VLSI Design
1995, Vol. 3, No. 1, pp. 53–57
Reprints available directly from the publisher
Photocopying permitted by license only

Input/Output Pad Placement Problem

KHALED AL-ZAMEL and MUKKAI S. KRISHNAMOORTHY
Department of Computer Science Rensselaer Polytechnic Institute, Troy, NY 12180

(Received September 29, 1993, Revised March 2, 1994)

We propose efficient heuristics for placing input/output (I/O) pads around the VLSI chip boundary. The heuristics are based on the circuit connectivity and do not require the placement of cells. This is useful for cell placement methods that require the knowledge of I/O pad placement. Using these heuristics, several pad placement candidates can be generated as input to cell placement algorithms.

Key Words: *VLSI; Placement; Pad placement; Layout; Floorplanning; Standard cell; Gate array.*

1. INTRODUCTION

A VLSI chip is a rectangular area with input/output (I/O) pads (we will refer to these simply as pads) arranged along the perimeter of the chip in which cells are laid out surrounded by routing channels. Cells (modules or blocks) are logic models of a rectilinear shape with a predefined internal layout. Nets are physical wires that provide interconnections between cells. They provide the physical means for logical signals to travel through. Nets are laid out on a reserved chip space known as a routing channel.

I/O pad placement is the problem of placing I/O pads around a VLSI chip boundary such that it leads to a reduction in the cost of cells placement. Pad placement is typically done as the last stage in a VLSI layout design. However, some cell placement packages are sensitive to the pad placement and require such placement to be done first [1].

To solve this problem Tsay et al. [6] proposed a similar approach of Hall's [4] by solving for the eigenvectors of the modified connectivity matrix given by the following equation.

$$B^* = B_{22} - B_{21}B_{11}^{-1}B_{12} \tag{1}$$

In practice, pad pair exchange produces a better and a more efficient placement. In this paper we present heuristics for pad placement. We also give some results obtained by running these heuristics over known industrial circuits.

2. CELL PLACEMENT

Cheng and Kuh [1] have devised a novel technique for cell placement. They have transformed the cell placement problem into the problem of minimizing the power dissipation in a resistive network. The objective function (square of Euclidean wire length) is given by

$$\Phi = \frac{1}{2}\sum_{ij} c_{ij}\left[(x_i - x_j)^2 + (y_i - y_j)^2\right] \tag{2}$$

Let C be the connection matrix. Let c_i be the sum of all elements in the ith row of C. Define a diagonal matrix D such that

$$d_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ c_i & \text{otherwise} \end{cases} \tag{3}$$

The matrix B is defined as

$$B = D - C \tag{4}$$

Further, let $X^T = [x_1, x_2, \ldots, x_n]$ and $Y^T = [y_1, y_2, \ldots, y_n]$ be the row vectors representing the x- and y-coordinates of the desired solution. Then the objective function can be rewritten as

$$\Phi = X^TBX + Y^TBY \tag{5}$$

This representation is similar to the one for resistive networks. The placement problem is solved by manipulating the corresponding network to minimize power dissipation using sparse matrix techniques. The equation for the power dissipation in a resistive network is

$$P = V^T Y_n V \tag{6}$$

We find that B is of the same form as the indefinite admittance matrix Y_n of an n-terminal linear resistive network. The coordinate x_i of cell i is analogous to the voltage at node i. The connectivity c_{ij} between cells i and j is analogous to the mutual conductance y_{ij} between nodes i and j, and d_{ii} is analogous to the self-admittance at node i. If the given netlist contains some fixed modules, such as pads, then that will be equivalent to having voltage sources in the resistive network. In a resistive network, currents always distribute so as to minimize the power dissipation. The current distribution can be computed using the two Kirchhoff laws,

$$B_{11}v_1 + B_{12}v_2 = 0 \tag{7}$$

$$B_{21}v_1 + B_{22}v_2 = i_2 \tag{8}$$

where v_1 and v_2 represent movable and fixed modules, respectively.

Tsay et al. [6] proposed a placement technique, *Proud*, based on the resistive network analogy. The slot constraints are bypassed, i.e. modules are considered as zero-area points and are not confined to the grid points. The method proceeds in two phases: First, slot-free placement is done by solving the linear system of equations represented by Eq. 7. Then, module shape and area are taken into consideration, and the chip is partitioned alternately in the vertical and horizontal directions along the cut axis. This process is repeated until each subregion consists of only one module.

The I/O pad placement is used as the seed for the cell placement. The final cell placement is usually very sensitive to the pad placement.

3. PAD PLACEMENT

In practice, the chip boundary is rectilinear in shape, but to simplify circuit analysis we assume that the boundary is circular. Pads are to be placed at the boundary forming a circular ring, with neighbors equally apart from each other. In our approach, pad placement is done in two phases. First we try to find a circular ordering based on the connection graph of the circuit. Second, we try to compute the optimal rotation of that ordering.

3.1 Circular Ordering

Let the distance d_{ij} between pads i and j be the shortest path length between i and j in the circuit connection graph $G(C, P, N)$, where C, P and N represent cells, pads and nets, respectively. A circular ordering is a sequence for pads around a circular boundary with their relative positions preserved. Our intuition is that distance dictates the ordering of the pads around the boundary. Pads that are graphically close to each other are more likely to occupy neighboring positions than pads that are graphically further apart.

Let the pad distance graph be a complete weighted graph where vertices represent pads and edge weights, d_{ij}, represent the length of the shortest path between pads i and j in the connection graph G. Note that this graph maintains the triangular inequality, i.e., $d_{ij} \leq d_{ik} + d_{kj}$ for all i, j and k. To construct such a graph we can use a modified version of Dijkstra's algorithm [3], to compute the single-source shortest-path for each pad in the connection graph. Dijkstra's algorithm runs in $O(N \log P + C)$, therefore the construction of the pad distance graph runs in $O(PN \log P + C)$.

Given a circular ordering we define the ordering length as the sum of the distances between adjacent pads in the ordering. The problem of finding a circular ordering with the minimum length is equivalent to finding the shortest traveling salesman tour (i.e., a closed tour that visits each vertex once and only once) in the distance graph. This problem is known to be NP-complete [5], but there exist fast approximation algorithms that find cycles with at most one and a half the optimal tour cost [2]. We present an algorithm to compute a circular ordering of the pads.

Algorithm: *Circular Ordering*
Input: $G(C, P, N)$
Output: circular ordering $Order(P)$
begin

1. Construct pad distance graph $D(P)$,
   d_{ij} = length of shortest path between i and j in G.
2. $Order(P)$ = traveling-salesman-tour($D(P)$)
   end

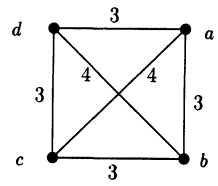

FIGURE 1 Pads distance graph.

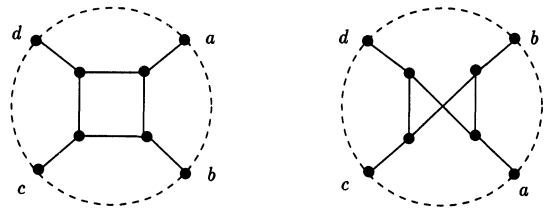

FIGURE 2 Slot-free cell placement.

Fig. 1 shows a pad distance graph for a circuit with pads *a*, *b*, *c* and *d*. Using Eq. 7 we computed the slot-free cell placement for pads ordering (*abcd*) and (*bacd*), and we get the placement shown in Fig. 2. We note that the pads ordering corresponding to the shorter tour gave a better placement.

3.2 Rotation of Pad Ordering

Empirically, we have observed that a good pads ordering is not sufficient to guarantee a good cell placement. For a cell placement algorithm like Proud, the alignment of the cut-axis is also crucial to obtain a good cell placement. Given a pads ordering the angle at which the cut-axis is aligned has a major effect on the final placement. Fig. 3 shows a pads ordering with the slot-free cell placement (computed using Eq. 7) and one possible cut-axis alignment.

To show the sensitivity of Proud to the cut-axis angle, we started with a pad placement and computed the cell placement using Proud, then we the cell placement for successive rotations by a small angle while preserving the pads ordering (similar in effect to changing the angle of the cut-axis). Table I shows the effect of pads ordering rotation for several circuits. In each circuit we used the procedure described above. Table I shows the worst and best cell placement obtained from pads ordering rotation. It also shows the percentage improvement of the best over the worst case. Clearly, we can obtain an improvement of as much as 17% on the cell

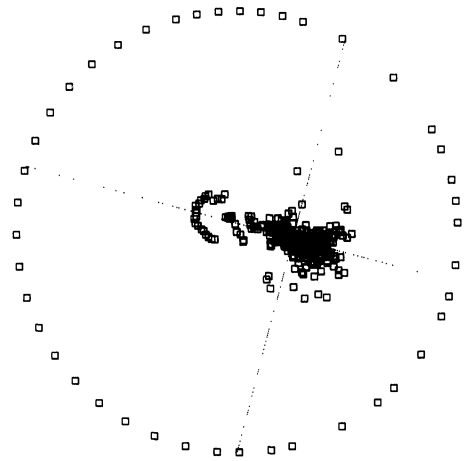

FIGURE 3 Global cell placement for a VLSI circuit.

TABLE I
Rotation data for several circuits

		rotation		
circuit	size	worst	best	diff. %
s953	424	4107	3395	17.3
primary1	752	8013	6800	15.1
mult16	1888	11915	9854	17.3

placement if we can tune the cut axis to the best angle of rotation.

The simplest way to find the best alignment angle for the cut-axis, can be done by incrementally rotating the pads ordering around the chip boundary and taking the alignment with the minimal cell placement cost. Unfortunately this procedure is computationally expensive because it requires a small rotational increment which in turn leads to many executions of the cell placement algorithm.

To approximate the alignment angle of the cut-axis, we compute the slot-free cell placement for the pads ordering using Eq. 7 (we assume that the chip boundary is circular). Note this does not involve computing the final cell placement. This gives us the optimal slot-free cell placement, where cells are

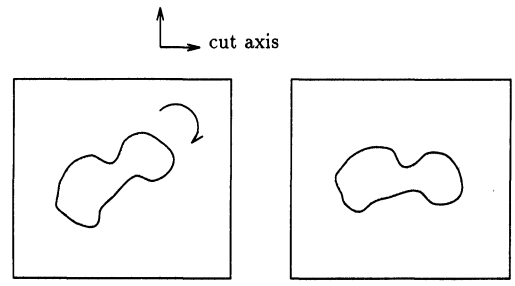

FIGURE 4 Aligning cluster orientation with cut axis.

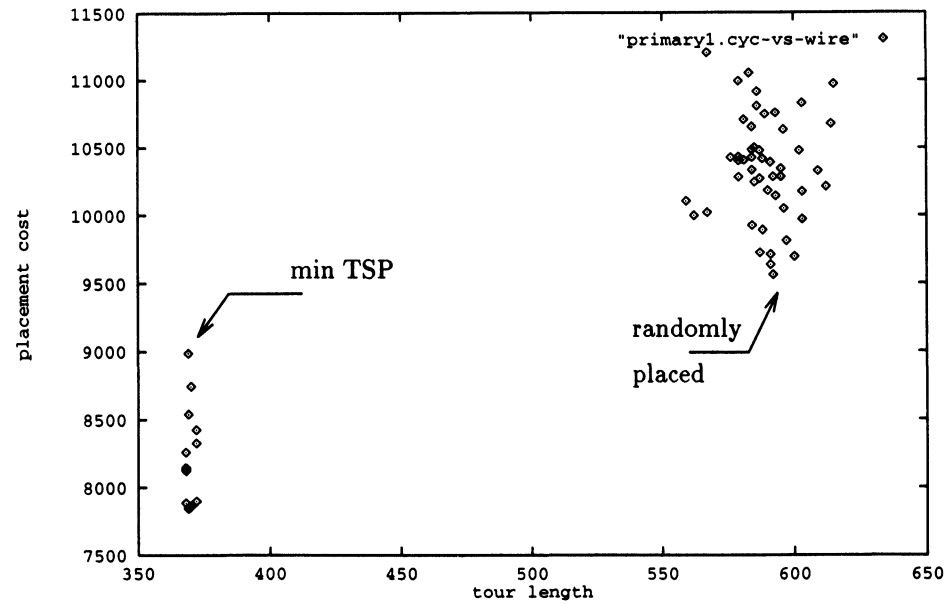

FIGURE 5 Placement quality vs. tour cost.

allowed to move freely without being confined to the slot constraints. Usually cells would cluster near the center of the layout. We think it is best to align the cut axis with the orientation of the main cluster [Fig. 3]. Aligning the cut axis with the orientation of the main cell cluster attempts to align the cluster with the chip boundary [Fig. 4]. Therefore as the cluster expands in later partitioning steps, it fills the chip area. To compute cluster orientation we used the second moment of the cluster distribution.

Algorithm: *Circular Ordering Orientation*
Input: $G(C, P, N)$ & $Order(P)$
Output: best rotation angle of $Order(P)$ θ
begin

1. Compute the slot-free placement using the linear system of eq.

$$B_{11}x_1 = -B_{12}x_2$$

2. θ = orientation of the main cluster.

end

4. RESULTS

In Section 3.1 we assumed that the quality of the final placement in Proud is dependent on the length of the circular ordering. Minimizing ordering length should lead to a better placement. To show such a relationship empirically, we randomly constructed several circular orderings for a given circuit and computed the final placement using Proud. We then constructed several circular orderings with minimum TSP tours. The placement quality versus the ordering length was then plotted. The relationship between the ordering length and the final placement is shown in Fig. 5. Clearly, Fig. 5 shows a direct relationship between the ordering length and the final placement. We can obtain as much as 17% improvement over the random pad placement.

In Section 3.2 we asserted that aligning the cut axis with the orientation of the main cell cluster in the slot-free placement would lead to a better placement quality. Given a circular ordering, we computed placements obtained by incrementally rotating the ordering by a few degrees covering a full cycle. Fig. 6 shows a plot of the placement of several circuits against the angle of the cut axis. Each curve was normalized by its mean value. The curves show a periodic relationship between the placement quality and the angle rotation. Table II shows statistics obtained from the pads rotation of several circuits. The standard deviation is about 5% of the mean. Fig. 6 also shows that the final placement is sensitive to the angle of rotation. So we expect that alignment of the cut-axis to the orientation of the main cluster would be a crude approximation and we might test few other angles in the neighborhood of the approximation.

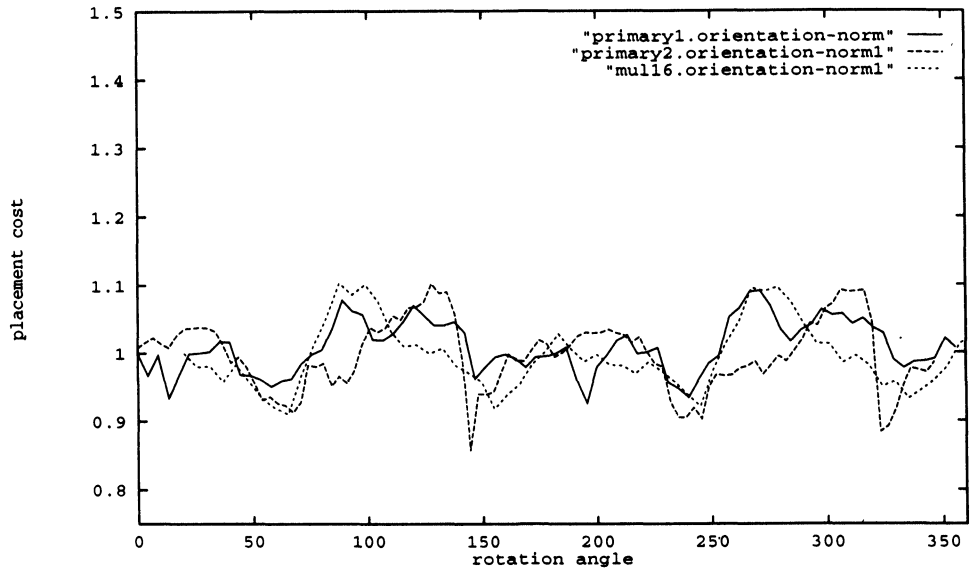

FIGURE 6 Final placement vs. angle of rotation.

TABLE II
Pads rotation statistics

circuit	size	min	max	std. deviation
s953	424	0.922	1.104	0.051
primary1	752	0.926	1.091	0.037
mul16	1888	0.910	1.103	0.048
primary2	2907	0.857	1.102	0.051

5. CONCLUSION

In this paper we present an I/O pad placement heuristics. These heuristics could be used as a preprocessing step for cell placement algorithms that are sensitive to the pad placement. Two heuristics were proposed for pad placement; namely, pads circular ordering and ordering angle of rotation. The combined heuristics may result in more than 20% improvement over using random pad placement.

Our heuristics are iterative in the sense that several pads ordering candidates (by using different TSP tours) can be used to generate good pad and cell placements.

References

[1] C. Cheng, and E. Kuh, "Module placement based on resistive network optimization," IEEE Trans. Computer-Aided Design CAD-3 (July 1984) pp. 218–225.

[2] N. Christofides, "Worst-case analysis of a new heuristic for the traveling salesman problem," Technical Report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, P.A., (1976).

[3] E.W. Dijkstra, "A note on two problems in connection with graphs," Numerische Mathematik (1959) pp. 269–271.

[4] K.M. Hall, "An r-dimensional quadratic placement algorithm," Manage. Sci. 17, 3 (Nov. '1970) pp. 219–229.

[5] R.M. Karp, "Reductibility among combinatorial problems, complexity of computer computations," R.E. Miller and J.W. Thatcher, eds., Plenum Press, New York, (1972) pp. 85–103.

[6] R. Tsay, E.S. Kuh, and C. Hsu, "Proud: a sea-of-gates placement algorithm," IEEE Design and Test of Computers (Dec. 1988) pp. 44–56.

Biographies

KHALED AL-ZAMEL received his B.S. degree in Electrical Engineering (with honors) from Kuwait University in 1987, and the M.S. degree in Computer Science from Rensselaer Polytechnic Institute, Troy, NY, in 1989. He is currently a Ph.D. candidate at Rensselaer. Al-Zamil's research interests are in CAD in VLSI, and parallel processing.

M.S. KRISHNAMOORTHY received the B.E. degree (with honors) from Madras University in 1969, the M. Tech degree in Electrical Engineering from the Indian Institute of Technology, Kanpur, in 1971, and the Ph.D. degree in Computer Science, also from the Indian Institute of Technology, in 1976.

From 1976 to 1979, he was an Assistant Professor of Computer Science at the Indian Institute of Technology, Kanpur. From 1979 to 1985, he was an Assistant Professor of Computer Science at Rensselaer Polytechnic Institute, Troy, NY, and since, 1985, he has been an Associate Professor of Computer Science at Rensselaer. Dr. Krishnamoorthy's research interests are in the design and analysis of combinatorial and algebraic algorithms and programming environments.