A Thermal-Driven Floorplanning Algorithm for 3D ICs

Jason Cong, Jie Wei, and Yan Zhang
Computer Science Department, UCLA
Los Angeles, CA 90095
cong,jwei,zhangyan@cs.ucla.edu

ABSTRACT

As the technology progresses, interconnect delays have become bottlenecks of chip performance. Three dimensional (3D) integrated circuits are proposed as one way to address this problem. However, thermal problem is a critical challenge for 3D IC circuit design. In this paper, we propose a thermal-driven 3D floorplanning algorithm. Our contributions include, (1) a new 3D floorplan representation, CBA and new interlayer local operations to more efficiently exploit the solution space; (2) an efficient thermal-driven 3D floorplanning algorithm with an integrated compact resistive network thermal model (CBA-T); (3) two fast thermal-driven 3D floorplanning algorithms using two different thermal models with different runtime and quality (CBA-T-Fast and CBA-T-Hybrid). Our experiments show that the proposed 3D floorplan algorithm with CBA representation can reduce the wirelength by 29% compared with a recent published result from [19]. In addition, compared to a non-thermal-driven 3D floorplanning algorithm, the thermal-driven 3D floorplanning algorithm can reduce the maximum on-chip temperature by 56%.

1. INTRODUCTION

As device sizes continue to shrink and integration densities continue to increase, interconnect delays have become critical bottlen[1]ecks of chip performance. Three-dimensional (3D) IC technologies, under which multiple device layers are stacked together, as schematically shown in Figure 1, can offer the potential to significantly improve circuit performance by reducing interconnect delays. It also offers a flexible way to carry out the system-on-a-chip (SoC) design concept by integrating disparate technologies, such as memory and logic circuits, radio frequency (RF) and mixed signal components, optoelectronic devices, etc., into one single block.

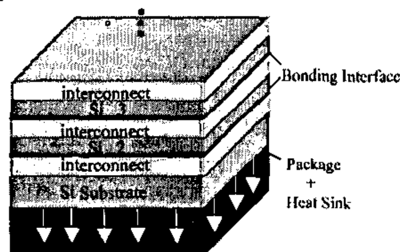


Figure 1 3D IC Technology

[1] This research is partially supported by DARPA under Prime Contract DAAH01-03-C-R193 and CFD Research Corp under Subcontract 03-102

There are mainly three kinds of fabrication technologies to realize 3D IC, including, (1) package level 3D integration, also called 3D MCM [22][27], under which fabricated chips are packaged into one 3D multi-chip module, (2) block level 3D integration, usually by wafer bonding [2], and (3) cell level 3D integration [15]. During 3D placement and floorplanning process, the basic movable objects in designs under technology (1) and (2) are ICs and macro blocks, with limited number and varied sizes and shapes. For designs under technology (3), the basic movable objects are standard cells, and are of very large numbers, usually over 500k. In this paper, we are going to focus on the floorplanning problem for designs under technology (1) and (2).

The most critical challenge of 3D IC design is heat dissipation, which has already been realized and studied even for 2D IC designs [1]. The thermal problem is exacerbated in the 3D cases for mainly two reasons, the vertically stacked multiple layers of active devices cause a rapid increase of power density [6][14] and the thermal conductivity of the dielectric layers inserted between device layers for insulation is very low compared to silicon and metal. For instance, the thermal conductivity for epoxy, k_{epoxy} is 0.05W/mK, while $k_{silicon}$ is 150 W/mK and k_{copper} is 285 W/mK. Thermal issue needs to be considered during every stage of 3D IC design, including the floorplanning process. Thermal constraint poses a new challenge to an efficient 3D floorplanning algorithm.

Floorplanning is a well-studied problem for 2D circuit design. Floorplans can be classified into two categories in terms of block configuration, slicing structure and non-slicing structure. Non-slicing structure is more general but requires more complicated representation data structures. Floorplanning algorithms are normally based on simulated annealing (SA) optimization scheme, using various kinds of floorplan representations. Many floorplanning representations for 2D non-slicing general floorplan have been proposed including sequence pair [18], BSG [16], B*-Tree[4], O-Tree[11], CBL[12], TCG [17], etc. TCG representation is effective in wirelength and area optimization since the relative distances of the modules are implicitly stored in addition to the relative positions between modules.

The existing works on 3D floorplanning can be divided into two groups according to the 3D block representation. One kind of approach is to extend the existing representation to a "true 3D" representation (true 3D), such as 3D TCG [3] and sequence triple [25], with a similar representation structure, a transitive graph and a sequence, respectively, added for the third dimension position representation. However, the chips and the blocks under the current 3D technology are located in a limited number of device layers. So the "true 3D" representation is not suitable for 3D IC floorplanning for two reasons; for designs with fixed device layer

0-7803-8702-3/04/$20.00 ©2004 IEEE.

number, additional constraints will be needed for "true 3D" representation to generate feasible results; also, for the same reason, the additional complicated data structure for the z-axis generates too much redundancy, which is not efficient in both time and space. Another kind of 3D floorplanning approach is to keep an array of 2D representations (2D Array), each representing all blocks located in one device layer, such as BSG [9] and sequence pair [19]. The solution perturbations in these algorithms are limited by the 2D array-based representation. No or very few effective z-axis moves can be enumerated, which deteriorate the solution quality.

For an efficient thermal-driven floorplanning algorithm, a fast thermal model is very important since millions of configurations will be generated for evaluation during the SA-based floorplanning process. The current thermal models can be divided into three groups, *(i)* numerical computing methods such as finite element method (FEM) [8] and finite difference methods (FDM) [21] [22], *(ii)* compact resistive network [20] and *(iii)* simplified closed-form formula [6][14]. Among them, FEM-based and FDM-based method is the most accurate and time-consuming, closed-form formula is the fastest but inaccurate.

There are several existing works on thermal-driven 2D floorplanning. Chu and Wong proposed to use a matrix synthesis approach to do thermal placement in [7], where they tried to even out the heat dissipation of the circuit. Tsai and Kang proposed a compact finite difference method (FDM) based temperature model, based on which two SA-based thermal-driven placement tools for standard cell placement and macro cell placement are developed [21]. Based on the compact model in [8], Chen and Sapatnekar presented a partition-driven standard cell thermal placement method in [5]. Later on, Goplen and Sapatnekar proposed a thermal-driven force-directed method in [10] formulating temperature as another force during force-directed placement. However, as far as we know, there is no existing work on thermal-driven floorplanning for 3D ICs.

In this paper, we propose a thermal-driven 3D floorplanning algorithm. An efficient 3D floorplan representation, combined-bucket-and-2D-array (CBA) is proposed with new solution perturbations added for 3D floorplanning. A compact resistive thermal model is integrated with the 3D floorplanning algorithm to achieve temperature optimization. Also, a thermal-driven 3D floorplanning algorithm based on a simplified closed-form thermal model is proposed for faster solution generation. Finally, a hybrid thermal model based on the two models mentioned above is proposed to get a good tradeoff between runtime and quality. The rest of the paper is organized as follows. In Section 2, we formulate the 3D floorplanning problem with a temperature constraint. In Section 3, we propose a new 3D floorplanning representation, CBA with novel z-axis related operations. In section 4, we proposed three thermal-driven floorplanning algorithm based on three thermal models of different accuracy while sharing the same underlying SA optimization scheme. We present the experiment results in Section 5 and conclude the paper in Section 6.

2. PROBLEM FORMULATION

Let $B = \{b_1, b_2, \dots, b_m\}$ be a set of rectangular modules, with block b_i of width W_i, height H_i, area A_i and a fixed power density P_i, $1 \le i \le m$. Each module is free to rotate. Assume a fixed number of device layer k, such as 4. Let (x_i, y_i, l_i) denote the coordinate of the left lower corner of the rectangle b_i, where $1 \le l_i \le k$, $l_i \in N$. A 3D floorplan F is an assignment of (x_i, y_i, l_i) for each b_i such that no two modules overlap. The goal of a 3D thermal-driven floorplanning algorithm is to minimize *(i)* chip area, *(ii)* wirelength and *(3) interlayer via* number, under the constraint of a maximum on-chip temperature T_0. Chip area is calculated as the maximum of all device layers. *Interlayer via* is a via that goes through a device layer for interlayer connection [2], which has expensive fabrication costs. Also, under the current technology, interlayer vias are of much larger sizes (pitch around $25\mu m \times 25\mu m$ [13]) than the regular vias between metal layers (pitch smaller than $0.5\mu m \times 0.5\mu m$ [28]).

As shown in Figure 1, a heat sink is usually attached to the substrate. We assume the bottom side of the tile stack is isothermal of constant room temperature, 27°C. The four side walls and top of the chip are treated as ambient, since the chip is usually packaged in thermally insulated materials.

3. 3D FLOORPLANNING ALGORITHM

A 2D array representation has the limitation of lacking the relative position information of blocks at different device layers. Although some interlayer operations like swapping two blocks at different layers, or moving a block from one layer to another can be performed, local interlayer operations that change the layout within a small range can not happen because no such information is kept in the 2D array representation. However, it is difficult to adopt a "true 3D" floorplan representation since it contains more z-direction information than necessary.

For 3D floorplanning algorithm, the physical constraint in z-axis is different from x-axis and y-axis, no packing is necessary in z-direction, and the position relationship can be derived directly form the layer number of each block. Based on such feature of 3D IC, we propose a new 3D floorplan representation, combined-bucket-and-2D-Array (CBA) for the current 3D IC technology, which can overcome the limitation of a 2D array representation without the inefficiency of a "true 3D" representation.

3.1 Combined Bucket and 2D Array (CBA)

The CBA representation is composed of two parts, a 2D floorplan representation is used to represent each layer, and a bucket structure to store the relationship between blocks at different device layers. We choose TCG [17] to represent the 2D floorplan on each layer, but in fact any 2D floorplan representation can be used. In order to encode the z-axis neighboring information, a bucket structure is posed on the circuit stack. In each bucket i, indexes of the blocks that intersect with the bucket are stored, the index set is referred to as *IB(i)*, no matter which layer the block is located. In the meantime, each block j stores indexes to all buckets that overlap with the block, the index set is referred to as *IBT(j)*. Figure 2 illustrates an example with 7 blocks, a, b... g and two device layers, L1 and L2, each represented by a separate TCG. A 2×2 bucket structure is imposed to the two layers with indexes listed in the buckets.

Every move in TCG will result in a packing procedure. One single move may change the positions of many blocks. However, updating the bucket every move is inefficient. We defined two kinds of bucket updating scheme, *local update* and *reset*. *Local update* only modifies the blocks in operation and further effects on other blocks are ignored. *Reset* is to scan all the blocks and calculate the indexes *IB(i)* and *IBT(j)* for every bucket i and block j. After every move, a *local update* is called to modify the corresponding bucket structure. *Reset* operation is applied to the

bucket structure only once a while, such as once every 20 moves. The reasons that we don't need to *reset* the bucket structure every time are (1) the bucket structure only provides a reference of where the close neighbors might locate and no packing operation will be based on the bucket indexes, so 100% accuracy is not necessary and (2) from our experiments, most TCG moves only have little effects on the indexes of other blocks. The time complexity of *reset* is $O(n)$, where n is the total block number. The average runtime for a *local update* is $O(n/B)$, where n is the total block number and B is the bucket number.

3.2 Solution Perturbation

There are seven kinds of operations on CBA, as shown in the following,

(1) *rotation*, which rotates a block;
(2) *swap*, which swaps two blocks in one layer;
(3) *reverse*, which exchanges the relative position of two blocks in one layer;
(4) *move*, which moves a block from one side (such as top) of a block to another side (such as left);
(5) *interlayer swap*, which swaps two blocks at different layers;
(6) *z-neighbor swap*, which swaps two blocks at different layers and are close to each other;
(7) *z-neighbor move*, which moves a block to a position at anther layer and are close to the current position;

Rotation, *swap*, *reverse* and *move* are the original moves defined in TCG. Since the changes only happen within one device layer, we also call them *intra-layer moves*. Please refer to [17] for more detailed information about these operations.

Interlayer swap operation randomly chooses two blocks at different layers and swaps them. The two blocks got swapped may be distant to each other, which will probably cause large changes to the circuit layout and the cost function used for optimization.

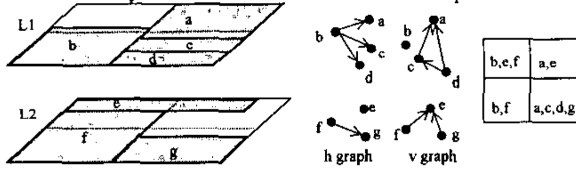


Figure 2 CBA of a 2-layer 7-block floorplan

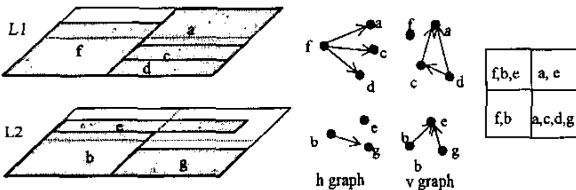


Figure 3 After a z-neighbor swap operation on b and f

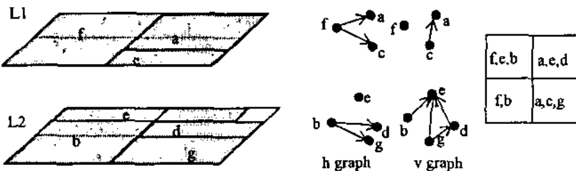


Figure 4 After a z-neighbor move operation on d

A well-known observation of the simulated annealing scheme is that better quality results will be generated when "local changes" are made during low temperatures to achieve higher acceptance rate. TCG provides two local moves for blocks in one layer, reverse and move. We propose two kinds of local operations for blocks at different layers in CBA, *z-neighbor swap* and *z-neighbor move*, which are based on the concept of *z-axis neighbor*.

The definition of a block's *z-axis neighbor* is based on the bucket structure. In order to find the z-axis neighbor of a given block j, $zn(j)$, we first find the set of blocks $B(j)$ that are close to block j.

$$B(j) = \bigcup_{i \in IBT(j)} IB(i)$$

$zn(j)$ is defined as the block k in $B(j)$ with the minimum neighboring cost $zc(j,k)=\alpha\cdot|Ak - Aj| + \beta\cdot dist(k,j)$, where $dist(k,j) = |x_k - x_i| + |y_k - y_i|$. If $zc(j,k)$ is small, then block j and k would probably have similar area and be close to each other, or even overlap along z-axis. Operations based *z-axis neighbors* will only cause small change to the floorplan and the cost function.

- *z-neighbor swap*

z-neighbor swap operation is to swap a block with its *z-axis neighbor*. Figure 3 shows the result of performing a *z-neighbor swap* on block b and block f in Figure 2.

- *z-neighbor move*

z-neighbor move operation is also based on the concept of *z-axis neighbor*. Given one block B_i and its *z-axis neighbor* B_j. B_i is deleted from its own layer l_i and inserted into the layer l_j where B_j locates. B_i is positioned adjacent to B_j at either top or right side. Then B_i and B_j are treated as one new node $B_{j'}$, with coordinate $(xj', yj', lj') = (x_j, y_j, l_j)$, $H_{j'}$ $(W_{j'})$ being H_i+H_j (W_i+W_j). After packing the graph, we can know the relations between B_i and other blocks in the layer l_j by simply copying the relations between $B_{j'}$ and other blocks. Figure 4 shows the result of performing a *z-neighbor move* operation on block d in Figure 3, from layer *L1* to layer *L2*.

3.3 Simulated Annealing Engine

A simulated annealing engine is used to perturb from one CBA representation to another CBA representation. Every time a block configuration is generated, a weighed cost of the optimization objectives and the constraints will be evaluated. The cost function can be written as

$$cost = \alpha \cdot nwl + \beta \cdot narea + \gamma \cdot nvc + \eta \cdot c_T \text{, (1)}$$

where *nwl*, *narea*, *nvc* are the normalized wirelength, chip area and interlayer via number. C_T is the cost of a certain temperature T. C_T is a 3-tie function of maximum on-chip temperature T, as shown in Figure 5, to achieve the temperature constraint.

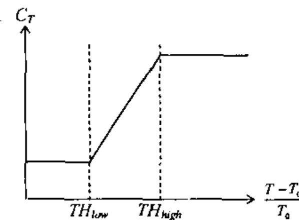


Figure 5 Cost function of Temperature

The SA engine will generate more global operations, including *rotation*, *swap* and *interlayer swap*. With the temperature cooling down, move local operations, including the intra-layer local operations, including *move*, *reverse* and interlayer local

operations, *z-neighbor swap* and *z-neighbor move*, so that the perturbations are more likely to be accepted.

4. THERMAL-DRIVEN FLOORPLANNING

In order to calculate the temperature distribution of the circuit, a bin structure is imposed to the circuit stack, as shown in Figure 6(a). The circuit stack is divided into an array of tile stacks. A node is generated to represent one tile in each device layer. These nodes are the "checking points", whose temperatures are calculated during thermal estimation. We assume each block has a uniform power density at everywhere, which is reasonable assumption since the device layer material, silicon, is a very good thermal conductor. The total power dissipation value at each tile is the sum of the power dissipation from the overlapping areas of all intersecting blocks. Apparently, the smaller the bins are the more accurate a thermal model would be. However, according to our assumption, one block has the same power density everywhere, it is not necessary to have tiles orders of magnitude smaller than the minimum block. In our implementation, a 10×10 bin structure is used.

Based on the bin structure, once a floorplan configuration is generated, the power dissipation at each tile can be calculated and the maximum temperature of that floorplan is calculated by the thermal model. Equation (1) then is updated as a guide for the SA engine. In this way, we can control the maximum on-chip temperature within a reasonable level. In our thermal driven floorplanning algorithm, we tried two different thermal models, a resistive thermal model (CBA-T) and a fast simplified closed-form equation model (CBA-T-Fast).

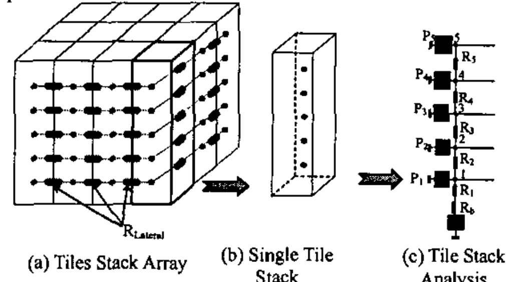
After the floorplan layout is generated, white space is calculated for each block for better routability and thermal control. White spaces are inserted between the blocks by corresponding layout rearrangement.

4.1 Thermal-Driven Floorplanning based on Compact Resistive Network (CBA-T)

For an accurate temperature profiling, we integrated a recently proposed compact resistive thermal model [24] to the thermal-driven 3D floorplanning algorithm. Figure 6(a) shows the resistive model of a 3D circuit consisting of 5 device layers, with the whole chip stack being divided to an array of tile stacks, each of which connected to its neighbor at the same layer by lateral thermal resistors, $R_{Lateral}$. Within each tile stack, a thermal resistor R_i is modeled for each device layer i, as shown in Figure 6(c), and the thermal resistance of the bottom layer material is modeled as R_b. The values of the resistors are determined by accurate FEM-based simulation [26].The power density value at each node i is treated as a current source P_i, and the temperature at each node of the network is then equivalent to the voltage value calculated on that node.

In the linear system built on this resistive network model, the thermal resistance matrix keeps the same for the different linear system introduced by different floorplan layouts. Thus the LR factorization of the thermal resistance matrix can be reused to speed up the calculation. The computing time for a 25×25×4 tile structure is several milliseconds on a 2GHz PC with less than 1% of error [24].

Further speedup can be achieved by reducing the number of thermal profiling during the floorplanning procedure since not every operation will have an impact on the temperature distribution. Large-scale operations such as, swapping two arbitrary blocks will probably cause big changes to the final temperature distribution which requires a thermal profiling to update the temperature cost in equation (1), while local intra-layer operations, such as moving neighbors around, will affect the temperature in a much smaller scale, which can be neglected by skipping the temperature evaluation step. Local interlayer operations, like *z-neighbor swap* and *z-neighbor move*, at the other hand, will make a change to the temperature distribution since interlayer temperature difference is much larger than horizontal temperature differences. Based on the above observations, we can speed up the floorplanning algorithm by calculating the thermal cost only after operations that tend to have large impacts on temperature distribution.



(a) Tiles Stack Array (b) Single Tile Stack (c) Tile Stack Analysis

Figure 6 3D Compact Resistive Network Thermal Model

4.2 Thermal-Driven Floorplanning based on Closed-form Equations (CBA-T-Fast)

Although the resistive thermal model is reasonably fast and speedup is achieved through less often thermal profiling, runtime of a thermal-driven floorplanning algorithm is still much longer than the non-thermal-driven version due to the millions of thermal evaluations during the floorplanning process. For further speedup, we also integrated a closed-form thermal model which separately considers the vertical heat flow and horizontal heat flow and neglects the interplay of the two heat dissipating paths.

4.2.1 Vertical Heat Flow Analysis

If only the vertical heat-dissipating path is considered, such as the single tile stack in Figure 6(c), the temperature increase can be written in an Elmore-delay-like closed-form formula. For example, the temperature at node 4 of the vertical tile stack in Figure 6(c) can be written as,

$$T = \sum_{i=1}^{4}(R_i \sum_{j=i}^{4} P_j) + R_b \sum_{i=1}^{4} P_i \qquad (2)$$

For faster thermal-cost generation, we divide the whole chip stack into an array of such vertical tile stacks. For each tile stack, we calculate the temperature of the top layer node, which will be the hottest node in that tile stack, and the highest temperature value of all tile stack will be put into the cost function (1) for temperature optimization.

During the floorplanning process, the thermal resistances are fixed since there will be no change to the material structure. If we rewrite (2) to (3),

$$T = \sum_{i=1}^{4}(P_i \sum_{j=1}^{i} R_j) + R_b \sum_{i=1}^{4} P_i \qquad (3)$$

then the rationale behind this thermal model can be viewed as penalizing more on the blocks (power density) assigned to higher layers, for example, P_1 is multiplied by R_1 while P_4 is multiplied by $(R_1+R_2+R_3+R_4)$ in (3). Such thermal model will then put the blocks with high power density close to the heat sink, and the blocks with low power density on the top layers. The maximum temperature then can be reduced because the heat generated by the "hot" blocks with high power density can be dissipated to the heat sink easily.

4.2.2 Horizontal Heat Flow Analysis

In addition to minimizing the temperature by assigning high power density blocks to lower device layers, the relative positions of blocks within the same device layer also need to be considered for further temperature reduction. Intuitively, we want to evenly spread out the "hot" blocks, avoiding putting several "hot" blocks close to each other and generating a "hot" area. Otherwise, the heat generated by a "hot" block with high power density cannot be dissipated easily since it is surrounded by other "hot" blocks. In order to consider horizontal heat flow, we try to minimize the maximum temperature difference at the same layer. Given all tiles $T(i,j)$ $i=1,\ldots,m$, $j=1,\ldots,n$, at layer k, where i and j are the indexes of the tiles at *x-axis* and *y-axis*, suppose the temperature at each tile $p(i,j,k)$ can be calculated by equation (2), the maximum temperature difference at layer k,

$$\Delta T(k) = \max_{i,j}(p(i,j,k)) - \min_{i,j}(p(i,j,k))\cdot$$

The normalized $\Delta T(k)$ is used as a weight of the maximum on-chip T to evaluate the cost function (1). After minimizing the maximum horizontal temperature difference, the layout has a more evenly distributed temperature at each layer, which can further bring down the maximum temperature

4.3 Thermal-Driven Floorplanning based on Hybrid Model (CBA-T-Hybrid)

The two thermal models mentioned above both have their advantages and disadvantages. For the resistive network model, it is very accurate but it requires very long runtime, while for the closed-form model, it is very fast but less accurate. In order to get a good tradeoff between these two models, we can use a hybrid model to measure the thermal effect in the thermal-driven floorplanning algorithm. In the implementation of this hybrid model, we will mainly use the closed-form thermal model during the simulated annealing process, but we will use the accurate resistive network model for 20 times every time the temperature of the simulated annealing process drops down. Experimental results prove that we can get a good tradeoff between runtime and quality by using this hybrid model.

4.4 White Space Insertion

For 3D IC technology, where interlayer connections need to go through device layers, white spaces need to be reserved between the blocks for routing. Interlayer vias are much larger than regular vias, and we need to guarantee that enough white space is allocated for them. There are two factors that need to be considered during white space insertion, routability and temperature reduction. In order to consider routability, the degree of the net connections of each block is calculated for white space budgeting. Furthermore, for thermal-critical designs, dummy vias can be inserted into the circuit. Dummy vias are vias inserted specifically for heat dissipation, and are not part of the circuit logic. Under such cases, more white space needs to be reserved around the "hot" blocks for dummy via insertion. So the estimated temperature of each block is also used for white space budgeting.

After the amount of required white space is calculated for each block, layout rearrangement needs to be carried on for a feasible non-overlapping floorplanning result. The rearranging process is guided by the geometric relation information stored in TCG. The enlarged blocks can be repacked using the packing algorithm. Thus we can get a new layout without any overlaps while every block has the planned white space around it.

5. EXPERIMENTAL RESULTS

We implemented the proposed thermal-driven 3D floorplanning algorithm in C++ in Unix with the TCG part based on the TCG source code [17]. The thermal model and its solver are provided by CFD Research Corporation (CFDRC). The experiments are run on a Sun Blade 1000 2x750MHZ machine with Sun OS 5.8. We tested our algorithm on MCNC benchmarks and GSRC benchmarks. Four device layers are assumed for all circuits. All listed results are the floorplanning results before white space insertion.

5.1 Impact of Wirelength/Area Optimization

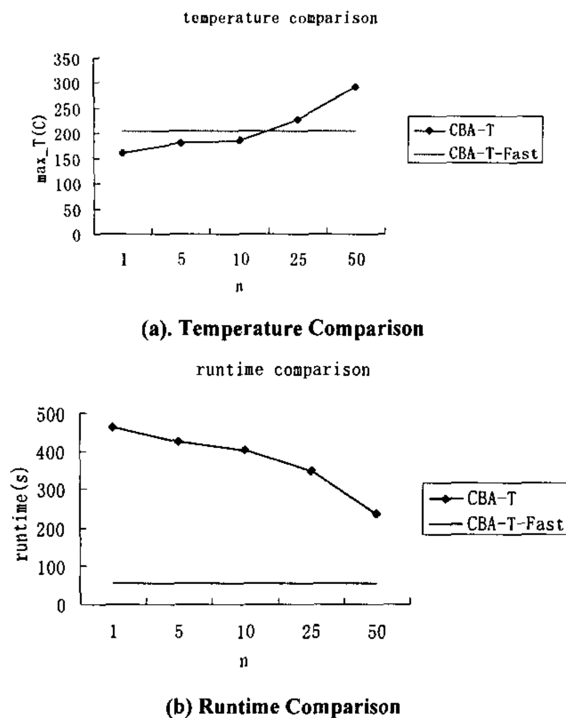
Table 1 shows the experiment results of the proposed floorplanning representation, CBA. We compared the CBA results with a recently published result in [19]. It shows that our algorithm can improve wirelength by 29% and area by 7%. We also compared CBA with a TCG-based 2D Array representation without the proposed local interlayer operations. We can see that CBA can improve the wirelength by 3% and area by 5%. For GSRC benchmarks, CBA can improve the wirelength by 5.5%. The local interlayer operation does not improve the wirelength for MCNC benchmarks because the block number is too small to show the effect of *z-axis neighbor* operations.

5.2 Impact of Thermal Optimization

For thermal-driven floorplanning, we assign a power density of random number between 10^5 (W/m^2) and 10^7 (W/m^2) to each block [21]. Table 2 shows the results of the thermal-driven floorplanning based on a compact resistive thermal model, CBA-T. The maximum on-chip temperature is reduced by 56% in average with a 21% area increase, a comparable wirelength and interlayer via number. The 9× runtime is mainly due to thermal profiling. Table 3 shows the results of the thermal-driven floorplanning based on a closed-form formula, CBA-T-Fast and also the thermal-driven floorplanning based on the hybrid model, CBA-T-Hybrid. For CBA-T-Fast, the maximum on-chip temperature can be reduced by 40% with $2\times$ run time compared with the non-thermal-driven 3D floorplanning algorithm. For CBA-T-Hybrid, the maximum on-chip temperature can be reduced by 50% with $3\times$ run time compared with CBA. In Table 2, Table 3 and Figure 7, all temperatures are in Celsius.

We also tried to control the runtime of CBA-T by reducing the frequency of thermal profiling with a thermal evaluation being performed every n operations or after the large scale swap operations and interlayer operations. Figure 7 shows the results on ami33 of different n. Figure 7(a) showing the final temperature of different n, with the horizontal line being the result of CBA-T-Fast, while Figure 7(b) shows the runtime, with the horizontal line

being the result of CBA-T-Fast. It shows that doing a thermal profiling every 5-10 iterations can still generate reasonably good results. However, even for CBA-T with n=50, the runtime is still 3× that of CBA-T-Fast.



(a). Temperature Comparison

(b) Runtime Comparison

Figure 7 Effect of Thermal Analysis Frequency

5.3 Accurate Thermal Simulation

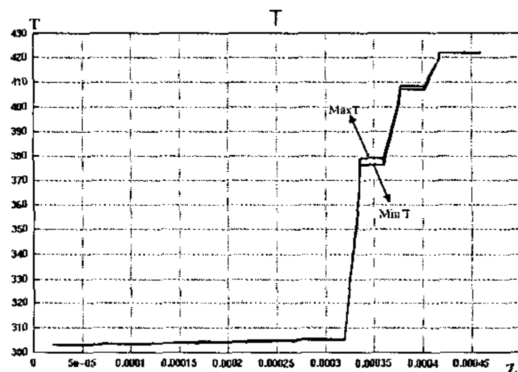


Figure 8 Temperature along Z-Axis

We also used a FEM-based thermal simulation tool [26] to verify our floorplanning results. The temperatures displayed from Figure 8 to Figure 9 are thermodynamic temperature instead of Celsius temperature. This thermodynamic temperature minus 273 is the normal Celsius temperature. Figure 9 shows the simulation results of ami33 before and after thermal optimization. The temperature value for ami33 without thermal-optimization is not exactly the same with the one in Table 1. The reason is that the non-thermal-driven floorplanning algorithm has no control over temperature, so the resulting temperature, to some extent, is random. In Figure 9, the maximum on-chip temperature for benchmark ami33 is 587.2 degree without thermal optimization. While the maximum on-chip temperature after thermal optimization is 425.7 degree.

Figure 8 shows two curves of the maximum and minimum temperatures at the same z coordinate. They reflect the temperature distribution along z-axis. We can see that the temperature distribution is quite uneven between different layers. The layer that is far from the bottom heat sink has much higher average temperature than the layer closer to the heat sink. However, the temperature distribution within one layer is more even. The reason is that silicon, which forms the device layer, is a good thermal conductor while the dielectric material and the epoxy material between different device layers are poor thermal conductors. That's also the reason why the horizontal heat flow analysis has less impact on minimizing maximum on-chip temperature than the horizontal heat flow analysis.

6. CONCLUSIONS AND FUTURE WORK

By adding local interlayer operations, our proposed 3D representation, CBA can exploit the 3D solution space more efficiently than the pervious works. In the meantime, we have effectively controlled the maximum on-chip temperature of 3D chip to a reasonable level through the thermal-driven floorplanning algorithm. Three thermal models, an accurate resistive thermal model, a fast closed-form temperature equation, and a hybrid model are integrated to the thermal driven floorplanning algorithm. Also, thermal-driven white space insertion can guarantee the following routing procedure to have enough space to plan interlayer vias for interlayer connections and heat dissipation.

There are several interesting topics in the field of thermal-driven 3D floorplanning. In the current approach, we don't consider the impact of large interlayer vias on the thermal resistance for each tile stack since we don't know the exact locations of those large interlayer vias before routing process. However, since those large interlayer vias are very good thermal conductors, we might need to consider their impact on the thermal resistance in the thermal driven white space stage and routing stage later. We are also interested in investigating new perturbations to further exploit the 3D solution space. In order to speed up the current thermal-driven 3D floorplanning algorithm, we also want to apply the closed-form temperature formula to faster floorplanning algorithms without going through simulated annealing.

7. ACKNOWLEDGEMENTS

The authors wish to thank Dr. Marek Turowski and Mr. Patrick Wilkerson from CFD Research Corporation (CFDRC) for the compact resistive network thermal model.

REFERENCES

[1] K. Banerjee, "Thermal effects in deep submicron VLSI interconnects," *IEEE Int. Symp. Quality Electronic Design*, 2000

[2] K. Banerjee, S. J. Souri, P. Kapur, K. C. Saraswat, "3D ICs: A novel chip design for improving deep submicron interconnect performance and systems-on-chip integration,"

Proc. IEEE, Special Issue on Interconnects, May 2001, pp.602-633.

[3] K. Bazargan, R. Kastner, and M. Sarrafzadeh " 3D Floorplanning: Simulated Annealing and Greedy Placement Methods for Reconfigurable Computing Systems", *Journal of Design Automation for Embedded Systems,* 2000

[4] Y.-C. Chang, Y.-W. Chang, G.-M. Wu, and S. –W. Wu, "B*-trees: A new representation for non-slicing floorplans," *Proc. DAC,* pp.458-463, June 2000.

[5] G. Chen and S. S. Sapatnekar, "Partition-Driven Standard Cell Thermal Placement," in *Proc. of the ACM Int. Symp. On Physical Design,* pp. 75-80, 2003.

[6] T.Y. Chiang, S.J. Souri, C.O. Chui, and K.C. Saraswat, "Thermal Analysis of Heterogeneous 3D ICs with Various Integration Scenarios", *Technical Dig. IEDM,* 2001, pp.681-684.

[7] C. N. Chu and D. F. Wong, "A Matrix Synthesis Approach to Thermal Placement," in *Proc. Int. Symp. On Physical Design,* pp. 163-168, 1997.

[8] W. K. Chu and W. H. Kao ,"A three-dimensional transient electrothermal simulation system for IC's", *Proc 1st THERMINIC Workshop,* pp201-207 Sept. 1995

[9] Y. Deng, W. Maly: "Interconnect characteristics of 2.5-D system integration scheme". *Proc. Int. Symp. On Physical Design*: 171-175

[10] B. Goplen and S. S. Sapatnekar, "Efficient Thermal Placement of Standard Cells in 3D ICs using a Force Directed Approach," *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design,* pp. 86 - 89, 2003.

[11] P.-N. Guo, C.-K. Cheng, and T. Yoshimura, "An O-Tree representation of non-slicing floorplan and its applications," *Proc. DAC,* pp.268-273, June 1999.

[12] X. Hong, G. Huang, Y. Cai, J. Gu, S. Dong, C.-K. Cheng, and J. Gu, "Corner Block List: An effective and efficient topological representation of non-slicing floorplan," *Proc ICCAD,* pp. 8-12. 2000.

[13] S. B. Horn "Vertically integrated sensor arrays VISA" (Invited), *Defense and Security Symposium 2004*

[14] S. Im and K. Banerjee, "Full Chip Thermal Analysis of Planar (2D) and Vertically Integrated (3D) High Performance ICs", *Tech. Digest IEDM,* 2000, pp.727-730.

[15] T. H. Lee, " A vertical leap for microchips", *Scientific American,* 2002

[16] K. W. Lee, T. Nakamura, T. Ono, Y. Yamada, T. S. Nakatake, H. Murata, K. Fujiyoshi, and Y. Kajitani, "Module placement on BSG-structure and IC layout applications," Proceedings of ACM/IEEE ICCAD, Nov.1996, pp. 484-491.

[17] J.-M. Lin and Y.-W. Chang, "TCG: A transitive closure graph based representation for non-slicing floorplans," in *Proc. of ACM/IEEE Design Automation Conference (DAC-2001),* pp. 764--769, Las Vegas, NV, June 2001

[18] H. Murata, E. S. Kuh, "Sequence Pair Based Placement Method for Hard/Soft/Pre-placed Modules," *Proc. ISPD,* pp. 167–172, 1998.

[19] P. H. Shiu, R. Ravichandran, S. Easwar, and S. K. Lim, "Multi-layer Floorplanning for Reliable System-on-Package", accepted by *IEEE International Conference on Circuits and Systems,* May, 2004

[20] M. Stan, K. Skadron, M. Barcella, W. Huang, K. Sankaranarayanan, and S. Velusamy. "HotSpot: a Dynamic Compact Thermal Model at the Processor-Architecture Level." *Microelectronics Journal,* Volume 34, Issue 12, Dec. 2003, Pages 1153-1165

[21] C. H. Tsai and S. M. Kang, "Cell-Level Placement for Improving Substrate Thermal Distribution," *IEEE Trans. On Computer.-Aided Des,* vol. 19, no. 2, pp. 253-266, Feb. 2000.

[22] Y. K. Tsui, S. W. R. Lee, J. S. Wu, J. K. Kim and M. M. F. Yuen, "Three-dimensional packaging for multi-chip module withthrough-the-silicon via hole", *Electronics Packaging Technology,* 5th Conference, pp 1-7 2003,

[23] T.-Y. Wang, Y.-M. Lee and C.C.-P Chen," 3D thermal-ADI: an efficient chip-level transient thermal simulator", *Proc. Int. Symp. On Physical Design,* pp 10-17 2003

[24] P. Wilkerson, A. Raman, and M. Turowski "Fast, Automated Thermal Simulation of Three-Dimensional Integrated Circuits", *ITherm 2004,* Las Vegas, Nevada, 1-4 June 2004.

[25] H. Yamazaki, K. Sakanushi, S. Nakatake, Y. Kajitani, "The 3D-packing by Meta Data Structure and Packing heuristics" *IEICE Trans Fundamentals,* pp 639-645 April 2000

[26] CFD-ACE+ Module Manual, Vol. I, Version 2002.

[27] http://www.irvine-sensors.com/r_and_d.html

[28] International Technology Roadmap for Semiconductors 2003

	Results from [19]			TCG-based 2D Array			CBA		
	area	wirelength	vias	area	wirelength	vias	area	wirelength	vias
ami33	—	—	—	3.52E+05	23139	106	3.44E+05	23475	111
ami49	—	—	—	1.49E+07	453083	191	1.27E+07	465053	203
n100	55081	117407	885	53295	97066	704	51736	90143	752
n200	55722	251626	1585	51714	189995	1487	50055	175866	1361
n300	79800	262042	1532	74712	232074	1603	75294	230175	1568
Avg.	1.07	1.29	1.1	1.05	1.03	1.017	1	1	1

Table1 Impact of CBA

	CBA					CBA-T				
	area	wirelength	vias	Temp(C)	runtime	area	wirelength	vias	Temp(C)	runtime
ami33	3.53E+05	22495	93	470.7	23	4.14E+05	24442	98	160	466
ami49	1.49E+07	446815	179	258.7	86	1.84E+07	477646	211	151	521
n100	5.29E+04	100525	955	391.4	313	6.56E+04	92450	1044	158	4322
n200	5.77E+04	210333	2093	322.6	1994	6.91E+04	190886	2021	155.7	6843
n300	8.90E+04	314980	2326	372.8	3480	1.06E+05	253837	2261	166.7	17484
Avg.	1	1	1	1	1	1.21	0.957	1.05	0.44	9.71

Table 2 Impact of CBA-T

	CBA-T					CBA-T-Fast					CBA-T-Hybrid				
	area	wirelength	vias	T(C)	runtime	area	wirelength	vias	T(C)	runtime	area	wirelength	vias	T(C)	runtime
ami33	4.14E+05	24442	98	160	466	3.65E+05	24090	103	204	56	4.25E+05	21853	104	192	170
ami49	1.84E+07	477646	211	151	521	1.85E+07	476339	205	196	144	1.61E+07	550381	233	195	250
n100	6.56E+04	92450	1044	158	4322	6.03E+04	111979	1156	222	446	6.65E+04	115315	1052	173	689
n200	6.91E+04	190886	2021	155.7	6843	5.50E+04	203530	2058	242	4474	6.91E+04	218739	1991	181	3415
n300	1.06E+05	253837	2261	166.7	17484	1.05E+05	326630	2350	208	4953	9.46E+04	283377	2314	202	8649
Avg.	1.21	0.957	1.05	0.437	9.71	1.1	1.06	1.09	0.6	1.82	1.16	1.06	1.08	0.5	3.21

Table 3 Impact of CBA-T-Fast and CBA-T-Hybrid

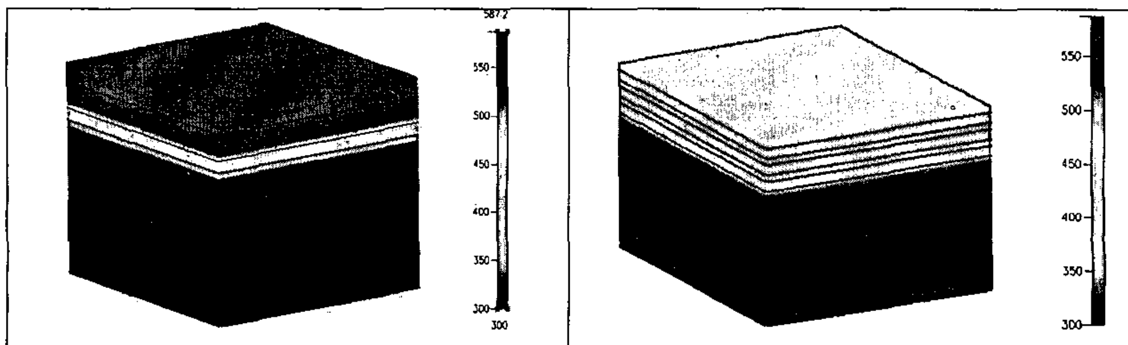


Figure 9 Thermal Simulation for the Floorplan Result w/o Thermal Optimization