General Floorplans with L/T-Shaped Blocks Using Corner Block List

Yu-Chun Ma (马昱春)¹, Xian-Long Hong (洪先龙)¹, She-Qin Dong (董社勤)¹, C. K. Cheng², and Jun Gu (顾骏)³
¹Department of Computer Science and Technology, Tsinghua University, Beijing 100084, P. R. China
²Department of Computer Science & Engineering, University of California, San Diego, La Jolla, CA 92033-0114, U.S.A.
³Department of Computer Science, Hong Kong University of Science & Technology, Hong Kong Special Administration Region P. R. China

E-mail: mayuchun@tsinghua.org.cn; {hx1-dcs, dongsq}@tsinghua.edu.cn
Received September 1, 2004; revised February 28, 2006.

Abstract With the recent advent of deep submicron technology and new packing schemes, the components in the integrated circuit are often not rectangular. On the basis of the representation of Corner Block List (CBL), we propose a new method of handling rectilinear blocks. In this paper, the handling of the rectilinear blocks is simplified by transforming the L/T-shaped block problem into the align-abutment constraint problem. We devise the block rejoining process and block alignment operation for forming the L/T-shaped blocks into their original configurations. The shape flexibility of the soft blocks, and the rotation and reflection of L/T-shaped blocks are exploited to obtain a tight packing. The empty rooms are introduced to the process of block rejoining. The efficiency and effectiveness of the proposed method are demonstrated by the experimental results on a set of some benchmark examples.

Keywords floorplanning, corner block list, L/T-shaped blocks

1 Introduction

Floorplanning is one of the most significant design stages in VLSI design. Its main objectives are area minimization, wire length minimization, and timing optimization. Corner Block List (CBL)¹ was proposed as a topological representation for general floorplans which needs a smaller amount of encoding storage and linear time computation to generate each placement configuration.

With the recent advent of deep submicron technology and new packing schemes, the L/T-shaped blocks have been specifiable and/or permissible, for the speciality is motivated by the enforced colocuation of a data path block with related control, or the shape of particular types of data path blocks². A good floorplanner should provide not only a good rectangle packing functionality but also the flexibility to handle L/T-shaped blocks to achieve high-performance design. The previous effort to the rectilinear block problem was mainly concentrated on the differences between the feasible solutions and infeasible solutions represented by SP³⁻⁵ or by BSG⁶. In [⁵], the local moves of stochastic search are used to transform an infeasible solution into a feasible one, which is very time-consuming. Another kind of method was to shift the blocks to avoid the overlapping by taking some post-processes⁷, but this method may not represent the optimum packing as the authors pointed out. In [⁸, ⁹], the processes of sub-blocks rejoining are operated based on the slicing tree, but they can only handle the slicing structure. In this paper, the handling of the rectilinear blocks is simplified by transforming the L/T-shaped block problem into the align-abutment constraint problem. Since our algorithm is based on the topological structure of the packing, the softening, reflection and rotation of L/T-shaped blocks can be easily implemented. Our algorithm has been implemented in the C programming language. The efficiency and effectiveness of the proposed method are demonstrated by the experimental results of MCNC benchmarks.

The rest of the paper is composed as follows. Section 2 is a brief review of the CBL model. A formulation of the floorplan with L/T-shaped blocks is described in Section 3. The new algorithm is presented in Section 4. The experimental results are shown in Section 5. Finally, the conclusion is given.

2 Corner Block List

A floorplan divides the chip into rectangular rooms with horizontal and vertical segments. Each room is assigned to no more than one block. Each pair of intersected segments forms a T-junction. A T-junction is composed of two segments: a non-crossing segment and a crossing segment. The non-crossing segment has one end touching point along the crossing segment. CBL is derived from a simplified version of general floorplan called mosaic floorplan. Corner Block is the block packed in the upper right corner room of the floorplan. In the floorplan, the joint of the left and bottom segments of the corner block is contained in a T-junction named corner T-junction, and the corner block’s orientation is defined by the orientation of the corner T-junction. The T-junction has only two kinds of orientations: T rotated by 90 degree (see Fig.1(a)) and by 180 degree (see Fig.1(b))
counter-clockwise respectively. If T is rotated by 90 degree counter-clockwise, we define the corner block as vertical oriented, and denote it by a “0”. Otherwise, the corner block is horizontal oriented, and we denote it by a “1”. The corner block list is constructed from the record of a recursive corner block deletion. When the corner block d is deleted, the attached T-junctions, whose crossing segments are the non-crossing segment of the corner T-junction, are pulled up to the top boundary of the chip. The insertion of the corner block is the inverse of the deletion. We use a binary list T_i to record the number of the attached T-junctions of the deleted corner block M_i. The number of successive 1s, which is ended by a “0”, corresponds to the number of the attached T-junctions.

For each block deletion, we keep a record of the block name, the corner block orientation, and the sequence of T_i. At the end of deletion iterations, we can obtain three lists: block name list {B_n, B_{n-1}, ..., B_1}, orientation list {L_n, L_{n-1}, ..., L_2}, T-junction list {T_n, T_{n-1}, ..., T_2} (n is the number of the blocks). We reverse the data of these three items respectively. Thus, we have a sequence (S) of block names, a list (L) of orientations, and a list of {T_2, T_3, ..., T_n} which is combined into a binary sequence (T). The three-element triple (S, L, T) is called a corner block list. The insertion process of corner block based on given (S, L, T) can construct the corresponding floorplan. Fig.2 is an example of a non-slicing floorplan and its corresponding CBL.

3 L/T-Shaped Blocks

To handle the L/T shaped blocks, we partition these blocks into rectangular blocks, which are called sub-blocks to distinguish them from individual circuit blocks.

- An L-shaped block should be partitioned into two abutted rectangular sub-blocks, L_1 and L_2, where one boundary of L_1 and one boundary of L_2 are aligned horizontally (vertically) (see Fig.3(a)).

- A T-shaped block should be partitioned into three abutted rectangular sub-blocks, T_1, T_2 and T_3, where one boundary of T_1, one boundary of T_2 and one boundary of T_3 are aligned horizontally (vertically), and the height (width) of the middle one is greater than the height (width) of the other two sub-blocks (Fig.3(b)).

Observation 1. The sub-blocks of L/T-shaped blocks must possess the following properties: 1) abutment, 2) alignment.

4 Our Approach

An L/T-shaped block is partitioned into abutted sub-blocks with edges aligned. After partitioning, we have transformed the floorplanning problem of L/T-shaped blocks into that of rectangular blocks with alignment constraints.

4.1 Preliminaries

Definition 1 (Align-Abutment). If blocks A and B abut with each other and one boundary l_a of A and one boundary l_b of B are aligned along the same segment, blocks A and B are said to have an align-abutment relation with each other. Each block has eight positions for align-abutment (Fig.4).

Definition 2. V_i[M_i] denotes the block which is at the position of V_i, whose left boundary is aligned with the Left boundary of M_i and V_i[M_i] is lying Above M_i, V_2[M_i], V_3[M_i], V_4[M_i], H_1[M_i], H_2[M_i], H_3[M_i], and H_4[M_i] are defined similarly. Block M_i is called master block since all of its abutted blocks have relative abutment relation to M_i.

4.2 Algorithm

An L/T-shaped block can be partitioned into a set of sub-blocks with horizontal or vertical lines. There are eight ways in which the two sub-blocks can form an L-shaped block (Fig.5) and T-shaped block.

Fig.4. Positions of align abutment.

Fig.5. Partition of L-shaped blocks.