Congestion-Driven Global Placement for Three Dimensional VLSI Circuits

Vidit Nanda, Karthik Balakrishnan, Mongkol Ekpanyapong, and Sung Kyu Lim

School of Electrical and Computer Engineering,
Georgia Institute of Technology
{gte272u,gte245v,pop,limsk}@ece.gatech.edu

ABSTRACT

The recent popularity of 3D IC technology stems from its enhanced performance capabilities and reduced wiring length. However, the problem of thermal dissipation is magnified due to the nature of these layered technologies. In this paper, we develop techniques to reduce both the local and global congestions of 3D circuit designs in order to alleviate thermal issues. Our approach consists of two phases. First, we use a multilevel min-cut based approach with a modified gain function in order to minimize the local congestion. Then, we perform simulated annealing to reduce the circuit's global congestion. Experimental results show that our local congestion is reduced by an average of over 44% and global congestion is reduced by over 16%. Moreover, we only see an 11% increase in the wiring length and the number of vias required.

1. INTRODUCTION

With the recent advent of three-dimensional Integrated Circuit technologies, there has been a positive impact on the performance and wiring length of these ICs. Typically, the layered placement of transistors in multiple planes (i.e. 2.5D placement) allows for a more compact chip with inherently better performance than one fabricated with traditional 2D placement techniques. However, the heat generation from a given area increases with increasing power density within that area [13]. Therefore, the problem of local and global wire congestion in these 2.5D chips is paramount. If a chip requires a large amount routing resources within a local area, the resulting thermal dissipation within that area will increase significantly. For example, consider Figure 1. Clearly, the circuit on the left is more problematic than the one on the right in terms of thermal considerations.

Previous work in the area of 2.5D placement has focused on minimizing the intra-layer wirelength and the number of inter-layer connections, or "vias." The results of [11] indicate an improvement in overall wirelength when implementing the 2.5D layered placement framework instead of equivalent traditional 2D placement. Other work has employed stochastic methods to determine the wirelength distributions, trends in power consumption, and performance capabilities of 2.5D-Ics [1,2,3,4,5]. The conclusions derived from stochastic analysis confirm that 2.5D chips will provide better performance with larger compaction, but also predict a non-trivial increase in thermal dissipation due to the current state of heat sink technology.

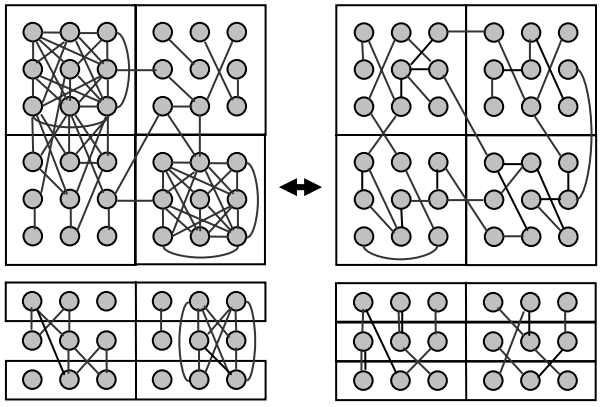

Figure 1. Balanced vs unbalanced local/global wire congestion for 3D circuits. Top-down and side views are shown.

In this paper, we provide a technique to reduce both local and global congestion in a 2.5D chip in order to lessen the inherent thermal costs of the chip. Our approach involves a two-stage refinement procedure: initially, we use a multilevel min-cut based method to minimize the congestion within confined areas of the chip. This is followed by a simulated annealing-based technique which works to minimize the amount of congestion created from global wires. We show that our congestion minimization does not have any significant negative impact on the wirelength or the number of vias.

The rest of this paper is organized as follows. Section 2 provides preliminaries of our approach. Section 3 discusses our min-cut based approach to reduce local wire congestion. Section 4 explains our simulated annealing approach aimed for global wire congestion. Section 5 provides experimental results. Section 6 concludes our paper and describes the ongoing research in this field.

2. PROBLEM FORMULATION

2.1 Physical Planning for 2.5D Layouts

Given a sequential gate-level netlist $NL(C, N)$, where C is the set of cells representing gates, clusters and flip-flops, and N is the set of nets connecting the cells, the purpose of the 2.5D Physical Planning problem is to assign the cells in NL to a given m x n x p (=K) slots while preserving area constraints. The 2.5D Physical Planning problem has a solution P: $C \rightarrow B$, wherein each cell in C is assigned to a unique block $B \in \{B_1(x_1,y_1,z_1), B_2(x_2,y_2,z_2),..., B_K(x_K,y_K,z_K)\}$,

where B denotes the set of blocks, and (x_i, y_i, z_i) represents the geometric locations of B_i, with area constraint $A(L,U)$, for $1 \le i \le K$. The 2.5D-PP solution must satisfy the following conditions:

1. $B_i \subset C$ and $L \le |B_i| \le U$
2. $B_1 \cup B_2 \cup \ldots \cup B_k = C$.
3. $B_i \cap B_j = \varnothing$ for $i \ne j$.

2.2 Congestion Objective

a. Local Congestion

Given a block B from the physical planning solution, we define the local wiring cost $LC(B)$ as:

$$LC(B) = \sum_{n_i \in N \ni n_i \cap B \neq \phi} |n_i| - 1$$

This value corresponds to the minimum number of wires required to construct the tree representation of a net of size $|n_i|$. Then, the local congestion of the entire solution P is given by:

$$LC(P) = \max\{LC(B_i)\} - \min\{LC(B_j)\},\ 1 \le i, j \le K$$

b. Global Congestion

For any two adjacent blocks B_i, B_j from the placement solution above, we denote the common incident hyperedge set by $H_{ij} = \{h \in H: h \cap B_i \neq \varnothing \text{ and } h \cap B_j \neq \varnothing\}$. Then, the global congestion at the boundary $<B_i, B_j>$ is measured as $GC_{ij} = |H_{ij}|$. Then, the global congestion of the placement solution P is given by:

$$GC(P) = \max\{GC_{ij}\} - \min\{GC_{ij}\},$$

for all i, j such that B_i and B_j are adjacent.

2.3 2D Wirelength Objective

We model the netlist NL using a hypergraph $H = (V, E_H)$, where the vertex set V represents cells, and the hyperedge set E_H represents nets in NL. Each hyperedge is a non-empty subset of V. The x-span of hyperedge h, denoted h_x, is defined as $h_x = \max_{c \in h}\{x_i \mid c \in B_i\} - \min_{c \in h}\{x_i \mid c \in B_i\}$. The y-span, denoted h_y, is calculated using the y-coordinates. The sum of x-span and y-span of each hyperedge h is the half-perimeter of the 2D bounding box of h, denoted by h_w. The wirelength $W(P)$ of global placement solution P is the sum of h_w for all hyperedges h in H.

2.4 Via Objective

We define the set of vias as a restriction on the set of edges E_H in H. A via is a z-directional wire that connects components from multiple layers of the circuit. Given a hyperedge h, we say that h contains a via if and only if it contains cells c_1 at (x_1, y_1, z_1) and c_2 at (x_2, y_2, z_2) with $z_1 \ne z_2$. To be precise, the via count of a hyperedge h, denoted h_v, is defined as $h_v = \max_{c \in h}\{z_i \mid c \in B_i\} - \min_{c \in h}\{z_i \mid c \in B_i\}$. Our via objective is to minimize $V(P)$, the sum of h_v for all hyperedges in H.

The overall objective of our congestion-driven 2.5D PP problem is to minimize $LC(P)$ and $GC(P)$ while maintaining an acceptable $V(P)$ and $W(P)$.

3. LOCAL CONGESTION-DRIVEN GLOBAL PLACEMENT (LC-CUT)

The purpose of this algorithm is to balance the amount of local congestion while maintaining wirelength and via results comparable to those of pure mincut-based techniques. The approach involves modifying the gain function of a multi-level cutsize based partitioner to reduce local congestion.

Our cut sequence is an extension of the two cut sequence techniques used in [11]. Their first method performs via-minimizing interlayer cuts (z cuts) before performing intralayer cuts (x, y) to minimize the 2D wirelength. Their second cut sequence does the opposite, making all (x, y) cuts first before performing (z) cuts to achieve minimal wirelength. For the purposes of maintaining a balanced combination of via count and wirelength during our algorithm, we devise a new cut sequence, (*z, x, y, z, x, y, ...)*. We experimentally determined that the best results in terms of balanced wirelength and via count were produced by this new cut sequence.

Instead of focusing only on wirelength and via minimization while making cell moves, **LC-CUT** reduces the overall local congestion as necessary. The necessity of congestion-driven moves is determined by a variable called *thresh*, which controls the nature of gain computations. The input to this algorithm is a netlist $NL(C, N)$, where C represents gates or clusters of gates, and N represents the nets that connect them. The output is a pair of blocks, B_i and B_j, each of which contains a subset of C. Figure 3 shows a pseudocode representation of this algorithm, which is explained in detail below.

3.1 Initialization Phase

The first stage of the **LC-CUT** algorithm involves the initialization of a bucket structure, which stores the weighted gains of all cells in C. In line 1 of Figure 3, all cells in the netlist $NL(C, N)$ are inserted into either B_i or B_j such that the area constraints are satisfied. Each cell will have the opportunity to move from its current block to a neighboring block in the later stages of the algorithm. Line 2 involves the computation of $g_\alpha(c_i)$, the cutsize gain. Additionally, the local congestion gain (defined below) is also computed. For a given cell c, moved from B_i to B_j, the change in congestion for blocks B_i and B_j are given by $c_{\Delta i}$ and $c_{\Delta j}$, respectively. Then,

$c_{\Delta i} = LC(B_i) - LC(B_i \setminus \{c\})$, and $c_{\Delta j} = LC(B_j) - LC(B_j \setminus \{c\})$

Finally, the local congestion gain of moving c from its current block B_i to a neighboring block B_j is given by:

$$g_\beta(c) = \left|LC(B_i) - LC(B_j)\right| - \left|LC(B_i) - LC(B_j) + c_{\Delta i} + c_{\Delta j}\right|$$

In line 3, the local congestion values for the two blocks B_i and B_j are calculated. Line 4 involves the initialization of two important variables that will determine the frequency of congestion-driven moves. The first is *thresh*, for which the maximum cell degree must be a lower bound. We use this lower bound value plus 4% of $LC(B_i) + LC(B_j)$. This is to ensure the accuracy of congestion gain computations. The second is *cong_mode*, a boolean which, if true,

will initiate local congestion-driven cell moves. Otherwise, the moves will be made purely on the basis of g_α, the cutsize gain. In line 5, the difference between the local congestions of the two blocks is stored.

3.2 Cell Movement Phase

The second stage of the **LC-CUT** algorithm continuously moves cells of maximum gain until there is no more positive gain left. In line 7, the cell of maximum gain, c, is extracted from the bucket. Then, lines 8-9 make sure that moving c will not result in an area constraint violation. Line 10 updates B_i and B_j, making the cell move, and updates $g_\alpha(c_i)$ for all cells c_i that neighbor c. Line 11 checks to see if the balance gain computations are necessary, and if so, updates all $g_\beta(c_i)$, for all c_i neighboring c, according to the above equation. These updates for local congestion gain can be done incrementally since the values of *LC* for each block are stored after every move and the calculations for $c_{\Delta i}$ and $c_{\Delta j}$ are trivial.

In line 12 of the **LC-CUT** algorithm, the overall gain is calculated as a weighted cost function of g_α and g_β. If this gain value is less than zero, then the loop is exited and B_i and B_j are returned. Otherwise, the values of *LC* are updated, and Δ_{LCnew} is becomes $LC(B_i) - LC(B_j)$ (line 13). As shown in Figure 2, the magnitude of the difference between $LC(B_i)$ and $LC(B_j)$ determines whether or not congestion-based moves are made. Line 14 updates the value of *cong_mode* accordingly.

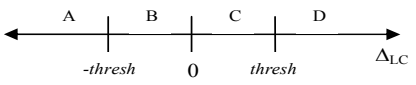

Figure 2. Congestion-driven move control.

During the next portion of the algorithm, the values of $g_\beta(c)$ are updated if necessary. Figure 2 shows the four regions of possible values for Δ_{LC}: A, B, C and D. When Δ_{LC} is in B or C, the congestions in B_i and B_j are relatively equal. However, when Δ_{LC} is in A or D, one block is significantly more locally congested than the other. For this reason, local congestion is considered when Δ_{LC} is in A or D, and it is ignored when Δ_{LC} is in B or C. This serves to improve runtime since congestion gain calculations are unnecessary when Δ_{LC} is in [*-thresh, thresh*].

Certain moves that cause Δ_{LC} to shift from one particular region to another will necessitate a bucket re-initialization. In line 15, the moves A→D and D→A result in the re-computation of $g_\beta(c)$ for every cell c and a bucket reset. This is necessary since the sign of Δ_{LC} has changed, and according to the equation above, $g_\beta(c)$ will change. In line 17, the algorithm checks for a cell move from {B, C} to {A, D}. For this case, local congestion must again be considered, so g_β is computed and the bucket is reset (line 18). Line 19 checks for a transition of Δ_{LC} from {A, D} to {B, C}. In this situation, the local congestion gains for all cells are set to zero and the bucket is reset, and therefore contains only cutsize gain information. Line 21 updates the value of Δ_{LC}, line 22 is the stopping condition for the movement phase, and line 23 returns B_i and B_j.

LC-CUT
Input: NL(C, N)
Output: B_i, B_j
1. Insert cells from C into B_i and B_j ∋ $L < |B_i|, |B_j| < U$
2. Compute $g_\alpha(c_i)$ and $g_\beta(c_i)$ ∀ $c_i \in C$, add to Bkt
3. Initialize $LC(B_i)$ and $LC(B_j)$
4. Initialize *thresh* and *cong_mode*
5. $\Delta_{LC} \leftarrow LC(B_i) - LC(B_j)$;
6. **loop**
7. c ← Bkt.**extract_max;**
8. **if** (Moving c violates area constraint)
9. **goto loop**
10. Update B_i and B_j and $g_\alpha(c)$ ∀ $c_i \in$ *nets*(c)
11. Update $g_\beta(c_i)$ ∀ $c_i \in$ *nets*(c) if *cong_mode* is *true*
12. *gain* ← $w_\alpha \cdot g_\alpha(c) + w_\beta \cdot g_\beta(c)$;
13. update $LC(B_i)$, $LC(B_j)$ and Δ_{LCnew}
14. *cong_mode* ← *true* if $|\Delta_{LCnew}| >$ *thresh*, else *false*
15. **if** $|\Delta_{LC}|, |\Delta_{LCnew}| >$ *thresh* **and** $\Delta_{LC} \cdot \Delta_{LCnew} < 0$
16. compute $g_\beta(c_i)$ ∀ $c_i \in$ C and reset Bkt;
17. **else if** $|\Delta_{LC}| \le$ *thresh* $< |\Delta_{LCnew}|$
18. compute $g_\beta(c_i)$ ∀ $c_i \in$ C and reset Bkt;
19. **else if** $|\Delta_{LCnew}| \le$ *thresh* $< |\Delta_{LC}|$
20. $g_\beta(c_i) \leftarrow 0$ ∀ $c_i \in$ C and reset Bkt;
21. $\Delta_{LC} \leftarrow \Delta_{LCnew}$;
22. **until** *gain* < 0
23. **return** B_i, B_j;
end LC-CUT

Figure 3. LC-CUT, algorithm for local congestion.

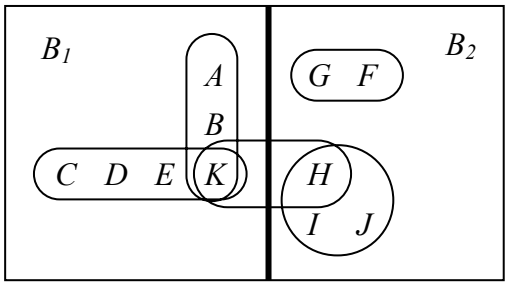

Figure 4. Illustration of balance gain computation

3.3 Example Cell Move

Figure 4 depicts an example of a stage in **LC-CUT** where the highlighted cell, K, is the one to be moved. Currently, $LC(B_i)$ is 5 and $LC(B_j)$ is 3. Therefore, Δ_{LC} is (5 – 3) = +2. Now, we will compute the values of $g_\alpha(K)$ and $g_\beta(K)$.

$g_\alpha(K) = 1 - 1 - 1 = -1$,
$c_{\Delta i} = -2$ and $c_{\Delta j} = +1$, and
$g_\beta(K) = |5 - 3| - |5 - 3 + (-2) + (1)| = +1$.

After moving cell K, $LC(B_i) = 3$ and $LC(B_j) = 4$. Then, Δ_{LCnew} will be equal to (3 – 4) = -1. Additionally, the cell gains for A, B, C, D, E and H must be updated before the next cell move is made.

4. GLOBAL CONGESTION-BASED SOLUTION REFINEMENT

4.1 Overview of the Approach

We use a simulated annealing-based block movement technique for minimization of global wirelength, via count and overall congestion of the placement solution obtained above. Since local pair-wise congestion, 2D wirelength and via count have already been minimized, it is sufficient to swap entire blocks (rather than individual gates, or clusters of gates) and check for improvements over the previous solutions. We use the min-cut results as the initial solution and compute initial temperature T_0. The temperature is then decremented in a standard one-parameter exponential format ($T_{i+1} = \alpha T_i$, $\alpha < 1$). We let $N(T)$ be the number of random block swaps made at every temperature T. The cost C after the I^{th} move made at temperature T is computed using the following linear function of wirelength, via count and global congestion:

$$C[I(T)] = \mathbf{a_1}\cdot\Delta V(I) + \mathbf{a_2}\cdot\Delta W(I) + \mathbf{a_3}\cdot\Delta GC(I).$$

Here,

- $I(T) < N(T)$ represents the I^{th} move made at temperature T,
- $\Delta V(I)$ is the total change in via count after the I^{th} move,
- $\Delta W(I)$ is the total change in wirelength after the I^{th} move,
- $\Delta GC(I)$ is the total change in global congestion after the I^{th} move, and
- $\mathbf{a_1}$, $\mathbf{a_2}$, $\mathbf{a_3}$ are graph-dependent experimentally obtained weight values.

As is standard with all annealing algorithms, improvements are guaranteed only at a significant runtime expense. In order to make the procedure as efficient as possible, it becomes necessary to perform highly optimized incremental evaluation, which is described in detail below.

4.2 Incremental Evaluation:

a) Congestion

Recall the global congestion metric: Given a boundary $<B_i, B_j>$ between neighboring blocks B_i and B_j, we defined the boundary congestion to be the number of hyperedges crossing that boundary, denoted by $|H_{ij}|$. Consider such a hyperedge h in H_{ij}. Let its bounding box lie between $(x_{min}, y_{min}, z_{min})$ and $(x_{max}, y_{max}, z_{max})$, as shown in Figure 5 below.

Then, we assume that h is routed randomly along one of 6 shortest outer edge paths. This model is exact for two-pin nets and fairly accurate for three-pin nets, which comprise a clear majority of all nets in any given benchmark circuit. On average, 83% of all nets in a benchmark circuit are either two-pin nets (69%) or three-pin nets (14%). Note that when two blocks are swapped, boundaries need to be updated only for nets in H_{ij}. This property allows us to conveniently ignore all nets that are not incident upon the two blocks central to the current move, thereby achieving remarkable improvements in overall runtime. Once B_i and B_j have been randomly selected for the I^{th} move, the boundary congestion contribution of all nets in the corresponding H_{ij} is computed. The difference in global congestions of the solutions before and after the move is measured, and $\Delta C(I)$ is updated.

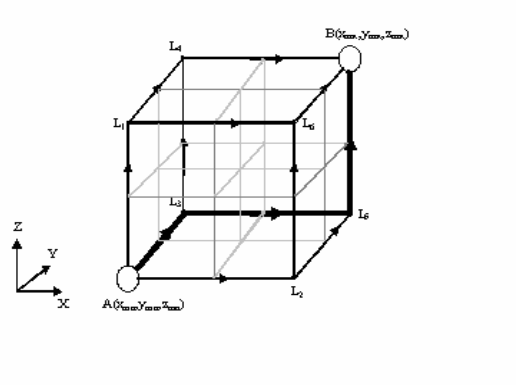

Figure 5. 3D Bounding Box Routing paths (eg. A-L_3-L_5-B).

b) Wirelength and Via Count

Given B_i and B_j, the blocks to be swapped for the I^{th} move at temperature T, we only update h_w and h_v, the wirelength and via contributions of every hyperedge h in H_{ij}. Thus, $\Delta V(I)$ and $\Delta W(I)$ can be incrementally updated at relatively low runtime costs.

Table 1. Benchmark circuits.

ckt	gates	nets
s5378	2828	3026
s9234	5597	5844
s13207	8027	8727
s15850	9786	10397
s35932	16353	18116
s38417	22397	24061
s38584	19407	20871
b14_opt	5401	5678
b15_opt	7092	7577
b17_opt	22854	24305
b20_opt	11979	12501
b21_opt	12156	12678
b22_opt	17351	18086

5. EXPERIMENTAL RESULTS

Our algorithms were implemented in C++/STL, compiled with gcc v2.96 with –O3, and run on Pentium III 746 MHz machines. The benchmark set consisted of seven circuits from ISCAS89 and six circuits from ITC99 suites. The relevant statistical information of the benchmark circuits is shown in Table 1. We ran our experiments using 4 x 4 x 4 block placement as well as 8 x 8 x 4 block placement. The pure mincut method was implemented using our own multilevel recursive bipartitioning with a (*z, x, y, ...)* cut sequence, which is a balanced combination of the two techniques suggested in [11]. The LC-CUT algorithm was run under the same framework as the aforementioned, with a w_α value of 3.5 and a w_β

value of 1. As shown in Figure 6, these weights tend to work best in terms of the overall quality of the solution.

5.1 Impact of LC-CUT on Congestion

As seen in Tables 2 and 3, LC-CUT achieves significant reduction in overall local congestion when compared to the traditional mincut approach. However, there is a slight increase in wirelength and via count, which adversely impacts the global congestion.

5.2 Impact of Simulated Annealing on Congestion

Simulated annealing achieves remarkable decrease in the global congestion without impacting local congestion. Due to the nature of the combined cost function, the wirelength and via count are also reduced to within 15% of the original mincut results.

6. CONCLUSIONS

We devised a two-step approach to reduce local and global congestion for three dimensional Integrated Circuits without adversely impacting the pure mincut wirelength and via results. The **LC-CUT** algorithm reduced local congestion by over 44% on average. The simulated annealing-based refinement improved global congestion by over 16% without affecting **LC-CUT**'s local congestion results. We also devised a new 3D cut sequence that allows for a balanced wirelength and via count. Our solution is flexible with respect to the desired amount of congestion minimization. We are currently developing efficient routing techniques to generate more accurate global congestion metrics.

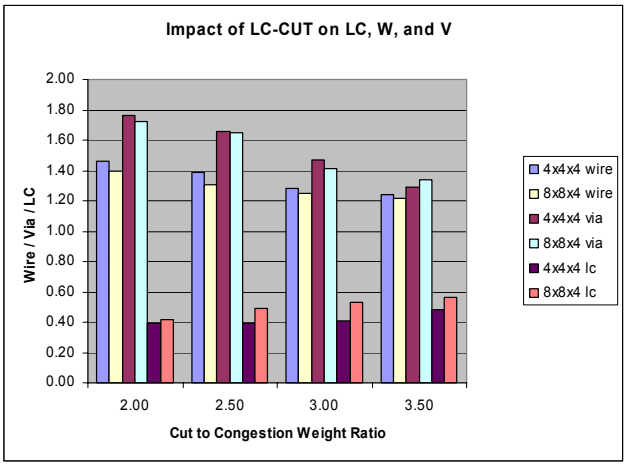

Figure 6. Impact of weight ratio on LC

Table 2. 4 x 4 x 4 Global Placement Results, with 2D wirelength (W), via count (V), local congestion (LC) and global congestion (GC)

	Pure Mincut				LC-CUT				LC-CUT with SA			
ckts	**W**	**V**	**LC**	**GC**	**W**	**V**	**LC**	**GC**	**W**	**V**	**LC**	**GC**
s5378	934	259	36	23	968	283	23	20	912	288	23	15
s15850	1072	330	162	28	1335	431	77	33	1245	427	77	26
s9234	888	253	123	33	1087	269	80	25	1169	252	80	19
s13207	967	271	297	32	1116	390	77	27	1022	379	77	20
b14_opt	2126	657	83	55	2635	808	31	56	2299	822	31	42
b15_opt	3365	915	116	89	3946	1181	62	89	3598	1186	62	78
b17_opt	6327	1681	350	176	9044	2415	201	220	7945	1755	201	161
b20_opt	3871	1071	226	75	4431	1085	67	80	4001	989	67	68
b21_opt	3630	1117	221	75	4199	1080	46	91	3783	1037	46	71
b22_opt	4541	1305	255	122	5076	1779	109	103	4812	1269	109	85
s38417	1403	421	242	33	1972	768	152	44	1574	794	152	32
s35932	1002	284	159	28	1397	428	150	39	1240	419	150	28
s38584	1770	504	281	43	2338	818	148	77	2207	688	148	41
avg	**2454**	**698**	**196**	**62**	**3042**	**903**	**94**	**70**	**2754**	**793**	**94**	**53**
wt avg	**1.00**	**1.00**	**1.00**	**1.00**	**1.24**	**1.29**	**0.48**	**1.11**	**1.12**	**1.14**	**0.48**	**0.84**
time(s)	1864				15289				37784			

Table 3. 8 x 8 x 4 Global Placement Results, with 2D wirelength (W), via count (V), local congestion (LC) and global congestion (GC)

	Pure Mincut				LC-CUT				LC-CUT with SA			
ckts	**W**	**V**	**LC**	**GC**	**W**	**V**	**LC**	**GC**	**W**	**V**	**LC**	**GC**
s5378	2193	229	22	26	2324	283	15	32	2105	265	15	16
s15850	2617	278	67	33	3139	399	36	50	2784	392	36	24
s9234	2183	245	61	34	2557	305	33	37	2039	308	33	31
s13207	2461	262	90	36	3002	382	35	45	2941	286	35	32
b14_opt	5298	700	31	60	6447	866	21	95	6355	851	21	48
b15_opt	7759	1023	43	90	8939	1112	31	101	8407	1004	31	88
b17_opt	15407	1681	123	182	20549	2989	84	226	16564	2344	84	159
b20_opt	9332	1071	59	97	10369	1292	31	140	10054	1148	31	68
b21_opt	8963	1122	66	88	10280	1174	34	140	9345	1014	34	65
b22_opt	10975	1346	92	143	13387	1773	53	177	12144	1058	53	97
s38417	3818	514	141	40	5138	769	67	72	4991	694	67	31
s35932	3113	354	71	35	3879	459	44	60	3208	415	44	29
s38584	4680	574	102	63	6426	798	55	92	6219	582	55	59
avg	**6061**	**723**	**74**	**71**	**7418**	**969**	**41**	**97**	**6704**	**797**	**41**	**57**
wt avg	**1.00**	**1.00**	**1.00**	**1.00**	**1.22**	**1.34**	**0.56**	**1.37**	**1.11**	**1.10**	**0.56**	**0.81**
time(s)	2034				21151				51746			

7. REFERENCES

[1] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and David B. Janes, ``Stochastic Wire-Length Distribution and Delay Distribution of 3-Dimensional Circuits," Proc. International Conference on Computer-Aided Design, November 2000, pp. 208-213.

[2] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and David B. Janes, ``Power Trend and Performance Characterization of 3-Dimensional Integration for Future Technology Generations," Proc. 2001 International Symposium on Quality of Electronic Design, March 2001, pp. 217-222.

[3] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and David B. Janes, ``Stochastic Interconnect Modeling, Power Trends, and Performance Characterization of 3-Dimensional Circuits," IEEE Trans. on Electron Devices, 48(4), April 2001, pp. 638-652.

[4] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and David B. Janes, ``Power Trend and Performance Characterization of 3-Dimensional Integration," Proc. 2001 International Symposium on Circuits and Systems, May 2001, Volume 4, pp. 414-417.

[5] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and David B. Janes, ``Exploring SOI Device Structures and Interconnect Architectures for 3-Dimensional Integration," Proc. 2001 Design Automation Conference, June 2001, pp. 846-851.

[6] Shukri J. Souri, Kaustav Banerjee, Amit Mehrotra, and Krishna C. Saraswat, "Multiple Si Layer ICs: Motivation, Performance Analysis, and Design Implications", DAC00.

[7] M Alexander, J Cohoon, J Colflesh, J Karro, E Peters, and G Robins, "Placement and Routing for Three-Dimensional FPGAs",

[8] Arnold Rosenberg, "Three-Dimensional VLSI: A Case Study", Journal of ACM, 1983.

[9] Thitipong Tanprasert, "An Analytical 3-D Placement That Reserves Routing Space", ISCAS00.

[10] Yangdong Deng, Wojciech Maly, "Physical Design of the 2.5D Stacked System", ICCD03.

[11] Shamik Das, Anantha Chandrakasan, Rafael Reif, "Design Tools for 3-D Integrated Circuits", ASPDAC03.

[12] Krishna C. Saraswat, K. Banerjee, A. R. Joshi, P. Kalavade, P. Kapur and S. J. Souri, " 3-D ICs: Motivation, Performance Analysis, and Technology."ESSCIRS 2000

[13] K. Banerjee,P. Kapur, S. J. Souri ,and Krishna C. Saraswat, " 3-D ICs: A Novel Chip Design for Improving Deep-Submicrometer Interconnect Performance and Systems-on-Chip Integration".Proceedings IEEE Vol. 89, 2001

[14] S. J. Souri,K. Banerjee,A. Mehrotra, ,and Krishna C. Saraswat, " Multiple Si Layer ICS: Motivation, Performance Analysiis, and Design Implications". DAC 2000 (pdf)

[15] M. C. Yildiz, and P. H. Madden, , " Improved Cut Sequences for Partitioning Based Placement". DAC01

[16] Maogang Wang and Majid Sarrafzadeh, "Congestion Minimization During Placement," Proceedings of International Symposium on Physical Design, 1999

[17] Xiaojian Yang, Ryan Kastner, and Majid Sarrafzadeh, "Congestion Reduction During Placement Based on Integer Programming", in Proc. International Conference on Computer-Aided Design, 2001.