Thermal-aware 3D Microarchitectural Floorplanning

Mongkol Ekpanyapong, Michael B. Healy, Chinnakrishnan S. Ballapuram, Sung Kyu Lim, and Hsien-Hsin S. Lee
School of Electrical and Computer Engineering
Georgia Institute of Technology

Gabriel H. Loh
College of Computing
Georgia Institute of Technology

***Abstract*— Next generation deep submicron processor design will need to take into consideration many performance limiting factors. Flip flops are inserted in order to prevent global wire delay from becoming non-linear, enabling deeper pipelines and higher clock frequency. The move to 3D ICs will also likely be used to further shorten wirelength. This will cause thermal issues to become a major bottleneck to performance improvement. In this paper we propose a floorplanning algorithm which takes into consideration both thermal issues and profile weighted wirelength using mathematical programming. Our profile-driven objective improves performance by 20% over wirelength-driven. While the thermal-driven objective improves temperature by 24% on average over the profile-driven case.**

I. Introduction

In next generation deep submicron processor design it is likely that repeaters will be inserted frequently on global wires to prevent wire delay from becoming non-linear [1]. Flip-flop insertion is a technique used to alleviate the impact of wire delay to achieve a target clock frequency. A deeper pipeline enabled by flip-flop insertion results in a higher clock frequency and higher BIPS (billions of instructions per second) [2]. Nevertheless, the improvement cannot always be anticipated; especially for designs with small feature size; flip-flop insertion may cause IPC degradation from its increased latency. Therefore, inserting flip-flops without a meticulous measure does not guarantee an overall performance improvement.

One technique that can alleviate IPC (Instructions per Cycle) degradation resulting from wire delay is communication aware floorplanning [3], [4], [5], [6]. Using floorplanners that consider the impact of wire delay by trying to move heavily communicating modules closer together can shorten latency on such paths and result in better performance improvement. Another technique is to move to three dimensional integrated circuits or 3D ICs. By moving to 3D ICs, total wirelength can be reduced and clock speed can be increased as shown in [7]. One bottleneck to the adoption of 3D ICs is heat dissipation. The structure of 3D ICs inherently implies that moving heat from the center of the chip will be more difficult. This can result in more complex cooling devices, circuit malfunctions, and shorter circuit life time. When designing ICs with many layers of transistors stacked together thermal issues become a large concern. In this paper, we propose a floorplanning algorithm that considers performance, area, and thermal issues using a mathmatical programming approach utilizing information gathered from cycle-accurate simulation.

Some recent works on wire-delay issues on microarchitectural design include [8], [5], [9], [2], [10], [11], [6]. Recent work on physical design for microarchitecture include [12], [4], [3]. Recent work on thermal-aware physical design algorithms include [13], [14], [15], [16], [17], [18].

The structure of this paper is as follows: Section II presents the problem formulation. Section III details our 3D thermal analysis technique. Section IV shows our infrastructure for cycle-accurate simulation. Section V presents our floorplanning algorithm. Finally, section VI shows our experimental results and we conclude in Section VII.

II. Problem Formulation

A. Design Flow

An overview of our profile-driven microarchitectural floorplanning is shown in Figure 1. Our framework combines technology scaling parameters and the execution profiling information of applications to guide the floorplanning step of a given microarchitecture design. First, a machine description is provided as input to the microarchitecture simulator, where profiling counters were instrumented for book-keeping module-to-module communication. Then a cycle-accurate simulation is performed using Simplescalar [19] to collect and extract the amount of interconnection traffic between modules for a given benchmark program. The microarchitecture simulator was integrated with Wattch [20] to provide the power numbers that are used to drive the 3D-thermal analyzer. For cache-like or buffer-like structures, the area and module delay are estimated using an industry tool from HP Western Research Labs called CACTI [21]. For scaling other structures such as ALUs, we use GENESYS [22] developed at the Georgia Institute of Technology.

After the timing, area, and access frequency information of each module is collected, we feed the module-level netlist, statistical interconnection traffic, and a target processor frequency into our thermal/profile-guided floorplanner. The power consumption of all the functional units are fed to the 3D-thermal analyzer to generate the thermal profile. The 3D-floorplanner takes in the netlist and the temperature information to generate a floorplan that maximizes the performance under the thermal and frequency constraints. The new floorplan is fed back to the 3D-thermal analyzer, along with the power numbers to generate a new thermal profile. With these new latency values architecture performance simulation is performed to obtain more realistic and accurate IPC and BIPS numbers. Few iterations take place before an optimum floorplan for the given constraint is achieved.

B. Problem Formulation

Given a set of microarchitectural modules and a netlist that specifies the connectivity among these modules, our thermal- and profile-driven microarchitectural floorplanner tries to place each module such that (i) there is no overlap among the modules, and (ii) a user-specified clock period constraint is satisfied. Our objective is to minimize the maximum temperature among all blocks and the overall execution time of a given processor. Because clock frequency is fixed, IPC (Instructions Per Cycle) is used for the performance measurement. IPC represents the average number of instructions that can be issued in one clock cycle. In VLSI circuit design clock period

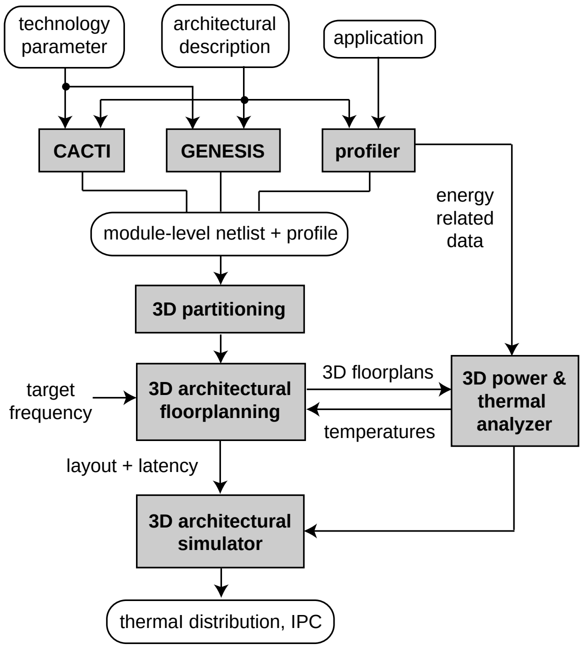
technology parameter
architectural description
application
CACTI
GENESIS
profiler
module-level netlist + profile
energy related data
3D partitioning
target frequency
3D architectural floorplanning
3D floorplans
temperatures
3D power & thermal analyzer
layout + latency
3D architectural simulator
thermal distribution, IPC

Fig. 1. Overview of our thermal-aware 3D microarchitectural floorplanning framework.

is used to evaluate the quality of Logic Synthesis and Physical Design solutions. This is equivalent to the longest path delay. F (Clock frequency), which denotes the total number of cycles per second, is the reciprocal of clock period. Finally, $BIPS = IPC \times F$ where F is in giga-hertz. In this paper, we maximize IPC under a clock frequency constraint so that overall BIPS is maximized.

III. 3D Thermal Analysis

A. 3D Power Analysis

While collecting the interconnection traffic between modules power consuption for all the functional blocks was gathered cumulatively and stored for every hundred thousand cycles. This traffic activity collection and the dynamic power consumpton is collected only once during the whole experiment cycle. These power numbers are fed to the 3D-thermal analyzer that gives the thermal profile. The 3D-floorplanner gives a new floorplan for the current thermal profile and module netlist based on the constraints. The new floorplan may have a different interconnect length between the modules. Therefore, interconnect power is calculated again based on the new lengths and added to the dynamic power consumption that was collected earlier. Wattch currently does not model global interconnect power. Hence it was modeled as a seperate module and is added to the dynamic power consumption. Here, it is safely assumed that the dynamic power consumption still remains the same even with different floorplans, as the activity factor does not change with the position of the modules. The activity factor is dependent on the program behavior rather than the position of the modules.

B. 3D Thermal Analysis

For thermal analysis we use a 3D resistor mesh [13], [23] as shown in Figure 2. Our floorplanning algorithm does not require precise temperature values to successfully minimize chip temperature profile. Therefore, we use a simplified thermal model during the floorplanning process to speed up the thermal calculation. Because we call the thermal model many times during floorplanning, this can dramatically improve runtime. From this thermal model we get temperature numbers that are relatively accurate. This relativism is all we need for our optimization process. This model uses a non-uniform 3D themal resistor mesh where grid lines are defined at the centers of each architectural block being considered. 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 define the themal 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 that block covers. After the floorplanning is done we use a slower but more accurate finely spaced uniform grid to report the final temperature profile.

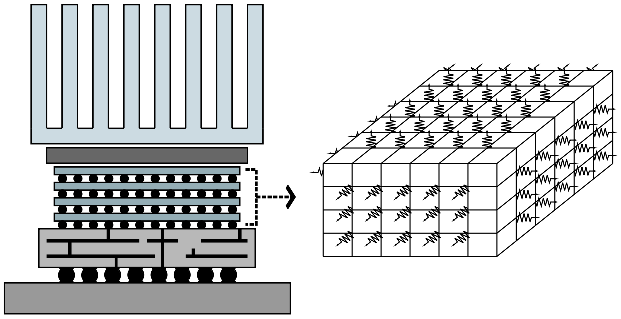

Fig. 2. 3D grid of a chip for thermal modeling

IV. Simulation Infrastructure

The detailed microarchitecture used in our experiment is illustrated in Figure 3. Each functional block represents a module used by our floorplanner. For accurate performance prediction and optimization wires can no longer be isolated from architecture-level evaluation but must be modeled as *units* that consume power and have delays. Therefore, provisions were made to consider wire delays in our simulator. The existing simulator assumes that the communication latency between functional blocks is always one cycle; this no longer holds while operating at extremely high frequency given the increased wire delays and ever-growing die areas. For performance evaluation we use the information provided by the floorplanner to derive essential simulation parameters such as pipeline depth and communication/forwarding latencies. The inter module latency is a function of the distance and number of flip flops between modules. If the floorplan has been optimized for clock speed, the pipeline depth of the processor reflects it. In our experiments, we expect an improvement in performance (from architectural simulation) if the frequency of forwarding traffic between units are included in our floorplan formulation. The profile-driven floorplanning tries to place highly communicating modules closer together, minimizing their latencies as a function of distance.

The approch used here is general enough to take in many different configuartions. For the sake of expediency one configuration was chosen for experimentation. This configuration is enumerated here. The machine width is 8. We use 512 entry branch prediction, branch table, and reorder buffers, 8 KB Level 1 instruction and data cache, 128 KB Level 2 unified cache, 2 MB Level 3 unified cache, 128 entry instruction and data transfer look-aside buffers, 8 ALUs, 4 FPUs, and 128 entry load store queue.

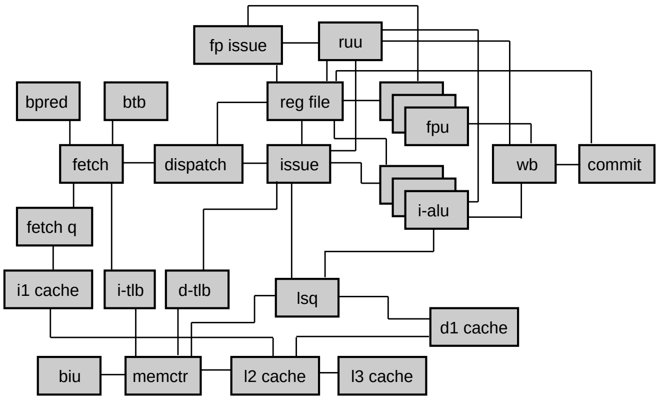

Fig. 3. Processor microarchitecture model.

V. Thermal-aware 3D Partitioning and Floorplanning

A. 3D Partitioning

We perform partitioning using the FM [24] algorithm. Due to the imbalance between module sizes some modules, such as the L3 cache, have to be removed before calling the FM algorithm. To improve the IPC of the system modules that have a lot of communication must be placed in different layer so that access time can be reduced. To accomplish this we first create pseudo edges between all modules. If two modules have no connection to each other the weight of that pseudo edge is set to one. If two modules have a lot of normalized traffic (the maximum number is one) λ, we set the weight of this pseudo edge to be $1 - \lambda$. Then we call the FM algorithm to balance each partition and minimize this pseudo weight.

B. MILP (Mixed Integer Linear Program)-based 3D Floorplanning

Figure 4 shows the MILP formulation of our thermal-aware 3D microarchitectural floorplanner. The variables used in this formulation are enumerated below. The objective of our ILP formulation (= Equation (1) is a weighted sum of performance-, thermal-, and area-related terms. The *weighted delay* of an edge (i, j) is defined to be $\lambda_{ij} \cdot z_{ij}$, where the weight λ_{ij} is based on module access frequency. z_{ij} is the number of FFs on the wire. The second term in our objective is thermal-related. We try to separate two hot blocks as much as possible by minimizing $(1 - T_{ij})(X_{ij} + Y_{ij})$. The final term minimizes area, where X_{max} is the maximum among all x values. Since minimizing $X_{max}Y_{max}$ is non-linear, we try to minimize only X_{max}. By letting A be the aspect ratio of the chip AX_{max} is greater than all y values. U_1, U_2, and U_3 are user defined parameters to weight among performance, thermal, and area objectives, respectively.

Let N denote the set of all flexible modules, and E denote the set of directed edges, where a directed edge (i, j) represents a wire from module i to module j. Let H be the height (maximum number of layers) of the 3D IC. Let T_{ij} denote the normalized product of the temperature of module i and j. Let α be the repeated wire delay per mm[1], β is via delay[2], and λ_{ij} be the statistical traffic on wire (i, j), i.e., the normalized access counts from module i to module j. g_i is the delay of module i. $w_{min,i}$ and $w_{max,i}$ denote the minimum and maximum half width of module i, respectively. The area of module i is denoted by a_i. Finally, f_{ij} is the number of flip-flops on wire (i, j) in the given microarchitectural design.

Let C (= $1/F$) denote the target cycle period of the given microarchitectural design, which is an input to our floorplanner. In the MILP model, we need to determine the values for the following decision variables: x_i, y_i, l_i, w_i, h_i, and z_{ij}. Let (x_i, y_i) denote the location of the center of module i in $\mathbb{R}^2_+$ space. l_i is the level of module i where there are from 1 to H levels. X_{ij}, Y_{ij}, and L_{ij} represent $|x_i - x_j|$, $|y_i - y_j|$, and $|l_i - l_j|$ between module i and j, respectively. X_{max} is the maximum value among all x_{ij}. A is the aspect ratio of the chip. z_{ij} is the number of flip-flops on wire (i, j) after FF insertion. w_i and h_i denote the half width and the half height of module i, respectively.

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 $d(i, j) = \alpha(X_{ij} + Y_{ij}) + \beta L_{ij}$. Then, $g_i + d(i, j)$ represents the latency of module i accessing module 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 value on x, y, and l distance are given in (3)–(5). To minimize the total area (6) constrains maximum value of all x and y locations by assuming that the aspect ratio of the chip is A. Constraint (7) requires that we do not remove any existing FFs from the wires. Constraints (8)–(11) represent relative positions among the modules and are used to guarantee that two modules will not be overlapped. (12) calculates e_{ij}. If e_{ij} is one then two modules are in diffent levels and can be overlapped otherwise the modules are in the same level and cannot be overlapped. Constraint (13) specifies the possible range of the half width of each module. (14) is a non-negative constraint for the module location. l_i represents the level of modules i as shown in (15). (16) states that (c_{ij}, d_{ij}, e_{ij}) are binary variables. c_{ij}, d_{ij}, e_{ij} denote the relative positions of the blocks, i.e. left, right, top, bottom, up, down. Finally, (17) specifies that the number of flip-flops must be an integer. Also note that M is a sufficiently large number.

C. Linear Relaxation

The MILP floorplanning problem is NP-hard and requires prohibitive runtime to obtain a legal solution. Specifically, z_{ij}, c_{ij}, d_{ij}, e_{ij}, and l_{ij} are the only integer variables. To remedy this problem, we relax MILP into Linear Programming (LP) model as follows.[3] We adopt a partitioning method similar to the one described in [25] to obtain l_{ij}. To relax the integrality while maintaining the feasibility and staying close to the optimal solution we first relax the integrality of z_{ij} to be a real number. We also solve several linear programming problems to determine the relative positions among the modules, i.e., c_{ij} and d_{ij}. If these c_{ij}, d_{ij}, and e_{ij} are used in Equation (10) and (11), and l_{ij} and z_{ij} can take real values, our MILP model shown in Figure 4 becomes a Linear Program.

Our thermal-aware 3D floorplanning algorithm consists of multiple iterations, where at each iteration a cutline is inserted to divide a region (alternatively called a block) into two sub-blocks. We start the algorithm by creating a large block containing all modules for each layer. At each iteration, we choose a block, divide it into two sub-blocks, and perform module floorplanning again so that the thermal/performance objective is further minimized. At the beginning of each iteration, we call thermal analysis to get temperature profile. Note that because of the bipartitioning method, the number of call to

[1] Based on the predicted values of resistance, capacitance and other parasitic parameters from [1], repeated wire delay is approximated to be $80pS/mm$ for $30nm$ technology. Note that a FO4 gate delay for $30nm$ is approximately $17pS$.

[2] The via length and β are small, hence via delay is negligible

[3] If a near-optimal solution is required, MILP is the better approach than LP. However, ILP in general requires an excessive amount of computational power to solve.

MILP Formulation

$$\text{Minimize} \quad \sum_{(i,j)\in E} (U_1 \times \lambda_{ij} z_{ij} + U_2 \times (1 - T_{ij})(X_{ij} + Y_{ij}) + U_3 \times X_{max}) \tag{1}$$

Subject to:

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

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

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

$$L_{ij} \geq l_i - l_j \text{ and } L_{ij} \geq l_j - l_i,\ (i,j) \in E \tag{5}$$

$$X_{max} \geq x_i \text{ and } A\ X_{max} \geq y_i,\ i \in N \tag{6}$$

$$z_{ij} \geq f_{ij},\ (i,j) \in E \tag{7}$$

$$x_i + w_i \leq x_j - w_j + M(c_{ij} + d_{ij} + e_{ij}),\ i < j \in N \tag{8}$$

$$x_i - w_i \geq x_j + w_j - M(1 + c_{ij} - d_{ij} + e_{ij}),\ i < j \in N \tag{9}$$

$$y_i + m_i w_i + k_i \leq y_j - m_j w_j - k_j + M(1 - c_{ij} + d_{ij} + e_{ij}),\ i < j \in N \tag{10}$$

$$y_i - m_i w_i - k_i \geq y_j + m_j w_j + k_j - M(2 - c_{ij} - d_{ij} + e_{ij}),\ i < j \in N \tag{11}$$

$$e_{ij} \geq \frac{L_{ij}}{H} \text{ and } e_{ij} \leq L_{ij},\ (i,j) \in E \tag{12}$$

$$w_{min,i} \leq w_i \leq w_{max,i},\ i \in N \tag{13}$$

$$x_i, y_i \geq 0,\ i \in N \tag{14}$$

$$l_i \in \{1, 2, ..., H\},\ i \in N \tag{15}$$

$$c_{ij}, d_{ij}, e_{ij} \in \{0, 1\},\ i < j \in N \tag{16}$$

$$z_{ij} \in \mathbb{N},\ (i,j) \in E \tag{17}$$

Fig. 4. Mixed integer linear programming model of our thermal-aware 3D microarchitectural floorplanning.

```
Thermal-aware 3D Floorplanning

partition blocks into layers;
Initialize $B(1) = \{1\}, M_1(1) = N$;
for ($u = 1$ to $log_2 N$)
    Call Thermal Analysis
    for (count =1 to run)
        Choose a block $j$ and divide it into two;
        Specify $S_{jk}(u), (\bar{x}_{jk}, \bar{y}_{jk})$;
        Solve LP with cutline $u$;
        Update $B(u+1), M_j(u+1), r_j, v_j, t_j, b_j$;
    Project best solution for $u + 1$;
Obtain $c_{ij}, d_{ij}$ from prior slicing floorplan;
Solve MILP;
return $x_i$, $y_i$, $w_i$, $h_i$, $z_{ij}$;
```

Fig. 5. Description of our thermal-aware 3D floorplanning algorithm. We perform a top-down recursive bipartitioning and solve LP-based floorplanning at each iteration. We then solve MILP again after the last iteration using the slicing floorplanning result.

the temperature analysis can be minimized to only $logN$. During this process, the modules in the chosen block should be enclosed by the block boundaries; then the area-weighted mean (= center of gravity) among all modules in each sub-block corresponds the center of the sub-block. In addition, the user specified clock period C constraint needs to be satisfied, i.e., the longest combinational path delay should be less than C. We terminate the for loop when each block contains exactly one module. Lastly, we obtain the relative positions among the modules from the slicing floorplanner result and solve MILP (shown in Figure 4) again. This time, however, the MILP formulation becomes an LP since c_{ij} and d_{ij} are already determined and z_{ij} are still allowed to have non-integer values.

Figure 5 shows a description of our LP-based 3D slicing floorplanning algorithm. First we perform partitioning to obtain l_{ij}. $B(u)$ denotes the set of all blocks at iteration u, and $M_j(u)$ denotes the set of all modules currently in block j at iteration u. $S_{jk}(u)$ is the set of modules assigned to the center of sub-block k ($k \in \{1, 2\}$) contained in block j at iteration u. We denote the center of sub-block k contained in block j by $(\bar{x}_{jk}, \bar{y}_{jk})$. Finally, let r_j, v_j, t_j, b_j denote the right, left, top, and bottom boundary of block j. Note that each iteration can be repeated multiple times to obtain different slicing floorplans. This is due to the fact that there exists 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 the total weighted wirelength for the next iteration. After the final slicing floorplan is obtained, we solve MILP again using c_{ij}, d_{ij} we obtained from the slicing floorplan to obtain a more compact solution. We return x_i, y_i, w_i, h_i, and z_{ij} as the final results of the thermal-aware 3D microarchitectural floorplanning.

Figure 6 shows the LP formulation for our thermal-aware 3d microarchitectural floorplanning, which is used at each iteration of our recursive bipartitioning-based (= slicing) floorplanning. At each iteration, a new cutline is inserted to divide a block into two sub-blocks while minimizing the total weighted wirelength is minimized. The block boundary constraints (18)–(21) require that all modules in the block be enclosed by these block boundaries. The center of gravity constraints (22)–(23) require that the area-weighted mean (= center of gravity) among all modules in each sub-block corresponds the center of the sub-block. We also develop another LP-based floorplanning model in which we minimize the total wirelength and

LP Formulation

$$\text{Minimize} \quad \sum_{(i,j)\in E} (U_1 \times \lambda_{ij} z_{ij} + U_2 \times (1 - T_{ij})(X_{ij} + Y_{ij}))$$

Subject to (2)—(7), (13)—(14) and the following:
If module i and j are in the same level

$$\begin{aligned} x_i + w_i &\leq r_j,\ i \in M_j(u), j \in B(u) && (18)\\ x_i - w_i &\geq v_j,\ i \in M_j(u), j \in B(u) && (19)\\ y_i + m_i w_i + k_i &\leq t_j,\ i \in M_j(u), j \in B(u) && (20)\\ y_i - m_i w_i - k_i &\geq b_j,\ i \in M_j(u), j \in B(u) && (21) \end{aligned}$$

And for $k \in \{1,2\}, j \in B(u)$

$$\begin{aligned} \sum_{i\in S_{jk}(u)} a_i x_i &= \sum_{i\in S_{jk}(u)} a_i \times \bar{x}_{jk} && (22)\\ \sum_{i\in S_{jk}(u)} a_i y_i &= \sum_{i\in S_{jk}(u)} a_i \times \bar{y}_{jk} && (23) \end{aligned}$$

Fig. 6. LP (Linear Programming) formulation of our thermal-aware 3D microarchitectural fborplanning. This LP is used to perform fborplanning at iteration u of the main algorithm shown in Figure 5.

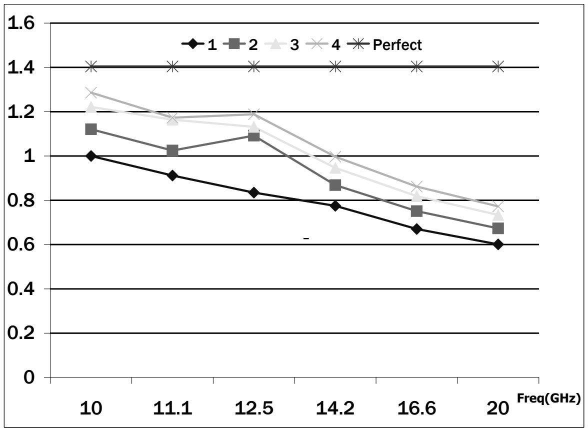

Fig. 7. Performance versus Frequency Scaling for different numbers of layers

area only with the following objective: $\sum_{(i,j)\in E}(X_{ij} + Y_{ij})$. We also use area and performance-only algorithm that uses the following objective:$\sum_{(i,j)\in E} \lambda_{ij} z_{ij}$.

VI. Experimental Results

We performed experiments on ten SPEC2000 benchmarks, six from the integer suite and four from the floating point, similar to [3]. The training input set was used for profile collection while the IPC performance results were gathered using the reference input set. Each simulation for the reference run was fast-forwarded by 200 million instructions and then simulated for 100 million instructions. Each simulation for training was fast-forwarded by 100 million instructions and then simulated for 100 million instructions. The main objective of moving to 3D ICs is to minimize wirelength such that the performance can be improved. Hence we select the largest configuration from [3] for our study. In addition, in [3], the authors also suggest that large/complex processores are more likely to benefit from profile driven floorplanning as used in this study.

First we show that, without the thermal constraint, by scaling the clock frequncy the inter-module communication latency is increased resulting in reduction of IPC despite the use of a good floorplanning algorithm. By moving to 3D ICs, in particular by increasing the number of layers in 3D ICs, the impact of the shortened communication path on IPC can be reduced, as shown in Figure 7.

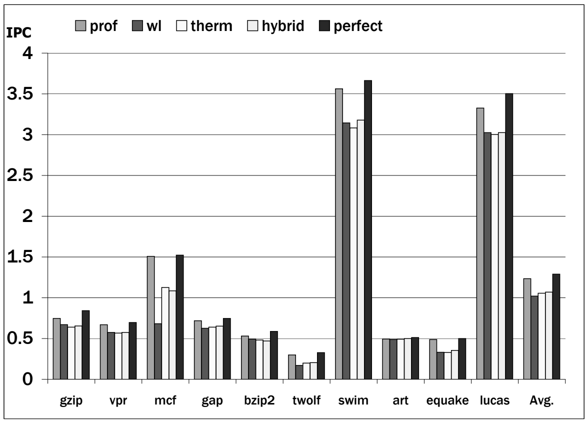

Fig. 8. IPC Performance

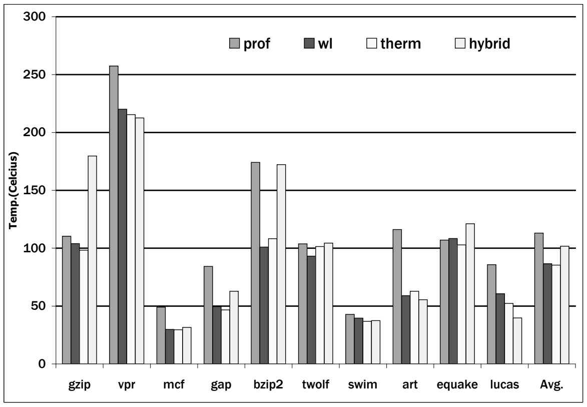

Fig. 9. Maximum Temperature

Next we study the IPC improvement of four layer 3D ICs for profile driven (prof), wirelength driven (wl), thermal driven (therm), hybrid, and perfect floorplanning as shown in Figure 8. In perfect floorplanning, we assume that there is no wire delay and all modules can communicate with each other with latency depending only on gate delay. In other words, we assume that wire delay is zero. This is to show the upper bound of what floorplanning can do. From the figure, it can be seen that profile driven floorplaning can do a good job, very close to perfect floorplanning. The hybrid approach can do a little bit better than thermal and wirelength driven floorplanning.

In terms of thermal reduction, our thermal driven floorplanning can result in 24% reduction comparing with the profile driven approach as shown in Figure 9. Here we report maximum temperature where zero degrees is ambient temperature. Thermal driven floorplanning results in the best result among all four floorplanning objectives. In addition, wirelength driven floorplanning also performs well in terms of maximum temperature. This is because wirelength driven floorplanning tries to balance distance among all modules without favoring any single module. Profile driven, as expected, results in a bad thermal result. This is because performance and temperature are conflicting objectives. Therefore the user has the ability to make balanced tradeoffs with the parameters U_1, U_2, U_3. The hybrid approach also has higher temperature compared with thermal and wirelength driven floorplanning but good performance.

When considering wirelength the profile-driven approach increased

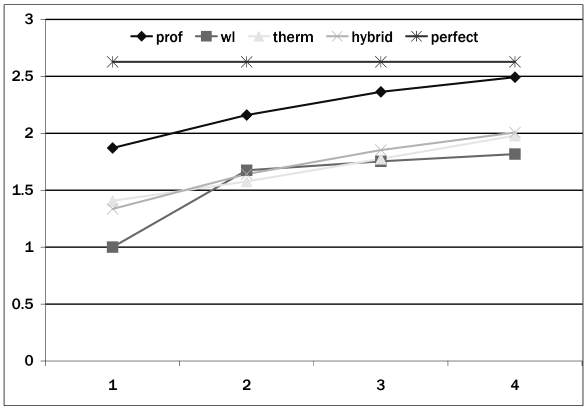

Fig. 10. Impact of number of layers on IPC, normalized values averaged across benchamrks

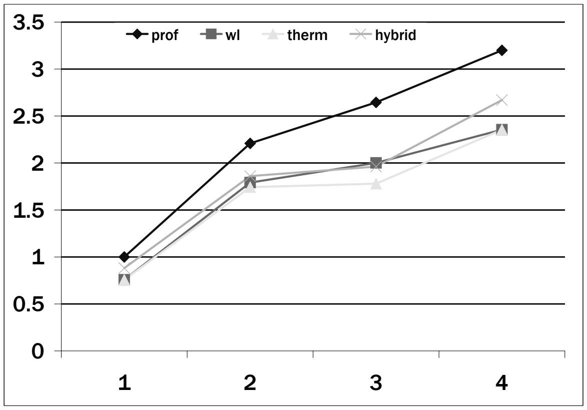

Fig. 11. Impact of number of layers on maximum temperature, normalized values averaged across benchmarks

wirelength by 49% on average, the thermal-driven approad increased wirelength by 35% on average, and the hybrid approach increased wirelength by 25% on average over the wirelength-driven case.

Next we study the impact of the number of layers on IPC as shown in Figure 10. By increasing the number of layers, profile driven floorplanning can reach the performance of perfect floorplanning faster compared with the other approaches. The hybrid approach can perform a little better compared with the thermal and wirelength driven approaches. By increasing the number of layers, all techniques result in the improvement of performance.

Finally, we study the impact on temperature when we increase the number of layers as shown in Figure 11. By increasing number of layers, the maxmimum temperature increases at a high rate. High temperature can result in circuit malfunction. Profile driven floorplanning results in a high temperature increase rate and has to be used with caution. The graph also demonstrates that thermal driven floorplanning results in the slowest rate of temperature increase.

Each of the benchmarks requires approximately 2 hours of CPU time on Pentium Xeon 2.4 GHz dual processor systems. The majority of that time is devoted to simulation.

VII. Conclusions

Here we study the impact of next generation microprocessor design by combining many proposed techniques to reduce wire delay impact. We show that by moving to multi-layer 3D ICs and profile driven floorplanning, it can help increase performance of next generation microprocessor. However by moving to 3D ICs, thermal will become the issue and it can result in circuit breakdown if designers do not aware of it. Here we propose thermal driven floorplanning that can result in 24% maximum temperature reduction comparing with profile driven floorplanning approach. In addition, we also propose hybrid approach that consider thermal and performance issue. In addition, we belive that there are still more room for this hybrid approach improvement and warrant further research.

References

[1] R. Ho, K. W. Mai, and M. A. Horowitz, "The Future of Wires," *Proceedings of the IEEE*, 2001.

[2] W. Liao and L. He, "Full-chip interconnect power estimation and simulation considering concurrent repearter and flip-flop insertion," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2003.

[3] 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 Design Automation Conf.*, 2004.

[4] C. Long, L. Simonson, W. Liao, and L. He, "Floorplanning optimization with trajectory piecewise-linear model for pipelined interconnects," in *Proc. ACM Design Automation Conf.*, 2004.

[5] J. Cong, A. Jagannathan, G. Reinman, and M. Romesis, "Microarchitecture evaluation with physical planning," in *Proc. ACM Design Automation Conf.*, 2003.

[6] W. Gosti and et al., "Wireplanning in Logic Synthesis," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 1998.

[7] R. Zhang, K. Roy, C.-K. Koh, and D. B. Janes, "Stochastic interconnect modeling, power trends, and performance characterization of 3-dimensional circuits," *IEEE Trans. on Electron Devices*, vol. 4, 2001.

[8] 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. on Computer Architecture*, 2000.

[9] P. Cocchini, "Concurrent flip-flop and repeater insertion for high performance integrated circuits," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2002.

[10] K. Sankaralingam, V. A. Singh, S. W. Keckler, and D. Buger, "Routed Inter-ALU Networks for ILP Scalability and Performance," in *Proc. IEEE Int. Conf. on Computer Design*, 2003.

[11] J. Cong, Y. Fan, G. Han, X. Yang, and Z. Zhang, "Architecture and synthesis for on-chip multi-cycle communication," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 550–564, 2004.

[12] M. Casu and L. Macchiarulo, "Floorplanning for throughput," in *Proc. Int. Symp. on Physical Design*, 2004.

[13] C. Tsai and S. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 2000.

[14] C. N. Chu and D. F. Wong, "A matrix synthesis approach to thermal placement," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 1998.

[15] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3D ICs using a force directed approach," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2003.

[16] S. Das, A. Chandrakasan, and R. Reif, "Timing, energy, and thermal performance of three-dimensional integrated circuits," in *Proc. Great Lakes Symposum on VLSI*, 2004.

[17] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3D ICs," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2004.

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

[19] T. M. Austin, "Simplescalar tool suite," http:/www.simplescalar.com.

[20] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *Proceedings of the 27th annual international symposium on Computer architecture*, 2000.

[21] P. Shivakumar and N. P. Jouppi, "CACTI 3.0: An Integrated Cache Timing, Power, and Area Model," HP Western Research Labs, Tech. Rep. 2001.2, 2001.

[22] 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 *Int'l ASIC Conference*, 1996.

[23] G. Chen and S. Sapatnekar, "Partition-driven standard cell thermal placement," in *Proc. Int. Symp. on Physical Design*, 2003.

[24] C. Fiduccia and R. Mattheyses, “A linear time heuristic for improving network partitions,” in *Proc. ACM Design Automation Conf.*, 1982, pp. 175–181.

[25] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich, “GORDIAN: VLSI placement by quadratic programming and slicing optimization,” *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 3, pp. 356–365, 1991.