 3-D floorplan. The existing bus power calculator is extended to consider interlayer interconnects. We assume that an H-tree is used for each layer, and these H-trees are connected by through-vias. The number of FFs and buffers included in the 3-D clock tree is calculated based on the area of each layer.
3) Temperature/Leakage: The thermal analysis for 3-D becomes more complex because of the multiple die structures. Thus, we add more layers in our 3-D mesh to model the multiple sets of device, metal, and bonding layers. The leakage power computation is straightforward in our model once the temperature for each module is known.

Finally, the architecture-to-floorplan design flow shown in Fig. 4 remains the same except that all the related boxes now are 3-D aware.

B. Vertical Overlap Optimization

A unique challenge in 3-D floorplanning is the issue of "vertical module overlap." The primary benefit that a 3-D IC provides is the ability to place the tightly connected modules "on top of" each other instead of "adjacent to" each other as in the 2-D case. This reduces the length and thus the delay/power of related interconnects significantly. Since the parasitics associated with the interdie vias is similar to those of short interconnects, the additional freedom in z-dimension promises higher-quality floorplans in terms of footprint area, performance, and power consumption. In addition, the shorter interconnects naturally mitigate the interconnect congestion problems. More specifically, the vertical overlap affects the quality of 3-D microarchitectural floorplanning in the following ways.

1) Performance: The performance of a 3-D microarchitectural floorplan tends to improve when the vertical overlap is maximized among blocks with higher access frequencies. This is mainly caused by the shorter interconnect and thus the lower access latency among the frequently communicating modules.
2) Thermal: The thermal profile of a 3-D microarchitectural floorplan tends to deteriorate due to compressed space. More hotspots are created when the vertical overlap is maximized among the hot modules. This harmful thermal coupling causes the leakage power to increase, raising the likelihood of thermal runaway.

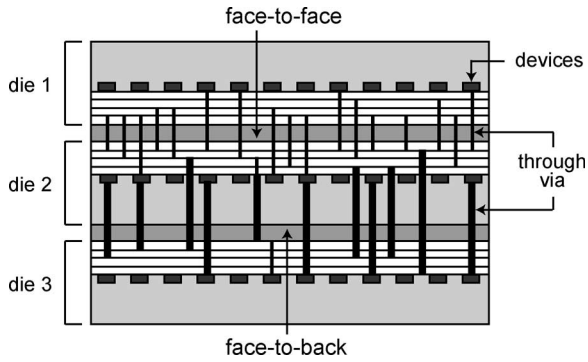

Fig. 7. Through-vias in 3-D ICs with F2F and F2B bonding. B2B style forms when the two substrate sides are attached (not shown in this figure).

3) Power: The dynamic module power and clock power are rarely affected by the vertical overlap. However, the overall bus power consumption tends to decrease with more vertical overlap among the modules with higher switching activities. This is because the dynamic power saving is greater when highly active modules drive shorter interconnects. Note that this contradicts with the thermal objective since highly active modules tend to become hotter.

In summary, our 3-D floorplanning tries to maximize the vertical overlap among the frequently communicating and highly switching modules while minimizing the vertical overlap among the hot modules.[5] Since these objectives are competing with each other, trading one objective off the other is inevitable.

C. Bonding-Aware Layer Partitioning

A 3-D IC requires special kinds of vias for interdie connection called "through-vias." There are three kinds of through-vias depending on the style of bonding mechanism used to bond two dies together, namely: 1) F2F; 2) F2B; and 3) B2B through-vias, as illustrated in Fig. 7. "Face" refers to the metal layer side of a die, whereas the substrate side is called "back." F2F through-vias ($\approx 0.5 \times 0.5\mu$) have a smaller pitch than F2B ($\approx 5 \times 5\mu$) and B2B through-vias ($\approx 15 \times 15\mu$) [46]. In addition, too many F2B/B2B through-vias fabricated on a single thinned wafer may adversely affect its reliability [47] since these vias actually penetrate the substrate. Thus, it is desirable to "reduce" the number of interdie connections in F2B/B2B bonding. In the case of F2F bonding, however, it is desirable to "increase" the number of interdie connections since the via density is much higher (almost the same as intradie via density) and thus enables a significantly higher bandwidth for interlayer communication. Note that F2B/B2B bonding is inevitable if the number of die exceeds two. Moreover, in the case that all three bonding styles are used in a single 3-D IC, 3-D floorplanning has to be done carefully to exploit both bonding styles.

[5]Note that it is possible to impose the vertical overlap constraints among the related groups of modules. The investigation of this direction is out of the scope of this paper, which may require the extension of floorplanning encoding scheme such as Sequence Pair [45].

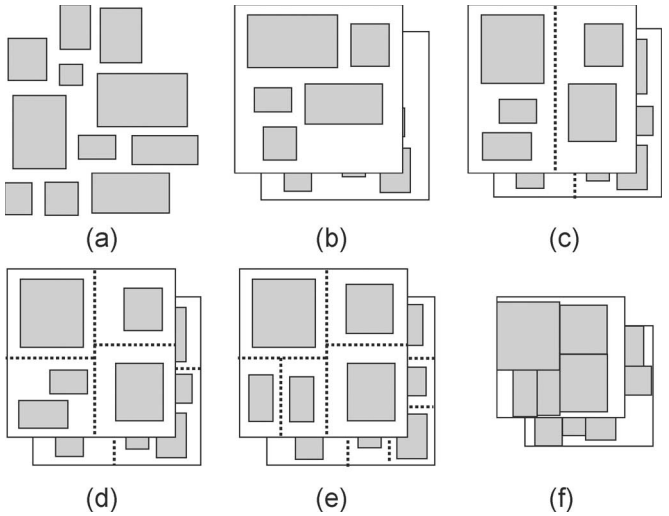

Fig. 8. Illustration of our 3-D microarchitectural floorplanning. (b) Layer partitioning. (c)–(e) LP-based 3-D slicing floorplan. (f) Nonslicing floorplan refinement.

In our two-step approach for 3-D floorplanning, we first partition the modules into layers (= die) and then floorplan these layers. The goal during our layer partitioning is to exploit the bonding style and vertical overlap opportunities, whereas our floorplanning optimizes the vertical overlap for performance, footprint area, and thermal objectives. During our layer partitioning, we assign a layer to each module such that the connection at the F2F boundary is maximized while the F2B/B2B connection is minimized. Next, we split the pair of modules connected via high profile-weighed edge into two layers with F2F bonding so that we can vertically overlap them during the subsequent floorplanning step for achieving better performance. In addition, we split highly active modules in the same way, i.e., two layers with F2F bonding, such that the shorter interconnect connected to these modules helps reduce the dynamic power. Since the temperature of the modules requires floorplanning, our layer partitioning is not temperate aware. Finally, we separate the modules with large area such as the RUU into different layers to help minimize the footprint area and reduce the amount of whitespace. In our greedy construction algorithm, we sort the modules according to their size, power density, and switching activity. We then assign the best possible layer for each module based on the performance, power, and area objectives mentioned earlier.

D. LP-Based 3-D Floorplanning

In our LP-based 3-D floorplanning, we extend the slicing floorplanning discussed in Section IV-A to handle multiple layers simultaneously. Specifically, we insert each slicing cutline to cut all layers simultaneously, as illustrated in Fig. 8. The goal of our slicing 3-D floorplanning remains the same as the 2-D case, i.e., to determine the dimension and relative position among the modules so that the multiobjective function is minimized. In addition, these locations will be refined via our 3-D nonslicing floorplanning during our postrefinement. The major difference between the 2-D and 3-D slicing floorplan is the interaction with different layers, which is the key element for an effective 3-D floorplan. More specifically, the vertical overlap discussed in Section V-B has a high impact on performance and thermal objectives. In addition, area optimization has to be footprint aware, i.e., the area increase from the smallest layer can be easily tolerated since it is less likely to increase the overall footprint area. Our LP formulation reflects this new optimization goal that is unique to 3-D floorplanning. Since layer partitioning has already addressed the bonding-style-related issues, we do not allow the modules to move to other layers during the floorplanning.

The following 3-D-related LP variables are used in conjunction with the 2-D-related variables shown in Section IV-A: l_i: layer of module i, $L_{ij} = |l_i - l_j|$, d_v: delay of interlayer vias. It is crucial to note that the LP objective function used for 2-D floorplanning, i.e., (1), can be used "as is" so long as we consider "all" layers simultaneously. Specifically, the $\alpha \cdot \lambda_{ij} \cdot z_{ij}$ term in (1) minimizes the distance between the frequently communicating modules if these are in the same layer; if not, the vertical overlap will be maximized as long as the reference point of module location is consistent.[6] In addition, the $\beta \cdot (1 - T_{ij})(X_{ij} + Y_{ij})$ term separates two hot modules in the same layer and minimizes the vertical overlap between two hot modules in different layers. Finally, the $\gamma \cdot X_x$ term still captures the minimization of 3-D footprint area as long as X_x and Y_x are computed based on the modules in all layers. The only difference between the LP formulations of 2-D and 3-D floorplanning is the latency constraint, for which we update (2) with

$$z_{ij} \geq \frac{g_i + d_r(X_{ij} + Y_{ij}) + d_v L_{ij}}{C}, \qquad (i,j) \in E. \quad (16)$$

This latency constraint considers the delay of interlayer via delay as well as interconnect delay during the computation of FFs needed to satisfy the clock period constraint C. We assume that d_r (= unit length delay of repeated interconnects) is larger than d_v (= delay of interlayer vias).

E. 3-D Stochastic Refinement

The goal of our 3-D stochastic refinement is to improve the 3-D slicing floorplanning solution we obtain from our LP-based construction algorithm. Our basic approach is the same as the 2-D case discussed in Section IV-B, i.e., nonslicing floorplanning with low-temperature SA to simultaneously refine the performance, thermal, and area objectives. The major difference between the 2-D and 3-D cases is that we use one sequence pair per layer to represent the entire 3-D solution. In addition, our perturbation scheme does not allow interlayer module movement to maintain the bonding-aware layer separation. Finally, temperature calculation takes even longer since our thermal model needs to be expanded to consider multiple dies. Thus, the annealing schedule is adjusted in such a way not to increase the runtime too much, which involves tuning such parameters as the initial/final annealing temperature, total number of moves each annealing temperature, cooling ratio, and annealing termination criteria.

[6]We use the lower-left corner of each module in our case.

TABLE I
COMPARISON WITH CBA-T [22]. OUR FLOORPLANNER IS LP + SA WITH A + W + T OBJECTIVE. THE BASELINE IS CBA-T

	CBA-T [22]			LP+SA (ratio)		
bench	area	wire	temp	area	wire	temp
ami33	4.14e+05	24442	160	0.94	1.11	0.96
ami49	1.84e+07	477646	151	0.79	1.21	0.94
n100	6.56e+04	92450	158	1.27	0.95	0.93

VI. EXPERIMENTAL RESULTS

A. Experimental Setting

Our experiments were performed on ten programs from the SPEC2000 benchmark suite. We chose four from the floating point and six from the integer benchmark suites. For IPC evaluation, we ran each benchmark on the average case floorplan using a modified SimpleScalar 3.0 [43] by fast-forwarding four billion instructions and simulating the next four billion instructions. The reported temperature is simulated after all floorplanning steps and is adjusted relative to a 45 °C ambient temperature. We report the maximum temperature among all blocks in the floorplan. Our 3-D floorplan is based on a four-layer stacked IC. We assume F2F bonding between layer 0 (topmost) and 1 and layer 2 and 3. A B2B bonding is used between layer 1 and 2. The heat sink is attached to layer 3. Wirelength is reported in millimeter. The "area" in our results refers to the footprint area (= maximum width × maximum height) of the four-layer floorplan and is reported in square millimeter. The runtime of our framework was collected on Pentium Xeon 2.4-GHz dual-processor systems. The runtime of profiling four billion instructions after fast-forwarding four billion instructions was about 4 h per benchmark as was the power collection simulation for the same sets of instructions. The floorplanning steps took approximately 25 min, and the simulations for the reported values of temperature and IPC took approximately 2 min and 1 h per benchmark.

B. Comparison to Existing 3-D Floorplanner

Table I shows the comparison of our floorplanner to CBA-T [22]. Here, we tested our floorplanner with the MCNC and GSRC benchmark circuits that were used in [22]. Since the power density values are randomly generated in [22], a fair temperature comparison is not possible. Since the MCNC/GSRC benchmarks are not microarchitecture designs, we cannot compute the power density using our tool. We note, however, that our floorplanner obtains comparable results in terms of area, wirelength, and temperature. In addition, tuning the weighting constants among the objectives may result in different results.

C. Floorplanning Results

Table II presents various tradeoffs existing in multiobjective 2-D floorplanning. We use our LP + SA method. One can see that the maximum module temperature increased markedly for A + P compared to the baseline A + W. The IPC result of A + P is the best among the four algorithms with an average IPC improvement over A + W of 35%. A + T decreases the temperature by about 24% over A + P, while the IPC decreases by 25%. The hybrid A + P + T decreases the temperature by 14% over A + P while maintaining a high IPC value of 22% above the baseline A + W. In general, as the IPC increases, the block-level dynamic power also increases due to the higher activity, which results in a high temperature. This is a reason why A + W obtained a lower temperature than A + P and A + P + T. This can also be seen from the fact that A + P obtains the highest IPC as well as temperature. Thus, the temperature drop in A + T compared to A + P is the result of smart floorplanning and lower IPC.

For the 3-D case shown in Table III, 3-D A + W achieves a 37% increase in IPC and a 34% increase in temperature over 2-D A + W while decreasing the total wirelength by almost 40%. The area result of 3-D A + W is the best among all objective functions. A + P increases the IPC by 18% over A + W and increases the temperature by 19%. As expected, A + T decreases the temperature result of A + P significantly and achieves the best temperature results among all four 3-D algorithms. The 4× increase in grid size for the temperature simulations in the 3-D case causes the runtime of those objectives incorporating temperature calculations to increase dramatically.[7] The hybrid A + P + T retains a temperature close to that of A + W while increasing the IPC by 14%. In summary, A + P + T: 1) obtains results that are between those of A + T and A + P and 2) outperforms A + W in terms of performance with comparable temperature results for both 2-D and 3-D. In case the temperature should be more emphasized, the thermal weight can be increased, which will likely lead to performance degradation.

Also shown in Tables II and III are the pipeline depth and whitespace percentages for the various objective functions, respectively. First, the pipeline depth ranges from 17 to 23, which agrees with current trends in commercial processor designs, e.g., a 90-nm Pentium-4 back-end pipeline has 31 stages, the Intel's NGMA has 14 stages, etc. Despite the increase in pipeline depth from FF insertion, our strategy to add FFs on noncritical wires does not degrade the performance, while removing FFs from critical wires improved performance. Second, whitespace ranges from 7% to 23%. In case of area-only objective, the whitespace is 7% for both 2-D and 3-D cases. This whitespace keeps increasing as we consider other objectives. The whitespace increase caused by wirelength consideration is only 2%–3%, while performance and thermal objectives cause the whitespace to increase by 9%–13% and 16%–18%, respectively. Due to the unbalance in block area, it becomes more difficult to optimize whitespace while placing frequently communicating blocks closer (= performance) or separating hot blocks apart (= temperature).

A tradeoff between performance and temperature is shown in Fig. 9. Temperature and IPC are reported as averages over ten benchmarks. The performance and area weights are held constant, while the thermal weight is varied. As expected, the graph shows that as the thermal weight is given more consideration by the floorplanner, the performance drops. Ideally, there would

[7]Our recent study [48] shows that the Random Walk method can improve the runtime of thermal simulation significantly. Our future work includes the integration of this scheme in our microarchitectural floorplanning.

TABLE II
MULTIOBJECTIVE 2-D FLOORPLANNING RESULTS WITH PERFORMANCE (P), MAXIMUM BLOCK TEMPERATURE (T), AREA (A), AND WIRELENGTH (W) OBJECTIVES. THE LP + SA-BASED FLOORPLANNER IS USED. TEMPERATURE IS IN DEGREE CELSIUS

	A		A+W		A+P		A+T		A+P+T	
bench	IPC	temp	IPC	temp	IPC	temp	IPC	temp	IPC	temp
gzip	2.04	80.4	2.01	78.3	2.83	100.4	2.03	75.2	2.69	86.2
swim	0.48	66.9	0.52	64.3	0.85	78.4	0.54	63.0	0.66	70.5
vpr	0.77	90.4	0.95	87.6	1.19	113.8	0.82	82.3	1.15	95.9
art	0.34	64.4	0.38	67.9	0.62	83.3	0.39	65.4	0.51	74.4
mcf	0.03	64.1	0.07	63.0	0.09	76.9	0.07	62.1	0.10	69.4
equake	0.34	65.5	0.40	62.7	0.47	76.3	0.41	61.8	0.43	69.0
lucas	0.58	101.3	0.63	95.6	0.75	123.2	0.64	88.3	0.80	103.5
gap	1.19	70.9	1.17	70.1	1.24	87.8	1.18	68.1	1.32	77.3
bzip2	1.43	82.1	1.42	80.4	1.90	103.6	1.47	77.1	1.65	88.4
twolf	0.59	97.4	0.60	92.3	0.94	120.8	0.61	85.8	0.61	101.1
AVG	0.78	78.4	0.81	76.2	1.09	96.46	0.82	72.9	0.99	83.6
area (mm^2)	50.5		52.46		57.23		58.66		60.37	
wire (mm)	380.23		345.20		412.15		358.86		449.67	
time (sec)	168		174		188		1116		1064	
pipeline stage	22		22		19		27		23	
whitespace %	7		10		20		23		21	

TABLE III
MULTIOBJECTIVE 3-D FLOORPLANNING RESULTS WITH PERFORMANCE (P), MAXIMUM BLOCK TEMPERATURE (T), AREA (A), AND WIRELENGTH (W) OBJECTIVES. THE LP + SA-BASED FLOORPLANNER IS USED

	A		A+W		A+P		A+T		A+P+T	
bench	IPC	temp	IPC	temp	IPC	temp	IPC	temp	IPC	temp
gzip	2.40	108.8	2.74	104.7	3.98	125.9	2.75	98.9	2.85	104.7
swim	0.72	91.9	0.71	92.9	0.85	106.9	0.72	84.1	0.92	88.0
vpr	0.98	120.7	1.30	111.5	1.40	137.0	1.25	107.1	1.29	114.4
art	0.58	95.6	0.52	95.6	0.59	111.4	0.52	87.9	0.61	92.0
mcf	0.21	97.8	0.10	92.0	0.11	105.4	0.10	83.1	0.07	86.6
equake	0.59	89.7	0.54	91.7	0.58	105.0	0.55	82.6	0.67	86.2
lucas	0.88	127.2	0.87	116.9	0.92	145.3	0.88	113.0	1.19	123.0
gap	1.47	96.5	1.59	97.0	1.59	114.2	1.62	89.6	1.61	94.5
bzip2	1.75	115.0	1.94	106.8	2.05	129.0	1.98	101.5	2.33	107.4
twolf	0.84	119.1	0.81	114.6	1.03	142.2	0.84	111.0	1.02	118.9
AVG	1.04	105.3	1.11	102.4	1.31	122.2	1.12	95.8	1.26	101.6
area (mm^2)	21.6		22.20		23.63		25.45		26.45	
wire (mm)	247.25		217.20		323.43		252.08		247.02	
time (sec)	175		180		438		16913		20016	
pipeline stage	22		22		17		24		21	
whitespace %	7		9		16		25		23	

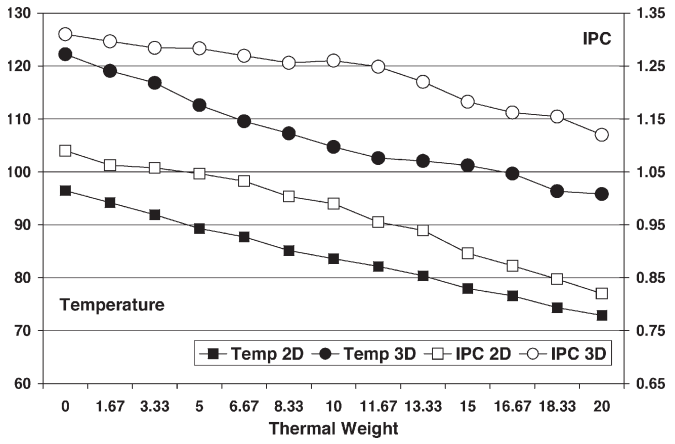

Fig. 9. Tradeoff between performance and temperature. Performance and area weights are held constant while thermal weight varies.

be some separation between the curves to indicate that high reduction in temperature could occur with little degradation in IPC value. The sweet spot of the curve appears when the thermal weight is around ten. The IPC drops sharply after this and so would be undesirable for the reduction in temperature achieved. One can observe that there is a 15% reduction in IPC and a 22% reduction in temperature between the performance-only objective (0) and the highest weight hybrid objective (20) for the 3-D case. As expected and also shown in Table II, the multilayer floorplans increase both the temperature and the IPC over the single-layer floorplans. Also of note is that the highest thermal weight multilayer floorplan has a temperature close to that of the lowest thermal weight single-layer floorplan while achieving a higher IPC. This demonstrates the benefits rendered by moving to multilayer ICs.

D. Optimization Method Comparison

Experimental results were also gathered across the three floorplanning algorithms, namely: 1) LP only; 2) SA; and 3) the combined approach of LP followed by SA refinement. Table IV presents a comparison of IPC, temperature, area, wirelength, and runtime of these three floorplanning algorithms for the 2-D and 3-D cases. One can observe for the 2-D case from the table that the LP floorplanner does very poorly on the area of the floorplan and is not as good as the combined approach for IPC. The wirelength values are within the acceptable range for all approaches, although it is interesting to note that while the

TABLE IV
COMPARISON AMONG PURE-SA, PURE LP, AND LP + SA APPROACHES. THE OBJECTIVE USED IS A LINEAR COMBINATION OF PERFORMANCE, TEMPERATURE, AND AREA, ALL WITH EQUAL WEIGHT. AREA IS IN SQUARE MILLIMETER, WIRELENGTH IS IN MILLIMETER, AND TEMPERATURE IS IN DEGREE CELSIUS

	2D floorplan						3D floorplan					
	pure SA		pure LP		LP+SA		pure SA		pure LP		LP+SA	
bench	IPC	temp	IPC	temp	IPC	temp	IPC	temp	IPC	temp	IPC	temp
gzip	2.38	102.2	1.94	80.19	2.69	86.2	2.74	109.5	2.31	97.5	2.85	104.7
swim	0.61	83.5	0.66	69.3	0.66	70.5	0.71	91.8	0.70	86.7	0.92	88.0
vpr	0.93	113.1	1.24	86.9	1.15	95.9	1.07	119.8	1.24	103.4	1.29	114.4
art	0.45	87.5	0.48	71.9	0.51	74.4	0.52	95.7	0.51	89.0	0.61	92.0
mcf	0.08	82.0	0.09	68.3	0.10	69.4	0.10	90.4	0.10	85.9	0.07	86.6
equake	0.47	81.6	0.49	68.1	0.43	69.0	0.54	90.0	0.53	85.7	0.67	86.2
lucas	0.75	122.6	0.79	93.8	0.80	103.5	0.87	128.7	0.85	108.1	1.19	123.0
gap	1.38	91.1	1.34	73.7	1.32	77.3	1.59	98.9	1.49	90.9	1.61	94.5
bzip2	1.68	105.2	1.59	81.8	1.65	88.4	1.94	112.2	1.81	99.4	2.33	107.4
twolf	0.70	118.6	0.68	90.1	0.61	101.1	0.81	124.8	0.77	106.2	1.02	118.9
AVG	0.94	98.7	0.93	78.4	0.99	83.6	1.09	106.2	1.03	95.3	1.26	101.6
AREA	60.90		314.72		60.37		21.59		70.64		26.45	
WIRE	388.13		524.81		449.67		230.47		207.57		247.02	
TIME	1225		826		1064		25157		18207		20016	

TABLE V
COMPARISON BETWEEN DIFFERENT LAYER PARTITIONING STYLES. THE HYBRID A + P + T OBJECTIVE IS USED WITH COMBINED LP + SA APPROACH. AREA IS IN SQUARE MILLIMETER, WIRELENGTH IS IN MILLIMETER, AND TEMPERATURE IS IN DEGREE CELSIUS

	area-greedy		bonding-prof		bonding-area	
bench	IPC	temp	IPC	temp	IPC	temp
gzip	2.98	108.9	2.88	108.8	2.85	104.7
swim	0.77	93.0	0.87	96.8	0.92	88.0
vpr	1.16	117.8	1.54	112.9	1.29	114.4
art	0.57	95.6	0.65	99.7	0.61	92.0
mcf	0.11	91.8	0.12	92.9	0.07	86.6
equake	0.59	89.8	0.66	95.3	0.67	86.2
lucas	0.96	127.2	1.06	117.1	1.19	123.0
gap	1.77	99.8	1.88	100.2	1.61	94.5
bzip2	2.14	110.4	2.29	109.2	2.33	107.4
twolf	0.90	126.9	0.95	118.0	1.03	118.9
AVG	1.20	106.1	1.29	105.1	1.26	101.6
AREA	22.68		52.54		26.45	
WIRE	270.73		263.26		247.02	
TIME	19872		20102		20016	

LP-only approach creates a large area, the wirelength values are actually less. This is because while wirelength was an objective during the recursive bipartitioning phase of the LP, the area is not because the formulation has no way to constrain the overlap. This was a large part of the motivation to use SA to refine the LP-only solution. In summary, LP + SA improves LP and outperforms SA consistently in terms of both performance and thermal objectives. The runtime of all approaches was roughly equivalent, showing that in a similar amount of time, the combined approach produces better solution quality. These trends are consistent for the 3-D cases with increased overall temperature averages and runtime. Again, the large runtime increase was due mainly to the increase in simulation time for the temperature.

Table V shows a comparison among the three different layer partitioning styles, namely: 1) area-greedy; 2) bonding-prof; and 3) bonding-area. In area-greedy, the blocks are sorted in decreasing order of their area and assigned to each layer so that the overall area is balanced among the layers. Cutsize is not optimized in this case. In bonding-prof, our goal is to optimize the profile-weighed cutsize among the modules. Lastly, bonding-area is the algorithm introduced in Section V-C. We observe that the bonding-area partitioning outperforms a pure area-based approach on IPC and temperature. It has a slightly lower IPC than the bonding-prof partitioning, but the area is completely unacceptable in bonding-prof. The wirelength and runtime of all approaches were comparable.

E. Architectural Analysis

Fig. 10 shows snapshots of our floorplanning solution. We use LP + SA with area, performance, and temperature objectives. The whitespace of the floorplan is somewhat less than optimal, but this is due to the higher weights placed on performance and temperature optimization.[8] Our flow provides the users with the ability to modify the objective weights to suit their needs. This figure demonstrates that there is indeed thermal coupling between adjacent modules and that the thermal portion of the objective has attempted to separate the hottest modules while the performance portion of the objective has caused some of the hottest modules to remain grouped. This stays in line with the rapid dropoff in performance with decreased temperature shown in Fig. 9.

Table VI shows the top ten microarchitectural modules under various metrics. Physical designers are often only able to view the modules at the floorplan level as little more than rectangles. Here, we provide some more detailed information about each of the modules that make up the floorplan. This can provide better opportunities for optimization at the physical design level. The RUU [36] with a large number of read/write ports is larger in area than the next two largest modules combined, which is why it was split up for the multilayer floorplans. The power density of the ALUs is higher than most of the other modules; hence, their temperatures are also generally among the highest in the floorplan. The 3-D floorplan is able to mitigate this by placing ALUs in different layers. Although several modules can have similar power consumption, their temperatures may be different because their nearest neighbors can have a large impact on

[8]These floorplans also highlight the challenge in area optimization for the multiobjective multilayer floorplanning problem. Our future work tries to address this problem more effectively. A possible solution is to utilize the whitespace for decoupling capacitors, thermal vias, buffers, etc.

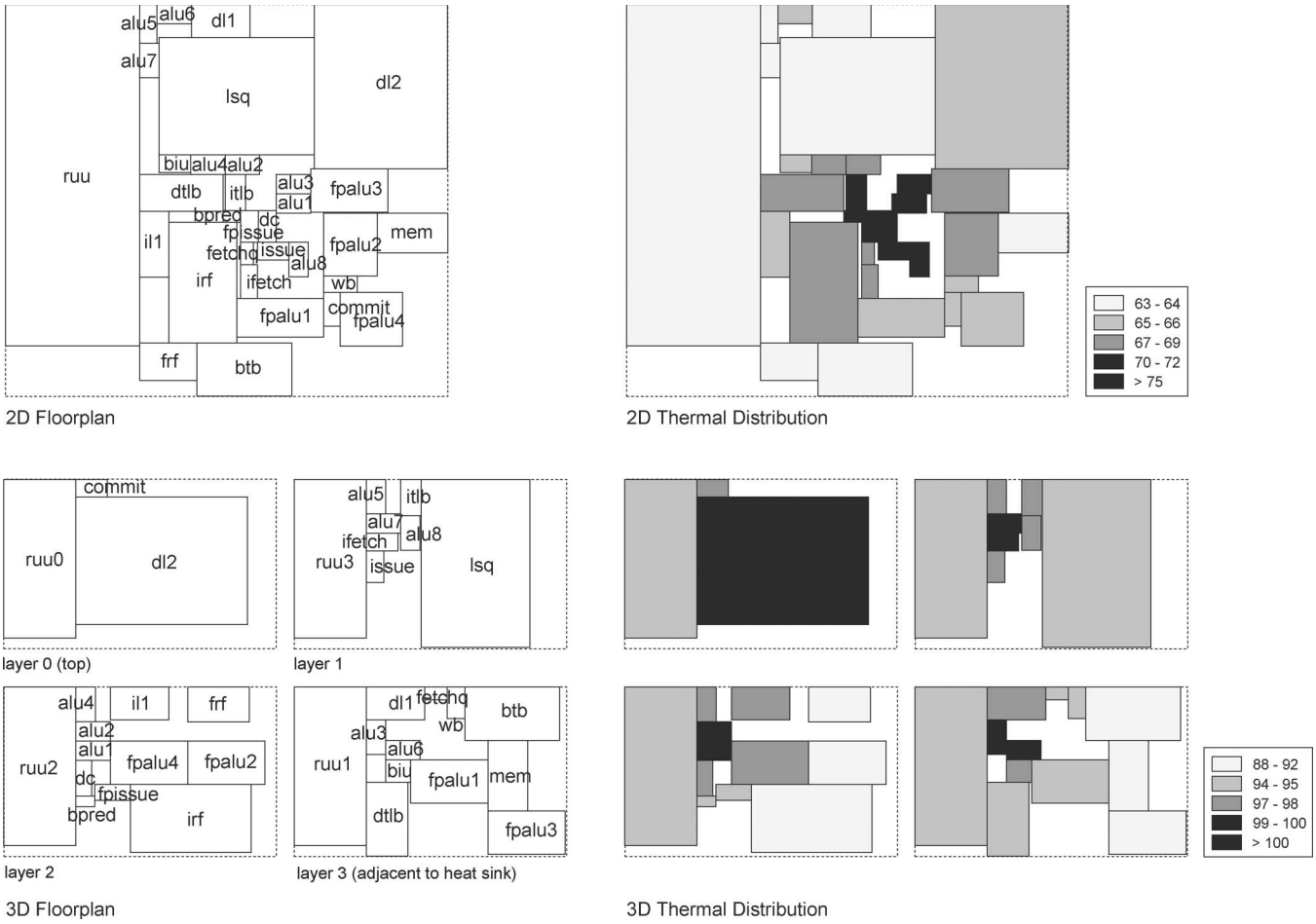

Fig. 10. Snapshots of our 2-D/3-D floorplanning.

TABLE VI
TOP TEN LIST OF BLOCKS UNDER VARIOUS METRICS

					2D floorplan				3D floorplan			
rank	area (mm^2)		power (mW/mm^2)		temperature ($^\circ C$)		leakage (mW)		temperature ($^\circ C$)		leakage (mW)	
1	RUU	16.38	IALU1	15408	IALU1	83.5	L2 cache	0.9020	IALU1	104.7	L2 cache	0.9343
2	L2 cache	7.83	BPRED	1971	ITLB	78.6	ITLB	0.2470	MEM	103.7	ITLB	0.2559
3	LSQ	6.53	COMMIT	1930	L1 icache	76.3	DTLB	0.2470	IALU5	103.1	DTLB	0.2559
4	IRF	2.94	FPISSUE	1930	FETCHQ	75.5	L1 icache	0.0588	ITLB	103.1	L1 icache	0.0609
5	BTB	1.81	ITLB	1049	FPALU1	75.4	L1 dcache	0.0588	L2 cache	102.2	L1 dcache	0.0609
6	FPALU 2	1.20	IALU2	1034	MEM	73.0	BTB	0.0088	IALU4	102.0	BTB	0.0091
7	FPALU 3	1.20	IALU3	884	COMMIT	72.5	FETCHQ	0.0035	FPALU4	101.7	FETCHQ	0.0036
8	FPALU 4	1.20	IALU4	746	IALU5	72.1	FPALU2	0.0014	IALU8	100.0	FPALU3	0.0015
9	DTLB	1.10	L1 cache	730	FPALU2	72.1	FPALU3	0.0014	IALU2	99.6	FPALU1	0.0015
10	MEM	1.00	IALU5	630	IALU7	70.8	FPALU1	0.0014	IALU3	97.3	FPALU2	0.0015

their final temperature. The leakage power profile among the modules is identical between the 2-D and 3-D floorplans except for the last two entries. This is because the logic styles of each module are more important in determining the relative leakage power than the variations in temperature. Table VII shows the top ten buses and interconnects under various metrics. It is interesting to note that the longest wire in the multilayer floorplan is almost half as long as the longest wire in the single-layer floorplan. The shortest wire list is dominated by inter-ALU connections. This is partly because the ALUs are generally small units, and so the center-to-center distance for them is smaller but also because there are many data passing lines among the ALUs so they are very tightly connected.

F. Fidelity Study

Our fidelity study is twofold. First, Table VIII shows a comparison of the temperatures provided by our 3-D mesh-based model and those provided by Hotspot v3.0 [49] across ten benchmarks. One can observe that our model provides a similar temperature. Second, we study the impact of the frequency of the thermal resistance matrix $\mathbf{R}$ update (= inversion of thermal conductance matrix) on the final temperature and IPC results. Under the "every move" column, we update $\mathbf{R}$ at every move during the SA-based refinement. The "no update" column contains the results based on our current implementation, where $\mathbf{R}$ stays constant throughout the SA refinement. Note that we

TABLE VII
TOP TEN LIST OF WIRES UNDER VARIOUS METRICS

rank	access frequency		2D floorplan wirelength (mm)		2D floorplan wirelength (mm)		3D floorplan wirelength (mm)		3D floorplan wirelength (mm)	
1	ITLB-FETCHQ	1.0	IRF-IALU5	8.575	IALU7-IALU6	0.53	IALU6-RUU	4.696	IALU1-FETCHQ	0.23
2	IF-DC	1.0	IRF-IALU1	7.132	IF-DC	0.62	FPALU3-RUU	4.479	IALU5-IALU1	0.33
3	BTB-IF	1.0	DL1-RUU	6.944	DC-ISSUE	0.62	IRF-IALU6	3.962	IALU5-IALU2	0.35
4	IL1-FETCHQ	1.0	FPALU3-RUU	6.710	IALU2-IALU3	0.65	WB-COMMIT	3.959	IALU8-IALU3	0.36
5	FETCHQ-IF	1.0	RUU-FPALU3	6.710	IALU4-IALU8	0.65	DTLB-RUU	3.688	IRF-FPALU1	0.57
6	DC-ISSUE	1.0	IRF-FPALU1	6.414	IALU4-IALU6	0.67	DL1-RUU	3.613	IALU4-IALU1	0.65
7	DL2-DL1	1.0	DL1-IALU5	6.414	ITLB-FETCHQ	0.96	IRF-IALU5	3.482	IALU8-IALU1	0.67
8	WB-COMMIT	1.0	IRF-IALU7	5.797	IL1-FETCHQ	1.00	IRF-IALU2	3.462	IALU2-IALU1	0.67
9	DTLB-RUU	1.0	IALU6-RUU	5.730	IALU4-IALU7	1.16	RUU-FPALU1	3.423	IALU2-IALU4	0.67
10	DL1-RUU	1.0	DL2-IL1	5.659	IALU6-IALU8	1.33	DL2-IL1	3.395	IALU4-IALU5	0.69

TABLE VIII
COMPARISON WITH HOTSPOT V3.0 [49]

bench	HotSpot	Ours	bench	HotSpot	Ours
equake	86.1	86.2	gzip	109.6	104.7
mcf	86.5	86.6	bzip2	112.7	107.4
swim	88.3	88.0	vpr	123.2	114.4
art	93.6	92.0	twolf	130.0	118.9
gap	97.2	94.5	lucas	134.6	123.0

TABLE IX
IMPACT OF THE FREQUENCY OF THERMAL RESISTANCE MATRIX UPDATE ON IPC AND TEMPERATURE. WE USE 3-D LP + SA FLOORPLANNER WITH A + P + T OBJECTIVES

	every move		no update	
bench	IPC	temp	IPC	temp
gzip	2.76	108.2	2.85	104.7
swim	0.74	89.5	0.92	88.0
vpr	1.08	115.2	1.29	114.4
art	0.52	95.6	0.61	92.0
mcf	0.09	89.2	0.07	86.6
equake	0.55	87.6	0.67	86.2
lucas	0.87	128.7	1.19	123.0
gap	1.60	98.4	1.61	94.5
bzip2	1.95	109.6	2.33	107.4
twolf	0.79	122.4	1.02	118.9
AVG	1.10	104.4	1.26	101.6
AREA	27.89		26.45	
WIRE	245.68		247.02	
TIME	433529		20016	

update $\mathbf{R}$ every time we add a slicing cutline in our LP-based floorplan construction. From Table IX, we observe that the accurate computation of temperature values (= updating $\mathbf{R}$ at every move) does not necessarily translate into better results. In fact, we obtained comparable IPC and thermal results within a fraction of runtime with our "no update" method. Thus, we conclude that our thermal analysis and the way we make use of it in SA optimization prove to be highly effective and efficient.

VII. CONCLUSION

In this paper, we presented the first multiobjective "microarchitecture-level" floorplanning algorithm for high-performance high-reliability microprocessors targeting both 2-D and 3-D ICs. We simultaneously considered performance and thermal objectives such that our automated floorplanner can provide a balanced or goal-directed processor organization that achieves user-specified design objectives. Moreover, we integrated leakage modeling into our thermal analyzer and monitored the temperature/leakage interaction to prevent thermal runaway. We investigated how vertical overlap among the modules in 3-D floorplanning affects the performance, thermal, and area objectives. In addition, we partitioned the modules into multiple layers while considering the through-via requirements for F2F and F2B bonding styles. Our hybrid approach that combines LP and SA proved to be very effective in obtaining a high-quality solution in a short runtime.

REFERENCES

[1] C. Long, L. Simonson, W. Liao, and L. He, "Floorplanning optimization with trajectory piecewise-linear model for pipelined interconnects," in *Proc. ACM Des. Autom. Conf.*, 2004, pp. 640–645.

[2] J. Cong, A. Jagannathan, G. Reinman, and M. Romesis, "Microarchitecture evaluation with physical planning," in *Proc. ACM Des. Autom. Conf.*, 2003, pp. 32–35.

[3] M. Casu and L. Macchiarulo, "Floorplanning for throughput," in *Proc. Int. Symp. Phys. Des.*, 2004, pp. 62–69.

[4] M. Ekpanyapong, J. Minz, T. Watewai, H.-H. Lee, and S. K. Lim, "Profile-guided microarchitectural floorplanning for deep submicron processor design," in *Proc. ACM Des. Autom. Conf.*, 2004, pp. 634–639.

[5] V. Nookala, Y. Chen, D. Lilja, and S. Sapatnekar, "Microarchitecture-aware floorplanning using a statistical design of experiments approach," in *Proc. ACM Des. Autom. Conf.*, 2005, pp. 579–584.

[6] V. Agarwal, M. S. Hrishikesh, S. W. Keckler, and D. Burger, "Clock rate versus IPC: The end of the road for conventional microarchitectures," in *Proc. IEEE Int. Conf. Comput. Architecture*, 2000, pp. 248–259.

[7] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proc. IEEE*, vol. 89, no. 4, pp. 490–504, Apr. 2001.

[8] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proc. IEEE Int. Conf. Comput. Architecture*, 2003, pp. 2–13.

[9] M. Huang, J. Renau, S.-M. Yoo, and J. Torrellas, "A framework for dynamic energy efficiency and temperature management," in *Proc. 33rd Annu. ACM/IEEE Int. Symp. Microarchitecture*, Monterey, CA, 2000, pp. 202–213.

[10] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proc. 7th Int. Symp. High-Performance Comput. Architecture*, 2001, p. 171.

[11] S. Dropsho, V. Kursun, D. Albonesi, S. Dwarkadas, and E. Friedman, "Managing static leakage energy in microprocessor functional units," in *Proc. Annu. Int. Symp. Microarchitecture*, 2004, pp. 321–332.

[12] N. Kim, K. Flautner, D. Blaauw, and T. Mudge, "Drowsy instruction caches: Leakage power reduction using dynamic voltage scaling and cache sub-bank prediction," in *Proc. Annu. Int. Symp. Microarchitecture*, 2002, pp. 219–230.

[13] D. Duarte, Y. Tsai, N. Vijaykrishnan, and M. Irwin, "Evaluating run-time techniques for leakage power reduction," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2002, pp. 31–38.

[14] S. Kaxiras, Z. Hu, and M. Martonosi, "Cache decay: Exploiting generational behavior to reduce cache leakage power," in *Proc. 28th Annu. Int. Symp. Comput. Architecture*, Goteborg, Sweden, 2001, pp. 240–251.

[15] L. He, W. Liao, and M. Stan, "System level leakage reduction considering leakage and thermal interdependency," in *Proc. ACM Des. Autom. Conf.*, 2004, pp. 12–17.

[16] C. Tsai and S. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 2, pp. 253–266, Feb. 2000.

[17] C. N. Chu and D. F. Wong, "A matrix synthesis approach to thermal placement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 17, no. 11, pp. 1166–1174, Nov. 1998.

[18] W. Hung, Y. Xie, N. Vijaykrishnan, C. Addo-Quaye, T. Theocharides, and M. Irwin, "Thermal-aware floorplanning using genetic algorithms," in *Proc. Int. Symp. Quality Electron. Des.*, 2005, pp. 634–639.
[19] B. Obermeier and F. Johannes, "Temperature-aware global placement," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2004, pp. 143–148.
[20] G. Chen and S. Sapatnekar, "Partition-driven standard cell thermal placement," in *Proc. Int. Symp. Phys. Des.*, 2003, pp. 75–80.
[21] K. Balakrishnan, V. Nanda, S. Easwar, and S. K. Lim, "Wire congestion and thermal aware 3-D global placement," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2005, pp. 1131–1134.
[22] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3-D ICs," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, 2004, pp. 306–313.
[23] S. Das, A. Chandrakasan, and R. Reif, "Design tools for 3-D integrated circuits," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2003, pp. 53–56.
[24] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3-D ICs using a force directed approach," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, 2003, pp. 86–89.
[25] T. Tanprasert, "An analytical 3-D placement that reserves routing space," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2000, pp. 69–72.
[26] R. Zhang, K. Roy, C.-K. Koh, and D. B. Janes, "Exploring SOI device structures and interconnect architectures for 3-dimensional integration," in *Proc. ACM Des. Autom. Conf.*, 2001, pp. 846–851.
[27] J. Minz, E. Wong, and S. K. Lim, "Thermal and power integrity-aware floorplanning for 3-D circuits," in *Proc. IEEE Int. SOC Conf.*, 2005, pp. 81–82.
[28] L. Cheng, W. Hung, G. Yang, and X. Song, "Congestion estimation for 3-D circuit architectures," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 51, no. 12, pp. 655–659, Dec. 2004.
[29] B. Goplen and S. Sapatnekar, "Thermal via placement in 3-D ICs," in *Proc. Int. Symp. Phys. Des.*, 2005, pp. 167–174.
[30] V. Pavlidis and E. Friedman, "Interconnect delay minimization through interlayer via placement in 3-D ICs," in *Proc. Great Lakes Symp. VLSI*, 2005, pp. 20–25.
[31] I. Kaya, M. Olbrich, and E. Barke, "3-D placement considering vertical interconnects," in *Proc. IEEE Int. SOC Conf.*, 2003, pp. 257–258.
[32] Y. Deng and W. Maly, "Physical design of the 2.5D stacked system," in *Proc. IEEE Int. Conf. Comput. Des.*, 2003, pp. 211–217.
[33] J. Cong and Y. Zhang, "Thermal-driven multilevel routing for 3-D ICs," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2005, pp. 121–126.
[34] L. Cheng, L. Deng, and M. Wong, "Floorplan design for 3-D VLSI design," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2005, pp. 405–411.
[35] J. Minz, S. K. Lim, and C. K. Koh, "3-D module placement for congestion and power noise reduction," in *Proc. Great Lakes Symp. VLSI*, 2005, pp. 458–461.
[36] G. Sohi and S. Vajapeyam, "Instruction issue logic for high performance interruptable pipelined processors," in *Proc. 14th Annu. Int. Symp. Comput. Architecture*, 1987, pp. 27–34.
[37] D. Duarte, N. Vijaykrishnan, and M. J. Erwin, "A clock power model to evaluate the impact of architectural and technology optimizations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 6, pp. 844–855, Dec. 2002.
[38] Y. Tsai, A. Ankadi, N. Vijaykrishnan, M. Irwin, and T. Theocharides, "ChipPower: An architecture-level leakage simulator," in *Proc. IEEE Int. SOC Conf.*, 2004, pp. 395–398.
[39] *eCACTI*. [Online]. Available: http://www.ics.uci.edu/~maheshmn/eCACTI/main.htm
[40] J. C. Eble, V. K. De, D. S. Wills, and J. D. Meindl, "A generic system simulator (GENESYS) for ASIC technology and architecture beyond 2001," in *Proc. Int. ASIC Conf.*, 1996, pp. 193–196.
[41] W. Liao, F. Li, and L. He, "Microarchitecture level power and thermal simulation considering temperature," in *Proc. Int. Symp. Low Power Electron. and Des.*, 2003, pp. 211–216.
[42] P. Shivakumar and N. P. Jouppi, "CACTI 3.0: An integrated cache timing, power, and area model," HP Western Res. Labs, Palo Alto, CA, Tech. Rep. 2001.2, 2001.
[43] T. M. Austin, *Simplescalar Tool Suite*. [Online]. Available: http:/www.simplescalar.com
[44] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proc. IEEE Int. Conf. Comput. Architecture*, 2000, pp. 83–94.
[45] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "Rectangle packing based module placement," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, 1995, pp. 472–479.
[46] S. B. Horn, "Vertically integrated sensor arrays VISA," in *Proc. Defense and Security Symp.*, 2004, pp. 332–340.
[47] M. Umemoto, K. Tanida, Y. Nemoto, M. Hoshino, K. Kojima, Y. Shirai, and K. Takahashi, "High-performance vertical interconnection for high-density 3-D chip stacking package," in *Proc. IEEE Electron. Compon. and Technol. Conf.*, 2004, pp. 616–623.
[48] S. Salewski and E. Barke, "An upper bound for 3-D slicing floorplans," in *Proc. Asia and South Pacific Des. Autom. Conf.*, 2002, pp. 567–572.
[49] *HotSpot*. [Online]. Available: http://lava.cs.virginia.edu/HotSpot

Michael Healy (S'06) received the B.S. degree in computer engineering and the M.S. degree in electrical and computer engineering, in 2004 and 2006, respectively, from the Georgia Institute of Technology, Atlanta. He is currently working toward the Ph.D. degree at the Georgia Institute of Technology.

From 1999 to 2001, he was with the Texas Academy of Mathematics and Science, University of North Texas. He is currently with the School of Electrical and Computer Engineering, Georgia Institute of Technology. His research interests include physical design automation, microarchitecture, and power and thermal reliability.

Mario Vittes (S'00) received the B.S. degree in computer engineering and the M.S. degree in electrical and computer engineering, in 2003 and 2005, respectively, from the Georgia Institute of Technology, Atlanta.

He is currently with Intel Corporation, Sta. Clara, CA. His research interests include computer architecture and power and thermal modeling.

Mongkol Ekpanyapong (S'04) received the B.E. degree from the Computer Engineering Department, Chulalongkorn University, Bangkok, Thailand, in 1997, the M.E. degree from the Computer Science Department, Asian Institute of Technology, Pathumthani, Thailand, in 2000, the M.S. degree from the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, 2003 and 2006, respectively.

He is currently with the School of Electrical and Computer Engineering, Georgia Institute of Technology. His research interests include physical very large scale integration design, computer architecture, and compiler.

Chinnakrishnan S. Ballapuram received the M.S. degree from the Georgia Institute of Technology, Atlanta, in 2006. He is currently working toward the Ph.D. degree at Georgia Institute of Technology.

He is with the Microarchitectural Research Society (MARS), Georgia Institute of Technology. His research interests include microarchitecture and very large scale integration design.

Sung Kyu Lim (S'94–M'00–SM'05) received the B.S., M.S., and Ph.D. degrees from the Computer Science Department, University of California at Los Angeles (UCLA), in 1994, 1997, and 2000, respectively.

From 2000 to 2001, he was a Post-Doctoral Scholar with the UCLA and a Senior Engineer with Aplus Design Technologies, Inc. He was an Assistant Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, in August 2001, and an Adjunct Assistant Professor with the College of Computing in September 2002. He is currently the Director of the Computer Aided Design Laboratory, Georgia Institute of Technology. His research focus is on the physical design automation for three-dimensional (3-D) circuits, 3-D system-on-packages, microarchitectural physical planning, field programmable analog arrays, and quantum cell automata.

Dr. Lim has been on the advisory board of the Special Interest Group on Design Automation, Association for Computing Machinery (ACM) since 2003. He has served the technical program committee of the IEEE International Symposium on Circuits and Systems, ACM Great Lakes Symposium on VLSI, IEEE International Conference on Computer Design, ACM International Symposium on Physical Design, and ACM/IEEE Asia and South Pacific Design Automation Conference. He has been awarded a Design Automation Conference Graduate Scholarship in 2003 and a National Science Foundation Faculty Early Career Development (CAREER) Award in 2006.

Hsien-Hsin S. Lee (M'96) received the Ph.D. degree in computer science and engineering from the University of Michigan, Ann Arbor.

He was a Senior Computer Architect with Intel and an Architecture Manager with StarCore DSP Center, Agere Systems. He is currently an Assistant Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. He holds four U.S. patents. His research interests include computer architecture, low-power circuits, information security, and three-dimensional ICs.

Dr. Lee is a member of Tau Beta Pi and the Association for Computing Machinery. His doctoral thesis was awarded the Horace H. Rackham School Distinguished Dissertation Award at the University of Michigan. He has coauthored three papers that won Best Paper Awards at MICRO-33, CASES-04, and IBM PAC^2. More recently, he received the Department of Energy Early CAREER Award and was named the recipient of the 2006 ECE Outstanding Jr. Faculty Member Award at Georgia Institute of Technology.

Gabriel H. Loh (M'04) received the B.E. degree in electrical engineering from Cooper Union, New York, NY, in 1998, and the M.S. and Ph.D. degrees in computer science from Yale University, New Haven, CT, in 1999 and 2002, respectively.

From 2003 to 2004, he was a Senior Researcher with the Microarchitecture Research Laboratory, Intel Corporation. He is currently an Assistant Professor with the College of Computing, Georgia Institute of Technology, Atlanta. His research interests include computer architecture, processor microarchitecture, simulation, circuit design, and three-dimensional integration technology.

Dr. Loh is a member of Tau Beta Pi, Eta Kappa Nu, and the Association for Computing Machinery.