An Efficient Tile-Based ECO Router Using Routing Graph Reduction and Enhanced Global Routing Flow

Yih-Lang Li, *Member, IEEE*, Jin-Yih Li, and Wen-Bin Chen

***Abstract*—Engineering change order (ECO) routing is frequently requested in the later design stage for the purpose of delay and noise optimization. ECO routing is complicated as a result of huge existing obstacles and the requests for various design rules. The tile-based routing model results in fewer nodes of the routing graph than grid and connection-based routers; however, the number of nodes of the tile-based routing graph has grown to over a billion for system-on-chip designs, while no notable progress has been achieved in the routing speed of the tile-based router since it was proposed. This paper first proposes a novel routing graph reduction (RGR) method for promoting tile propagation speed and then depicts a new ECO routing design flow with RGR and enhanced global routing flow (EGRF). RGR can be used to remove redundant tiles as well as align and merge neighboring tiles in order to diminish tile fragmentation such that the tile-based ECO router can run twice as fast while still producing an optimal path. Compared with a commercial placement and routing tool, the proposed tile-based router with RGR obtains better routing performance and routing quality for three ECO routings. EGRF incorporates ECO global routing considering via-resource congestion metric with extended routing and global cell (GCell) restructuring to prevent routing failure in routable designs. The ECO router with the proposed design flow can perform up to 20 times faster than the original tile-based router at the cost of only a slight decline in routing quality. Experimental results also demonstrate that a more congested layout tends to have higher graph reduction rate. Also discussed herein are further refinements by dynamic weighting of via and wire resources based on the vacancy density of the routed design and further application of RGR to multiple-net routing.**

***Index Terms*—Deep submicrometer, detailed routing, engineering change order (ECO) routing, global routing, gridless routing, layout, physical design, system-on-chip.**

I. Introduction

THE NANOMETER scale in semiconductor technology has resulted in an increased number of components embedded in an integrated circuit. Small transistors possess a small gate channel length and, thus, a short travel distance for electrons and holes such that transistors switch faster than before. The increase in components enriches circuit functions, whereas fast switching causes the operating clock frequencies of designs to continuously increase. However, for interconnections, slim wires typically have high resistance since the interconnection resistance is inversely proportional to wire width and height. To overcome this side effect, wires tend to be designed with high wire height to wire width (height/width) aspect ratios. High-performance interconnection design methods, such as topology optimization, device and wire sizing, and high-performance clock routing, have been proposed to deal with the increasing wire delay [1]. A tall and slim wire is effective for resolving the problem of increasing wire resistance; however, this solution increases the coupling capacitance between wires. A decreasing wire separation increases the coupling capacitance, which is a source of signal integrity problems. The wide wire separation rule is then usually applied to the victim wire for reducing the coupling effect.

A. Engineering Change Order (ECO) Routing

ECO operations include device editing (insertion/deletion), wire rerouting using area constraints to guide path search, and larger width and separation rules, and are frequently applied to eliminate excess delay and noise and enhance circuit functionality following placement, routing, and compaction operations. Efficient ECO operations can rapidly modify designs to meet desired constraints such that the need to restart a new design cycle can be avoided. ECO routing, i.e., normally a point-to-point routing operation, is considered to refer to any wire rerouting for ECO operations. For example, a rising transition on a victim wire can be postponed if its aggressor wires switch in the opposite direction. Using interactive ECO editing commands to widen the separation of the victim wire to all aggressor wires can solve the excess delay problem resulting from the crosstalk effect. Unfortunately, in most congested designs, finding sufficient available space around the victim and aggressor wires is extremely difficult. Rerouting some or all of the whole victim wires or rerouting some major aggressor wires with wider separation rule can solve this problem.

B. Gridless ECO Routing

When the delay and noise optimization objectives of ECO routing are considered, different wire width and separation

Manuscript received June 23, 2005; revised September 19, 2005 and December 20, 2005. This work was supported in part by the National Science Council of Taiwan under Grant NSC 92-2220-E-009-031, Grant NSC 93-2220-E-009-021, and Grant NSC 94-2220-E-009-021. This paper was recommended by Associate Editor P. H. Madden.

Y.-L. Li is with the Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: ylli@cs.nctu.edu.tw).

J.-Y. Li is with the Design Automation Department, Taiwan Semiconductor Manufacturing Company Ltd., Hsinchu 300, Taiwan, R.O.C. (e-mail: jyliv@tsmc.com).

W.-B. Chen is with Global Unichip Corporation, Hsinchu 300, Taiwan, R.O.C. (e-mail: gis92623@cis.nctu.edu.tw).

Digital Object Identifier 10.1109/TCAD.2006.883923

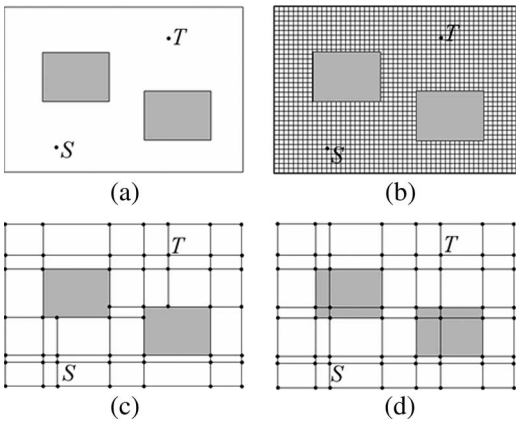


Fig. 1. (a) Routing example for gridless models. (b) Manufacturing grid model. (c) Connection graph model. (d) Implicit connection graph model.

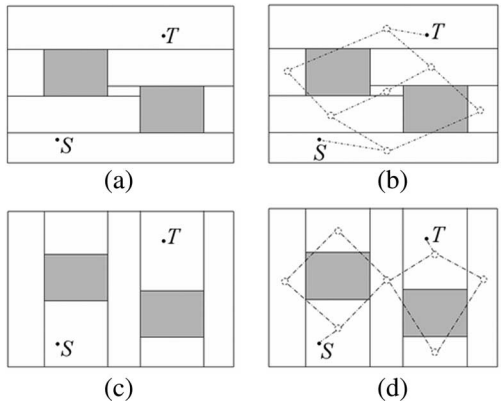


Fig. 2. (a) Maximum horizontally stripped tile plane. (b) Related routing graph in (a). (c) Maximum vertically stripped tile plane. (d) Related routing graph in (c).

rules are frequent requests. Traditional grid routers are highly effective means of solving routing problems involving fixed wire width and separation rules. Although grid routers can be enhanced to deal with different rules, wasted space is inevitable. A gridless router is more flexible than a grid router in accommodating different wire width and separation rules. Therefore, this paper proposes an efficient gridless point-to-point router, which is a major kernel technique for ECO routing. Besides the changes in design rules, the large numbers of interconnections and the flattened traverse operation for the possible reextraction and reconstruction of a routing environment make ECO routing highly complicated. For instance, if the compactor does not share the same internal database as the router, a compaction operation invalidates the efficient ECO routing functions supported by modern placement and routing environments, thus necessitating routing environment reextraction and reconstruction. Similarly, performing ECO routing on designs that come from the migrated layout with new technology or hard IP reuse also requires extracting the huge existing interconnections and then constructing the routing environment. Therefore, a survey on which type of gridless routers are qualified to perform ECO routing must consider routing environment construction and path searching, with the latter dominating the total computation time.

C. Comparisons of Gridless Routers

A straightforward realization of the gridless router uses fine uniform grids, or manufacturing grids, as shown in Fig. 1(b). Although capable of accommodating various routing rules, the induced huge routing graph for a large design makes this method inapplicable because it requires too much searching time and memory space. Of the different models investigated to reduce the routing graph [2]–[15], the connection graph and the tile-based graph are the most popular types. Zheng *et al.* [5] constructed a connection graph by extending lines through the boundaries of all obstacles until they intersect with other obstacles or boundaries of the routing region, as shown in Fig. 1(c). Owing to its irregularity, the graph suffers from inconvenient preconstruction and graph representation; however, the legality of all reachable nodes during path searching is guaranteed. A novel implicit connection graph whose extended lines may pass through obstacles is presented in [6]–[8], while [6] focuses on ECO routing. The penetrating lines form a regular routing graph, which can be easily and rapidly represented in matrix form. Fig. 1(d) displays a routing graph for the implicit connection graph model. Obviously, not every node in the graph, if included in the search path, is guaranteed to be free of design rule violations. Therefore, a legality check is performed each time an unvisited node is visited during path searching. The implicit connection graph model guarantees to identify an optimal path. However, this model is limited in that it has more nodes and edges than that in [5]. Actually, the connection graph model generates too many graph nodes to have a fast path search for a large design, especially for the case where the width of the metal enclosing a via exceeds the width of a metal wire. For instance, the implicit connection graph model generates a total of 536 214 892 nodes in its routing graph for a real design C2, thus serving as a test case in this study; meanwhile, the tile-based graph model only produces 3 173 533 tiles.

The tile-based model [9]–[15] is another gridless approach in which the routing region is partitioned by obstacles into space tiles and block tiles and represented using a corner-stitching data structure [16]. Fig. 2(a) and (c) presents a horizontal and a vertical tile plane with maximum horizontally and vertically stripped property, respectively, where, for example, the maximum horizontally stripping is referred to as the manner of partitioning the tile plane by extending the horizontal borderlines of all obstacles until they intersect with other obstacles or boundaries of the routing region. A space tile corresponds to a node of the tile-based routing graph, and an edge is present between two nodes if the related space tiles are adjacent to each other. Fig. 2(b) and (d) presents the related routing graphs in Fig. 2(a) and (c), respectively. Tile propagation between two adjacent tiles of a layer or two overlapping tiles of neighboring layers is used to seek the path. Xing and Kaog [15] proposed a precise piecewise linear cost model with linear minimum convolution (LMC) for calculating the accumulated path cost in a piecewise linear function from tile-to-tile and guiding the search for the shortest path. Consequently, both the implicit connection graph and the tile-based routers can find an "optimal" path for point-to-point routing.

Compared with the implicit connection graph router, the maximum horizontally or vertically stripped property in the tile plane results in fewer tiles on a tile plane, or fewer nodes in the related graph, and a larger distance for further tile propagation than for a grid move. However, the tile-based router requires additional memory resources for storing tiling data structure and more construction time for corner-stitching tile planes. Meanwhile, the interval tree used in the implicit connection graph router, which has an average complexity of only logarithmic order for most of the fundamental operations, is superior to corner stitching with an average complexity of sublinear order. The implicit connection graph router is therefore faster than the tile-based router in establishing routing environments and querying operations. Generally, the path searching time takes most of the time required for a point-to-point routing, except for cases involving extremely short-distance routing within a very large design. The tile-based router is thus more appropriate for ECO routing than the implicit connection graph router.

D. Acceleration of ECO Routing

Ongoing progress in semiconductor technology integrates more and more devices in a chip. Modern design, such as a chipset design, can yield several billion tiles for a single routing layer. For full-chip routing, the multilevel framework can be introduced to remarkably promote the performance of a routing system [17]–[20]. In [17] and [18], the multilevel framework is effectively applied to global routing. In [19] and [20], global routing and detailed routing algorithms were first integrated well into a multilevel framework to gain significant improvement in routing performance and completion rate. However, all detailed routings during the coarsening and uncoarsening stages are performed at the primitive level of the grids, and the routing grids remain unchanged. If detailed routing is performed on different levels, the performance can be improved if the identified coarse path can be rapidly uncoarsened to a legal path in the primitive level. If a legally finer path cannot be realized in the current level during the uncoarsening stage, local modification on current or previous coarser level is required. For a multilevel tile-based ECO router, refining a coarse path to a finer level can fail due to the difficulty in precisely determining the coarse tile type (routable or blocked) and whether it is routable between two coarse tiles on adjacent layers. Designing an efficient multilevel gridless ECO detailed routing is extremely challenging. Routing graph reduction (RGR) or a newly novel routing algorithm provides a better way to accelerate optimal ECO routing.

E. Paper Contributions and Organization

This paper first presents a novel RGR method for accelerating tile propagation by simplifying the tile plane and its associated routing graph to improve the tile propagation performance of a tile-based ECO router by roughly two times without sacrificing routing quality. This paper represents the first significant improvement to tile-based router performance since it was proposed. The proposed tile-based router with RGR is also superior to a commercial placement and routing tool in both routing performance and quality for three ECO routings. RGR can also be applied to improve the performance of general-purpose tile-based routers; the application of RGR is discussed in Section VII. A new ECO routing design flow comprising RGR and enhanced global routing flow (EGRF) that reduces path searching time by around 86% with minimal cost to routing quality is then presented. The rest of this paper is organized as follows: Section II briefly reviews the corner-stitching data structure and tile-based router and defines the basic terminology. Section III presents the new design flow for increasing the ECO routing speed. Section IV presents the RGR, and Section V presents the EGRF. Section VI discusses the experimental results obtained using two real designs. Section VII discusses possible refinements and further applications of this study. Finally, Section VIII draws conclusions.

II. Preliminaries

A. Corner-Stitching Data Structure

Ousterhout [16] proposed the corner-stitching data structure to efficiently manipulate rectilinear layout shapes. Each rectilinear shape is first partitioned into several disjoint tiles. Each tile contains four pointers, called "corner stitches," linking it with its four neighbors. The four pointers are rt, tr, lb, and bl, and they point to the rightmost top neighbor, the topmost right neighbor, the leftmost bottom neighbor, and the bottommost left neighbor, respectively. All tiles are linked using the corner stitches. The currently processed tile is named the "active tile." Each tile plane contains a hint tile to preserve the final active tile from the previous operation. Operations begin from the hint tile. Basic operations for corner-stitching data structure include point finding, neighbor finding, and area enumeration. Point finding locates the tile containing the target point. Neighbor finding obtains all the tiles neighboring a specific tile on one side. Area enumeration numbers exactly once the tiles within or intersecting with a given tile window. For routing applications, point finding is typically utilized to locate the start tile for a current operation. Neighbor finding is employed to locate all neighboring space tiles to the space tile in front of a routing path for further tile propagation during path searching. Area enumeration is commonly applied to search for the legal space tiles on adjacent layers for layer switching.

B. Tile-Based Router

Tile-based routers have been developed and thoroughly investigated [9]–[15]. Fast queries and powerful geometric Boolean operations on the corner-stitching tile plane make routing efficient. The concept behind the tile-based router is to route the centerline of a path along the space tiles on corner-stitching tile planes that are generated by adding the contours of width $w_s + w_w/2 - 1$ to all existing shapes, where w_s is the space rule of the net to be routed to all shapes of the same layer as the routed net, w_w is the wire width, and the subtraction is to make sure the contour insertion will produce a space tile of width 2 if the separation between two shapes exactly equals $2 \times w_s + w_w$. For example, in Fig. 3(a), paths $P1$ and $P2$ are two existing paths, and path new_p is the new path for insertion.

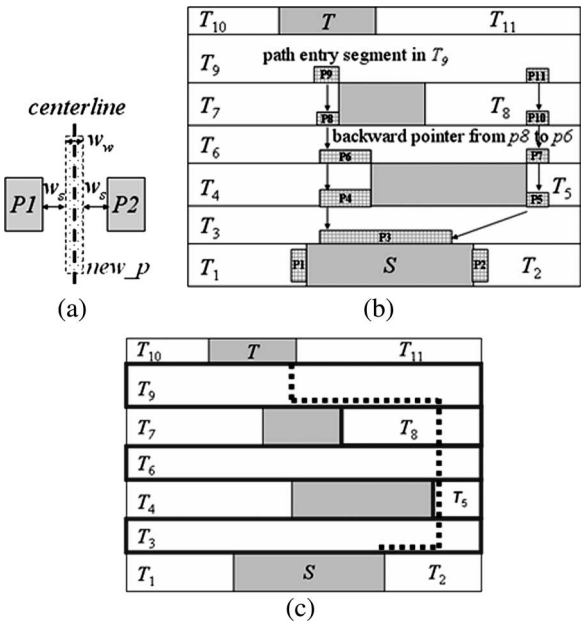


Fig. 3. (a) Centerline routing model for the tile-based router. (b) Example of single-layer tile propagation. (c) Example of path construction.

Obviously, no space tile exists between paths $P1$ and $P2$ if the contours of width $w_s + w_w/2$ are added to $P1$ and $P2$. The contours can guarantee that the newly created path will not induce any design rule violation. In multilayer routing, the available via regions at which the center of a new via can be placed can also be calculated in a similar way.

Tile-based point-to-point routing consists of two stages, namely: 1) tile propagation and 2) path construction [14]. Tile propagation is undertaken to reach the target in all possible ways over the space tiles of a single layer and across adjacent layers. Single-step tile propagation is from the current space tile to its neighboring space tile of the same layer or an adjacent layer; the neighboring space tile of an adjacent layer is the adjacent-layer space tile that can accommodate a via that connects the current space tile to itself. The tiles abutting the start blockage have a zero cost entry segment. In Fig. 3(b), tiles T_1, T_2, and T_3 abut the start blockage S and contain $p1$, $p2$, and $p3$ entry segments, respectively. A "path entry segment" is created in the target tile when a source tile propagates to it; furthermore, a backward pointer, indicated in Fig. 3(b) by arrows, is created and points from the new path entry segment to that in the source tile. Tile propagation, for example, from T_3 to T_4 in Fig. 3(b), induces a backward pointer pointing from path entry segment $p4$ to $p3$. The range of the new path entry segment indicates the range with minimum distance to the path entry segment of the path in the source tile. A tile allows the passage of multiple paths; therefore, a tile can have multiple path entry segments. In Fig. 3(b), tile T_6 contains two path entry segments $p6$ and $p7$ representing two different routing paths. Tile propagation applies a predefined cost function to guide the search for the path; it then finds a list of free tiles. In multilayer routing, the tile list may include tiles in different layers. When the target is found, the path can be obtained by traversing the visited tile list through the backward pointers. Finally, path construction generates a minimum-corner path that passes through the list of tiles. In Fig. 3(c), the tile list, comprising T_3, T_5, T_6, T_8, and T_9, can be obtained by traversing the path entry segments in the order of $p11$, $p10$, $p7$, $p5$, and $p3$; a minimum-distance and minimum-corner path can then be found, as shown in Fig. 3(c) by a bold dotted line.

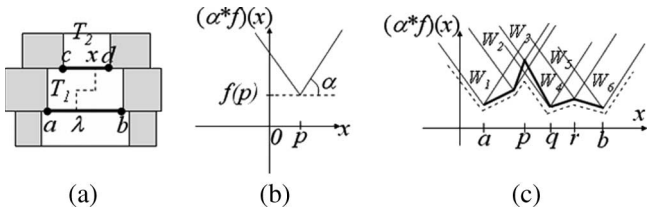


Fig. 4. (a) Example of source cost to the parallel edges of space tiles. (b) LMC kernels for $x = a$. (c) LMC of a piecewise linear function.

In [14], each edge of a space tile can have several path entry segments for storing different path costs. The range of a new path entry segment only contains the range that has a minimum distance to the path entry segment in the source tile. The path $p3{-}p5{-}p7{-}p10{-}p11{-}T$ in Fig. 3(b) can reach the target. Each space tile is assumed to have different horizontal costs. The path cost cannot precisely represent the minimum cost since the horizontal segment of this path can be realized in tiles T_6, T_8, or T_9. In [15], each edge uses a continuous function to represent the source cost function for all incoming paths. A similar case to [15] is used here to briefly illustrate how a tile-based router identifies an optimal path. In Fig. 4(a), the bottom edge of space tile T_1 records the source cost of the path from its bottom space tile, while the bottom edge of space tile T_2 records the source cost of the path from T_1. The vertical and horizontal costs in T_1 are α and β, respectively, and the height of T_1 is h. Thus, the source cost for the interval $[c, d]$ is

$$g(x) = \min_{\lambda \in [a,b]} \{f(\lambda) + \alpha|x - \lambda|\} + \beta h$$

where $f(\lambda)$ denotes the source cost for the interval $[a, b]$, βh is constant, and the function $\min_{\lambda\in[a,b]}\{f(\lambda) + \alpha|x - \lambda|\}$ is continuous. The LMC of weight $\alpha \geq 0$ with function $f(x)$ is denoted and defined as $(\alpha * f)(x) = \min_{\lambda\in[a,b]}\{f(\lambda) + \alpha|x - \lambda|\}$. Assuming the entry point on the interval $[a, b]$ is fixed at a point p, which is within the interval $[c, d]$, then the LMC is reduced to $\alpha|x - p| + f(p)$, where the term $f(p)$ is a constant. The function value is minimized when $x = p$ because the entry points on two parallel intervals align and no additional horizontal costs exist. When $x > p$, LMC increases along a line with a slope α; when $x < p$, LMC increases along a line with a slope $-\alpha$. This wedge-shaped function is referred to as the LMC kernel, as shown in Fig. 4(b). As x varies on the interval $[c, d]$, one LMC kernel can be derived for each fixed x. The LMC $(\alpha * f)(x)$ is then determined based on the minimum of all the LMC kernels. The dashed lines in Fig. 4(c) indicate the contour of LMC, while t he bold lines show the piecewise linear function of $f(x)$. An important property is stated as follows: if $f(x)$ is piecewise linear, then so is LMC $(\alpha * f)(x)$. Therefore, the source costs of all paths to reach an edge can be accurately preserved, and the path of the minimum value among all LMC kernels on the edge of the target tile can be selected to yield an optimal path.

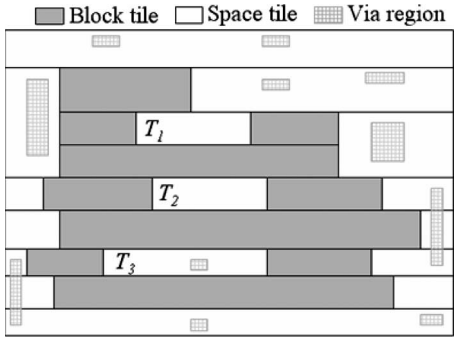


Fig. 5. Example for conjunct tile.

C. Terminology

The following definitions are used in this paper.

1) Essential/Redundant Tile: If a space tile can contribute to further propagation, it is called an "essential tile"; otherwise, it is called a "redundant tile."
2) Conjunct Tile: A tile A is referred to as a "conjunct tile" of tile B if a single-step tile propagation from A to B on the same layer or across adjacent layers is feasible.
3) One-Conjunct: A space tile is said to be "one-conjunct" if it has only a single conjunct tile.
4) 0-Conjunct: A space tile is said to be "0-conjunct" if it has no conjunct tile.
5) Global Cell: An entire layout is partitioned into tiles. Each tile is referred to as a "global cell" ("GCell").
6) Active GCell: Connected GCells, the result of ECO global routing, which are used to guide tile propagation, are called "active GCells."
7) Idle GCell: GCells that are not active GCells are called "idle GCells."

Fig. 5 illustrates these definitions. Fig. 5 depicts three redundant tiles, i.e., T_1, T_2, and T_3. The via regions are superimposed on the tile plane to indicate that the tile plane can be accessible from the tiles of other layers through the via regions. For instance, T_1 and T_3 can be reached by tile propagation from their top neighboring space tile and a neighboring space tile of an adjacent layer, respectively; however, tile propagation from T_1 or T_3 cannot explore any new space tile. Accordingly, T_1 and T_3 are the one-conjunct space tiles, and tile T_2 is 0-conjunct. Notably, T_3 is accessible only from a tile of another layer through the via region. If T_3 is accessible from more than two tiles of other layers through the via region, then it is an essential tile.

III. ECO Routing Design Flow

The tile-based router comprises three stages, namely: 1) corner-stitching tile plane construction; 2) tile propagation; and 3) path construction. The first part of this study reduces the complexity of the tile plane or routing graph nodes following tile plane construction and before tile propagation by means of RGR or tile plane simplification (TPS). A new ECO routing design flow, which contains RGR and EGRF, is then proposed. The new ECO routing design flow introduces RGR following corner-stitching tile plane construction, and then an iterative EGRF is applied after RGR. RGR reduces the complexity of the tile planes, while EGRF constrains the path search regions on the tile planes. RGR includes the removal of redundant tiles and the alignment of neighboring tiles; the former removes space tiles that do not contribute to further tile propagation, and the latter shrinks space tiles in an attempt to merge adjacent block tiles. The EGRF incorporates ECO global routing with extended routing and GCell restructuring to prevent routing failure in routable designs. Whenever tile propagation fails to identify a feasible solution, extended routing expands the GCell where routing failure occurs to increase the search space and thus the chance of completing the routing. If extended routing also fails to obtain a feasible solution, a new global routing is initiated. Before initiating a new global routing, GCell restructuring is employed to mark unroutable orders in the GCell list to prevent their reselection during subsequent global routing. GCell restructuring is realized by introducing six internal edges, namely: 1) nw; 2) ws; 3) se; 4) en; 5) ns; and 6) ew, for each GCell to configure the connectivity between any two neighboring GCells. The two endpoints of an internal edge for a GCell show the two neighbors of the GCell; whereas, the "connected" or "disconnected" status of an internal edge indicates whether a path crosses the GCell to connect relative neighbors, where all internal edges are assumed to be initially connected. For instance, a disconnected nw edge of a GCell, say, GA, represents an unroutable connection between the partition's north and west boundaries through GA. If the left and top adjacent GCells of GA are GL and GT, the subsequent global routing will not select the GL–GA–GT order of the GCell list. New global routing uses GCells visited during previous tile propagation as start points for yielding new results following GCell restructuring. More visited GCells will be included as start points for the next global routing if global routing continues to iterate. This process is repeated until a feasible path is identified or all GCells have been visited and no feasible solution is found. Fig. 6 depicts the proposed new ECO routing design flow, where the original design flow of the tile-based router consists of the functional blocks outlined in bold, and RGR and EGRF are outlined by dotted lines.

IV. RGR

The corner-stitching tile planes of a design with numerous existing obstacles are generally fragmented. This phenomenon increases the computational complexity of tile propagation. This section presents two methods, namely: 1) the removal of redundant tiles and 2) the alignment of neighboring tiles, to reduce tile fragmentation.

A. Removing Redundant Tiles

The process of tile propagation was observed. Many paths are terminated because they enter a space tile that has no exit that allows further tile propagation. Such space tiles do not contribute to routing and instead increase the problem of tile fragmentation. Clearly, the redundant tiles are the one-conjunct space tiles and the 0-conjunct space tiles. The redundant tiles can be efficiently removed by examining one-conjunct and 0-conjunct space tiles within an enumeration operation

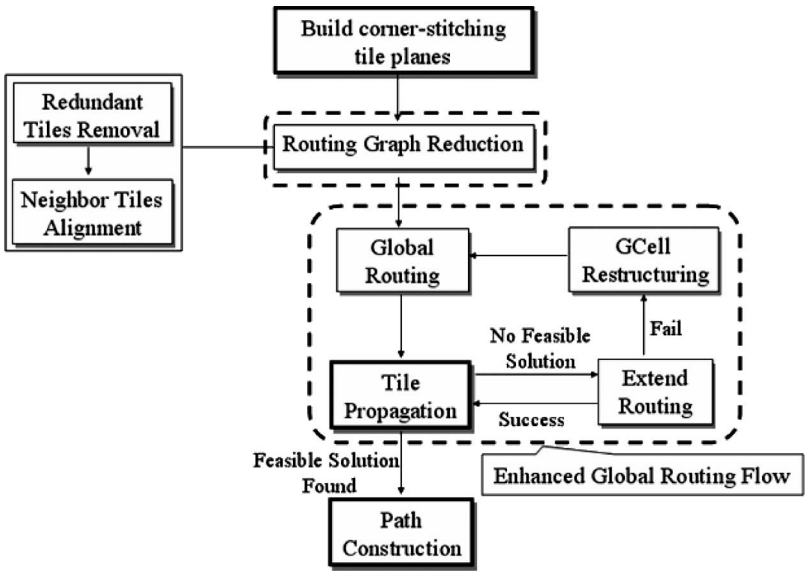


Fig. 6. Proposed new ECO routing design flow.

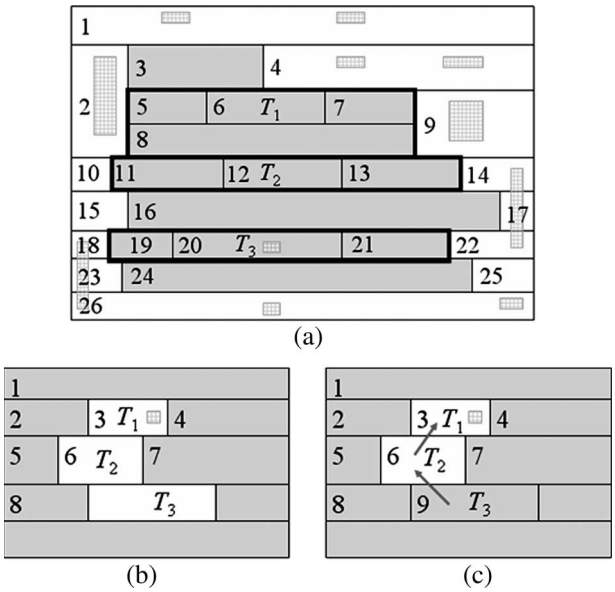


Fig. 7. (a) Example for removing redundant tiles. (b), (c) Examples of turning essential tiles into redundant tiles.

over the entire tile plane. Notably, only the redundant space tiles are changed into block tiles without merging them with their neighboring block tiles; the intermediate tile plane is not maximally horizontally or vertically stripped because merging during enumeration may disrupt the order of enumeration. After the enumeration operation has been completed, the tile plane is reconstructed to preserve the maximally horizontal or vertical strip. Fig. 7(a) illustrates the enumeration order while removing redundant tiles of the example in Fig. 5, where T_1, T_2, and T_3 are initially space tiles and, following enumeration, become block tiles. The three tiles with bolded borders in Fig. 7(a) result from removing redundant tiles and creating new tiles by tile merging; tile complexity is reduced by seven tiles.

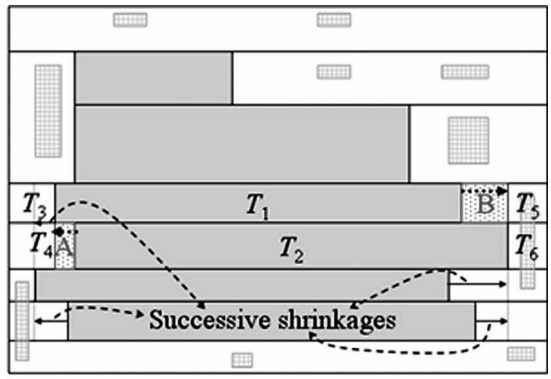
Essential tiles may become redundant after their neighboring space tiles are identified as redundant, as depicted in Fig. 7(b) and (c). Initially, tiles T_1 and T_2 are essential tiles. After T_3 is identified as redundant, T_2 also becomes redundant, as illustrated in Fig. 7(c). Moreover, T_1 becomes redundant after T_2 becomes redundant. This phenomenon is referred to as the "redundancy propagation effect." Whenever a redundant tile is found, a backward check for the redundancy propagation effect starts from the redundant tile, or the currently processed space tile, and decreases by one the number of conjunct tiles of top neighboring space tiles of the redundant tile. If none of its top neighbors become redundant, the process stops; otherwise, the process continues with the newly found redundant tile. The arrows in Fig. 7(c) indicate that the backward check process (BCP) first advances to T_2 and then to T_1.

Lemma 1: A tile-based ECO routing with redundant tile removal produces an optimal routing.

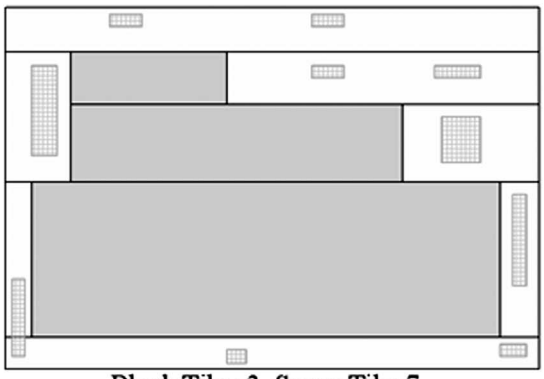
Proof: This study considers ECO routing as a point-to-point routing. Section II-B presents a brief explanation of how an optimal path can be identified with a tile-based router by using the piecewise linear function to store the source costs of all incoming paths. Redundant tile removal removes space tiles that never appear in the tile lists of found paths and does not invalidate any routing path; besides, only block tiles may merge after removing redundant tiles, and all nonredundant space tiles remain unchanged. That is, the piecewise linear function on every entry interval will remain unchanged. Consequently, a tile-based point-to-point routing can still obtain an optimal path on the simplified tile planes by removing redundant tiles. ■

B. Aligning Neighboring Tile

This section illustrates the alignment of neighboring tiles using the tile plane of the maximally horizontal strip. On a maximally horizontally stripped tile plane, the adjacent block tiles typically form ragged left and right boundaries. Ragged boundaries cause tile fragmentation. The basic idea is to shrink space tiles or enlarge block tiles to align them with the ragged border such that the neighboring block tiles can merge, as depicted in Fig. 8. In Fig. 8(a), tiles T_4 and T_5 are shrunk, so tiles T_1 and T_2 are enlarged such that they can be merged to become



(a)



(b)

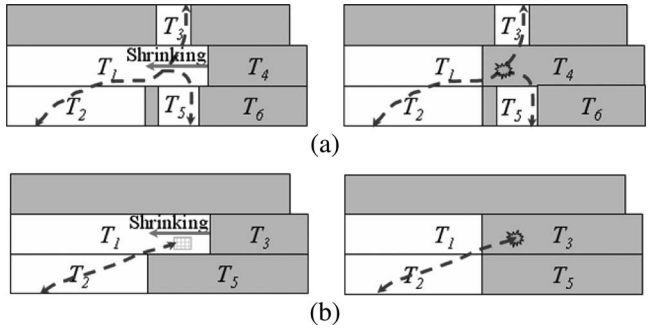
Fig. 8. (a) Example of leftward and rightward shrinkage on space tiles. (b) Final result after successive space tile shrinking.

a tile; leftward shrinkage is applied to T_4, rightward shrinkage is applied to T_5, and regions A and B are the "reduction regions" for tiles T_4 and T_5, respectively. After the shrinkage on T_4 and T_5, tile merging can be performed on three tile pairs, i.e., (T_1, T_2), (T_3, T_4), and (T_5, T_6). Eventually, a simplified tile plane is produced after successive shrinkages are completed, as shown in Fig. 8(b). Here, a rightward/leftward shrinkage on T_i is considered, and a rightward/leftward shrinking is applied to T_i's left/right border.

Notably, not all space tiles can be shrunk. If a space tile, say T_1, is to be shrunk to be merged with another tile, say T_2, then a feasible shrinking must conform to the following two rules.

Rule 1: A shrinking cannot obstruct an existing path from T_1 to its neighboring tile, say T_3, of the same layer. Fig. 9(a) depicts an illegal shrinking that does not satisfy this property. The left part of Fig. 9(a) shows two original routing paths between tiles T_1 and T_3 and tiles T_1 and T_5. However, these paths will disappear if a leftward shrinking is performed on T_1 to align it with T_2, as shown in the right part of Fig. 9(a).

Rule 2: A shrinking cannot obstruct an existing path from T_1 to its neighboring tile in the adjacent layer. This rule requires that no via region can overlap the reduction region. In Fig. 9(b), a leftward shrinking of T_1 will wipe out a routing path that is connected to an adjacent-layer space tile.



(a)

(b)

Fig. 9. (a) Example of illegal shrinking that violates Rule 1. (b) Example of illegal shrinking that violates Rule 2.

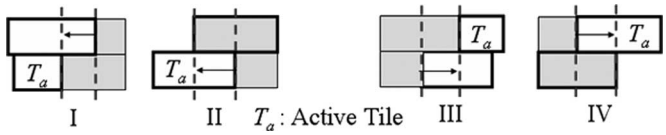


Fig. 10. Four shrinking cases during enumeration.

Before the shrinking process can be introduced, the following notation is presented:

T_a	currently processed space tile;
$l(T_i)/r(T_i)$	x coordinate of the left/right edge of tile T_i;
$N_{\mathrm{rt}}(T_i)$	rightmost top neighbor of tile T_i;
$N_{\mathrm{tr}}(T_i)$	topmost right neighbor of tile T_i;
$N_{\mathrm{lb}}(T_i)$	leftmost bottom neighbor of tile T_i;
$N_{\mathrm{bl}}(T_i)$	bottommost left neighbor of tile T_i;
$R_{\mathrm{LS}}(T_i)/R_{\mathrm{RS}}(T_i)$	leftward/rightward shrinking on T_i;
$l(R_{\mathrm{LS}}(T_i))/r(R_{\mathrm{LS}}(T_i))$	stop/start position of a leftward shrinking on T_i;
$l(R_{\mathrm{RS}}(T_i))/r(R_{\mathrm{RS}}(T_i))$	start/stop position of a rightward shrinking on T_i.

Fig. 10 lists four cases of shrinkage during enumeration to elucidate how to decide whether a shrinking on a space tile is feasible, where T_a is the active tile or the currently processed tile. Cases I and II are differentiated by $N_{\mathrm{rt}}(T_a)$. If $N_{\mathrm{rt}}(T_a)$ is a space tile, then $N_{\mathrm{rt}}(T_a)$ is the tile to be shrunk (Case I); otherwise, T_a is the tile to be shrunk (Case II). Similarly, by replacing $N_{\mathrm{rt}}(T_a)$ with $N_{\mathrm{lb}}(T_a)$, Cases III and IV can be differentiated. Only space tiles are considered because shrinking a useless space region will neither reduce routability nor produce incorrect routing results. On the contrary, shrinking block tiles may make a path across present blockages. Only Rule 1 is discussed in the following since Rule 2 is clear.

Case I: $N_{\mathrm{rt}}(T_a)$ is a space tile and the candidate to be shrunk. The goal is to perform $R_{\mathrm{LS}}(N_{\mathrm{rt}}(T_a))$, represented by a leftward arrow in Fig. 11(a), where $l(R_{\mathrm{LS}}(N_{\mathrm{rt}}(T_a))) = r(T_a)$, and $r(R_{\mathrm{LS}}(N_{\mathrm{rt}}(T_a))) = r(N_{\mathrm{rt}}(T_a))$. Rule 1 requires that a top or bottom neighboring space tile of $N_{\mathrm{rt}}(T_a)$, say T_1, cannot exist such that $l(R_{\mathrm{LS}}(N_{\mathrm{rt}}(T_a))) \leq l(T_1) < r(R_{\mathrm{LS}}(N_{\mathrm{rt}}(T_a)))$. Fig. 11(a) depicts a legal shrinking, while Fig. 11(c) shows an illegal shrinking.

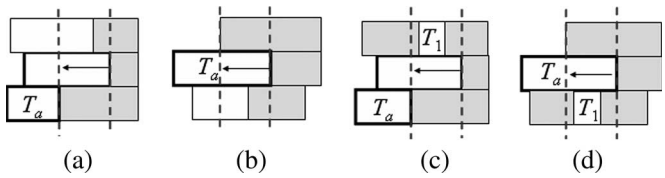


Fig. 11. (a), (b) Legal shrinking for Case I and Case II. No feasible solution is blocked if either Case I or Case II shrinkage is applied. (c), (d) Illegal shrinking for Case I and Case II. Feasible routing between T_a and T_1 is blocked if either Case I or Case II shrinkage is applied.

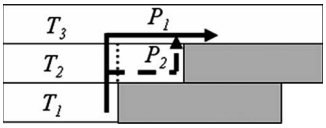


Fig. 12. There are two possible routing paths in the beginning. P_2 will disappear if a leftward shrinkage is applied to T_2.

Case II: $N_{\rm rt}(T_a)$ is a block tile, and T_a is the candidate to be shrunk. The goal is to perform $R_{\rm LS}(T_a)$, represented by a leftward arrow in Fig. 11(b), where $l(R_{\rm LS}(T_a))=l(N_{\rm rt}(T_a))$, and $r(R_{\rm LS}(T_a))=r(T_a)$. Rule 1 requires that a bottom neighbor space tile of T_a, say T_1, cannot exists such that $l(R_{\rm LS}(T_a)) \leq l(T_1) < r(R_{\rm LS}(T_a))$. Fig. 11(b) depicts a legal shrinking, while Fig. 11(d) shows an illegal shrinking.

Case III: $N_{\rm lb}(T_a)$ is a space tile and the candidate to be shrunk. The goal is to perform $R_{\rm RS}(N_{\rm lb}(T_a))$, where $l(R_{\rm RS}(N_{\rm lb}(T_a))) = l(N_{\rm lb}(T_a))$, and $r(R_{\rm RS}(N_{\rm lb}(T_a))) = l(T_a)$. Rule 1 dictates that a top or bottom neighboring space tile of $N_{\rm lb}(T_a)$, say T_1, cannot exist such that $l(R_{\rm RS}(N_{\rm lb}(T_a))) \times (r(T_1) \leq r(R_{\rm RS}(N_{\rm lb}(T_a)))$.

Case IV: $N_{\rm lb}(T_a)$ is a block tile, and T_a is the candidate to be shrunk. The goal is to perform $R_{\rm RS}(T_a)$, where $l(R_{\rm RS}(T_a)) = l(T_a)$, and $r(R_{\rm RS}(T_a)) = r(N_{\rm lb}(T_a))$. Rule 1 dictates that a top neighbor space tile of T_a, say T_1, cannot exist such that $l(R_{\rm RS}(T_a))(r(T_1) \leq r(R_{\rm RS}(T_a))$.

Tiles are only shrunk during enumeration to maintain the correct enumeration order. After the enumeration is completed, the tile plane is reconstructed to preserve the maximum horizontal strip. Neighboring tile alignment may change the jogging position of a wire and thus increases its neighboring parallel wire length such that the router obtains a different routing result for a crosstalk-driven routing with a parallel wire length constraint.

The following considers the optimality of the tile-based ECO routing with neighboring tile alignment. The conformation of the two feasible shrinking rules preserves the routability but may sacrifice some routing paths. Fig. 12 shows two possible routing paths P_1 and P_2, where P_2 will be lost after a leftward shrinking on the space tile T_2. The difference between P_1 and P_2 lies in the position of the horizontal segment between $l(R_{\rm LS}(T_2))$ and $r(R_{\rm LS}(T_2))$. If tiles T_2 and T_3 have different horizontal routing costs, then P_1 and P_2 will have different costs, and P_1 may not be the optimal path. On the contrary, P_1 is superior to P_2 if each routing layer has only one cost for the whole tile plane in a routing direction and a corner induces extra penalties; therefore, sacrificing P_2 will not influence the routing optimality.

Lemma 2: A tile-based ECO routing with neighboring tile alignment produces an optimal routing if each routing layer has only one cost in a routing direction.

Theorem 1: An improved tile-based ECO routing using RGR can find an optimal path if each routing layer has only one cost in a routing direction.

C. Complexity Analysis

Before analyzing the time complexity of RGR, the following notations are introduced:

$N_{\rm av}$ average number of tiles on a tile plane;
$NA_{\rm av}$ average number of adjacent-layer tiles that are enclosed in or intersect with a space tile;
$N_{\rm tb}$ average number of top and bottom neighbors of a space tile.

Redundant tiles can be removed by an enumeration operation followed by successive tile merging. The enumeration identifies and marks the redundant tile, which involves conjunct tile identification for the active tile and BCP for the redundancy propagation effect. The time complexities for examining the conjunct tiles of the active tile on the same and adjacent layers are $O(N_{\rm tb})$ and $O(\sqrt{N_{\rm av}} + NA_{\rm av})$, respectively. The term $\sqrt{N_{\rm av}}$ is denoted as the time complexity of locating the tile intersecting with T_a on the adjacent layer, which is a point-finding operation. The average number of visited tiles for a point-finding operation on a tile plane is in the order of $\sqrt{N_{\rm av}}$, which shows the average number of tiles in a row or a column for a tile plane containing $N_{\rm av}$ tiles. The BCP can be treated as a k-way point-finding operation for finding k tiles with neighboring space tiles that are not made redundant by the redundancy propagation effect. Each way visits an average of $\sqrt{N_{\rm av}}$ tiles, and each tile has to decrease by one the number of conjunct tiles of its top neighbors; therefore, the time complexity for BCP is $O(k \times \sqrt{N_{\rm av}} \times N_{\rm tb})$. In fact, k is zero or very small in most cases. Consequently, the time complexity of redundant tiles removal is $O(N_{\rm av} \times (N_{\rm tb} + \sqrt{N_{\rm av}} + NA_{\rm av} + \sqrt{N_{\rm av}} \times N_{\rm tb}))$. $N_{\rm tb}$ and $NA_{\rm av}$ are very small as compared to $N_{\rm av}$. Therefore, $N_{\rm tb}$ and $NA_{\rm av}$ are dominated by $N_{\rm av}$ and can be discarded. Experimental results show that, on average, $N_{\rm tb}$ ranges from 3 to 4, while $NA_{\rm av}$ can reach up to several tens of tiles. $NA_{\rm av}$ is larger because tiles on the tile plane of a horizontal routing layer are frequently wide; meanwhile, tiles on the tile plane of a vertical routing layer are slim and tall owing to the maximum vertical stripping. Finally, the time complexity can be estimated as $O(N_{\rm av} \times \sqrt{N_{\rm av}})$.

The neighboring tile alignment has to conform to two feasible shrinking rules. The time complexities for Rules 1 and 2 are $O(N_{\rm tb})$ and $O(\sqrt{N_{\rm av}} + NA_{\rm av})$, respectively. Therefore, the complexity for neighboring tile alignment is $O(N_{\rm av} \times (N_{\rm tb} + \sqrt{N_{\rm av}} + NA_{\rm av}))$, which can be simplified to $O(N_{\rm av} \times \sqrt{N_{\rm av}})$.

TABLE I
STATISTICS OF TWO CIRCUITS UNDER TEST

	# standard cell	# Rect. M2	# Rect. M3	# Rect. V12	# Rect. V23	# Rect. total	Vacancy density
C1	121856	629750	448250	256413	550219	1884632	66.1%
C2	114155	632634	399993	399852	618218	2050697	30.2%

V. EGRF

Based on routing region partitioning, the global routing graph $G(V, E)$ is defined as follows: each GCell corresponds to a node of $G(V, E)$, and an edge exists between two nodes when their related partitions are adjacent. Dijkstra's algorithm is applied in G to find the minimum-cost global path for the detailed routing of two terminals.

A. Via-Based Congestion Metric

Most likely, an ECO routing region is very congested, and numerous existing obstacles fragment the available routing regions, so finding a straight path across a GCell is difficult, and the path may be zigzag. Accordingly, the via region is more important than the wire region in helping a path pass through a congested region. Estimating the routability of the resources based on the via region model for ECO global routing differs greatly from doing so for traditional global routing, which mainly calculates the utilization ratio for the regions. The availability of via resources is estimated from the total area of the available via regions, which is calculated quickly by enumerating the space tiles on the via tile planes. The via capacity VC of a GCell is defined as

$$\mathrm{VC} = \mathrm{VA}/A \tag{1}$$

where VA is the total area of the available via region of the GCell, and A is the area of the GCell. Each vertex in G is assigned a cost c, which is inversely proportional to VC. Cost c is defined by a piecewise linear function as

$$c = \begin{cases} 1/\mathrm{VC}, & \text{if } \mathrm{VC} > t \\ m/\mathrm{VC}, & \text{if } \mathrm{VC} \leq t \end{cases} \tag{2}$$

where t is a threshold value, and m is an amplification scalar. Experimental observations reveal the predicament of obtaining a path through the GCell when $\mathrm{VC} < 0.01$, so the cost is amplified by multiplying it by a scalar. Also, each edge of G is assigned a length cost lc to indicate the net timing factor. The length cost lc is set to be $2/t$, and the total cost is to add up every visited vertex cost c and their length cost lc. Based on cost design, the global router attempts to pass through most routable regions while simultaneously conforming to the timing constraint.

B. Extended Routing and GCell Restructuring

When detailed routing cannot find a feasible path, a visited GCell through which a path cannot pass is called a "blocked GCell." Extended routing is conducted to expand the search scope by half of the width of a GCell around the blocked GCell. Since further propagation entirely out of active GCells is prohibited, and the path along such propagation is called an "idle path" stored in the "idle-path heap" of the GCell containing this idle path, the tile propagation of extended routing starts from the idle paths in the idle-path heaps of those idle GCells adjacent to the blocked GCell.

A new global routing following GCell restructuring is performed to acquire a new global path. This process may be repeated several times if an extended routing still cannot pass through a blocked GCell. The new area constraints formed by active GCells for the next tile propagation comprise the new active GCells along the new global path and the previously visited GCells. The new active GCells were idle GCells before. Some of them must be adjacent to the visited GCells and at least one of them must have a nonempty idle-path heap. These nonempty idle-path heaps are popped up to initiate the new tile propagation. Global rerouting may find an entirely new global path from the original path if that path has a minimum cost. When the active GCells are rerouted, the number of visited GCells increases. The visited GCell will be revisited if new entering paths are of lower cost than before. The worst case is to make all of the GCells active such that the tile propagation propagates over the whole tile plane. Accordingly, a feasible solution will always be found for routable designs. Additionally, even when the worst case happens, the routing speed does not substantially drop off because most of the routing regions that have already been visited are not visited again as a result of good routing resource estimation.

VI. EXPERIMENTAL RESULTS

A tile-based ECO router with the new routing design flow proposed in this paper was implemented using C++ language. The routing was undertaken using a 2.4-GHz Pentium-4 PC with 1-GB RAM using two real very large scale integration (VLSI) designs, say $C1$ and $C2$, whose statistics are listed in Table I. The ratio of vacant area to chip area is the vacancy density, which is shown at the last column in Table I. A layout with higher vacancy density is probably more routable. Although $C1$ contains more number of standard cells, more vias in $C2$ and less vacancy density make $C2$ harder to perform ECO routing.

Table II presents the number of tiles of the metal-2 layer and metal-3 layer tile planes. The results without RGR are listed at the second column, and the data in the third, fourth, and fifth columns are the reduction rates with redundant tiles removal (RTR), neighboring tiles alignment (NTA), and both approaches, respectively. RTR can be up to six times more effective than NTA in reducing routing graph. In $C1$ and $C2$, the metal-2 layer is more congested than the metal-3 layer. RTR does not have obvious differences in reduction rate for different

TABLE II
TILE REDUCTION RATES USING RTR, NTA, AND RGR

			Original # of tile	R.R. of RTR	R.R. of NTA	R.R. of RGR
C1	M2	BT	514,826	37.9%	14.4%	53.5%
		ST	533,294	43.8%	14.8%	60.0%
		BT+ST	**1,048,120**	**40.9%**	**14.6%**	**56.8%**
	M3	BT	331,990	39.5%	7.8%)	45.8%
		ST	327,595	43.4%	6.6%)	49.3%
		BT+ST	**659,585**	**41.4%**	**7.2%**	**47.5%**
	M23	**BT+ST**	**1,707,705**	**41.1%**	**11.7%**	**53.2%**
C2	M2	BT	1,067,179	49.4%	11.1%	55.9%
		ST	973,528	54.0%	9.9%	60.9%
		BT+ST	**2,040,707**	**51.6%**	**10.6%**	**58.3%**
	M3	BT	591,120	48.4%	4.6%	51.1%
		ST	541,706	57.1%	3.5%	59.7%
		BT+ST	**1,132,826**	**52.6%**	**4.1%**	**55.2%**
	M23	**BT+ST**	**3,173,533**	**51.9%**	**8.3%**	**57.2%**

- BT: Block Tile; ST: Space Tile; BT + ST: block tile and space tile; M23: metal 2 and metal 3
- R.R. (Reduction rate): (# of tiles without reduction - # of tile after reduction) / (# of tiles without reduction)

TABLE III
PREPROCESSING TIME IN SECONDS

	CS Plane Construction / T_{cs}	RGR(RTR,NTA) / $T_{rgr}(T_{rtr},T_{nta})$
C1	8.244	1.875(0.859,1.016)
C2	21.656	5.889(3.71,2.179)

degrees of congestion; however, NTA has roughly two times better reduction rate for more congested tile planes. The number of space, block, and both space and block tiles are all reduced by over 50% for both cases. Notably, less space tiles result in less tile propagation, and less total number of tiles favors rapid query during tile propagation. Based on these observations, tile propagation time can be expected to be reduced by around 50%. Table III lists the preprocessing time before routing. The second and third columns present the times taken to construct corner-stitching planes and for RGR ($T_{\rm rgr}$), where the corner-stitching planes are necessary for both the conventional approach and the proposed design flow, and RGR ($T_{\rm rgr}$) is the additional computation time required. The individual time required for redundant tile removal ($T_{\rm rtr}$) and neighboring tile alignment ($T_{\rm nta}$) are listed following $T_{\rm rgr}$.

Four and seven point-to-point routings were performed on $C1$ and $C2$, respectively, using three methods, namely: 1) pure tile-based router; 2) tile-based router with RGR; and 3) tile-based router with EGRF, to yield the tile propagation time in seconds (RT), the length of the wire in micrometers (WL), and the number of vias. Tables IV and VI compare the results without reduction to those with RGR and with EGRF. The second column in Table IV concerns pure tile propagation, and the third column concerns the application of RGR. The point-to-point routings performed on the two cases include short-distance routing, long-distance routing, and hard-to-route routing, which requires numerous layer switches and detours. The RGR reduces the tile propagation time by approximately 50% for all point-to-point routings, as shown in columns RT and T1. This reduction demonstrates that the tile-based router achieves the same profit from TPS for routings with different degrees of difficulty. The speed of the ECO router can be improved from 29% to 44%, as shown in column T2. Since the ECO routing time includes the preprocessing time, RGR appears not to gain significant improvement if the required tile propagation time is comparable to the preprocessing time. On the other hand, the optimal path is preserved while achieving runtime reduction of almost half of the tile propagation time, as shown in columns WL and Via. Experiments also demonstrate that a layout with lower vacancy density has higher reduction rate and runtime speedup. This phenomenon is not surprising since a congested design is probably highly fragmented and thus is also more likely to be reduced.

To compare the tile-based router with RGR with a commercial grid-based routing tool, another test case with RTL codes, consisting of 9670 cells and 10 479 nets, is used to perform three point-to-point routings. Both routers are run on a SUN Blade 2000 workstation with 1-GB RAM. The source codes of the proposed router are compiled using g++ 3.4.2v with a $-$O2 optimization option that can achieve nearly three times increase in speed compared with that achieved without optimization. The experiments are conducted using the following processes.

TABLE IV
RESULTS ON RUNTIME FOR TILE PROPAGATION, WIRE LENGTH, NUMBER OF VIAS, AND RUNTIME REDUCTION RATE

		Pure Tile-based Router			Apply RGR			*T1*	*T2*
Circ.	Test	RT (T_a)	WL(W_a)	#Via(V_a)	RT (T_b)	WL (W_b)	#Via (V_b)	(%)	(%)
C1	R1	**11.7**	426488	92	**6.4**	426487	92	**45.1**	**29.1**
	R2	**14.2**	371223	126	**8.1**	371223	126	**42.8**	**29.6**
	R3	**20.2**	486011	154	**11.7**	486018	154	**42.1**	**32.8**
	R4	**64.4**	595268	6	**38.3**	595268	6	**40.6**	**37.7**
Average		**27.6**			**16.1**			**42.6**	**32.3**
C2	R1	**128.3**	12035.07	480	**65.8**	12035.08	480	**48.7**	**44.2**
	R2	**82.6**	11553.93	478	**41.8**	11553.96	478	**49.5**	**42.3**
	R3	**73.9**	11399.45	394	**38.3**	11399.48	394	**48.2**	**39.4**
	R4	**65.3**	9667.94	398	**33.1**	9667.97	398	**49.3**	**40.3**
	R5	**52.2**	11194.86	508	**26.5**	11194.92	508	**49.7**	**38.0**
	R6	**121.5**	12618.18	498	**64.4**	12618.21	498	**47.0**	**42.2**
	R7	**147.1**	11732.99	502	**75.6**	11733.03	502	**48.7**	**44.7**
Average		**95.8**			**49.4**			**48.7**	**41.6**

- *T1* : $(T_a\text{-}T_b)/T_a$, *T2* : $(T_a\text{-}(T_b+T_{rgr}))/T_a$.

TABLE V
COMPARISONS OF THREE POINT-TO-POINT ROUTINGS FOR THE TILE-BASED ROUTER WITH RGR AND A COMMERCIAL GRID-BASED ROUTER ON ROUTING QUALITY AND TIME

	R1			R2			R3		
	WL	Via	Time (s)	WL	Via	Time (s)	WL	Via	Time (s)
Ours	417.15	15	3.2	523.56	22	3.04	464.19	20	2.92
tool	468.94	26	6	589.84	30	5	497.04	29	7

The design under test is first routed using a commercial tool (59 min) based on 0.18-μm CMOS technology using three metal layers (M2–M4). Then, for each test, one net is selected, and most of its wires are removed. The incomplete routing is completed by the commercial tool. The three point-to-point routings are also conducted by the proposed router on the same machine. The total tile reduction rate is 48.86%, and the times required for tile plane construction and RGR for R1, R2, and R3 are 2.21, 2.28, and 2.24 s, respectively. Table V shows the experimental results for the three ECO routings, where the time used by the proposed router includes corner-stitching tile plane construction, RGR, and routing. The experimental results reveal that the tile-based router with RGR achieves routing performance and quality superior to that achieved by the commercial grid-based router. Actually, the traditional tile-based router also outperforms the commercial grid-based router in routing performance and quality.

Table VI shows the results of the new ECO routing design flow, and the two right items in the second column reveal how many times the extended routing and GCell restructuring and rerouting are performed. Four routings are finished without extended routing and GCell restructuring. On average, the rate of increase in wire length for both cases is approximately 11%, and the rate of increase in via number for the two cases is 32.8% and -5.5%, respectively. Experiments demonstrate that the routing resource estimation model using via resource produces better routing quality for $C2$. The routing quality of using a congestion metric is affected by the vacancy density of a design and the via rule in use. The impact of different via design rules on routing resource estimation will be discussed in further details in the following section. Table VI shows two important results. First, as shown in column T3, the routing time is 78%–86% lower than that required for the pure tile-based router for $C1$ and $C2$, whose impact turns to be obvious as the design size increases. Second, the new design flow requires little average iteration (less than 1) of global rerouting to obtain the first routing result. This demonstrates that the proposed routing design flow can rapidly converge provided the routing is routable. The wire is much longer for R5 as the start terminal is located in a very dense GCell, whose neighboring GCells close to the target also tend to be dense and are regarded as unroutable such that a detour global path instead of a nondetour global path is found.

VII. DISCUSSION

The improvement in ECO routing speed is encouraged; however, a few problems need to be considered for further refinement and application.

Point-to-point routing is a key technique for detailed routers. The proposed method of reducing the routing graph to accelerate tile-based point-to-point routing can also be applied to accelerate tile-based multiple-net routing. Initially, most of the routing area is empty except for the area occupied by

TABLE VI
RESULTS ON TILE PROPAGATION RUNTIME, WIRE LENGTH, NUMBER OF VIAS, AND RUNTIME REDUCTION RATE

		New ECO routing design flow					*T3*(%)	*W*(%)	*V*(%)
Cir.	Test	RT (T_c)	WL (W_c)	#Vias (V_c)	#ER	#GCRS			
C1	R1	**1.7**	455485	146	2	0	**69.2**	6.80	58.70
	R2	**2.1**	465713	196	2	1	**72.1**	25.45	55.56
	R3	**2.4**	553910	180	3	0	**79.0**	13.97	16.88
	R4	**2.9**	595268	6	0	0	**92.5**	0.00	0.00
Average		**2.3**			**1.8**	**0.3**	**78.2**	**11.6**	**32.8**
C2	R1	**8.6**	12497.55	184	0	0	**88.7**	3.8	-61.6
	R2	**4.6**	12607.58	530	1	0	**87.2**	9.1	10.8
	R3	**4.3**	12173.36	536	0	0	**86.2**	6.7	36.0
	R4	**4.6**	10687.03	416	4	4	**83.9**	10.5	4.5
	R5	**5.3**	14437.81	588	4	2	**78.5**	28.9	15.7
	R6	**5.6**	13533.45	516	0	0	**90.5**	7.2	3.6
	R7	**6.9**	13097.15	262	1	0	**91.2**	11.6	-47.8
Average		**5.7**			**1.4**	**0.9**	**86.6**	**11.1**	**-5.5**

- The global routing partitions C1 and C2 into 30x45 and 28x28 GCells, respectively, and each GCell' s width is about 122 pitches.

ER : Extended Routing, *GCRS*: GCell restructuring and re-routing. *T3* : $(T_a-(T_c+T_{rgt}))/T_a$, *W*: $(W_c-W_a)/W_a$, *V*: $(V_c-V_a)/V_a$

the prerouted nets and placed blocks. The tile planes become more fragmented as more routing paths are inserted into the related tile planes. Single-net routing with n terminals can be achieved by first finding a minimum-cost Steiner tree and then decomposing the tree routing into $(n-1)$ point-to-point routings. Performing RGR when the tile fragmentation in the routing area has increased to a certain level can diminish tile fragmentation and then performance degradation. The level of tile fragmentation can be measured by the ratio of the number of tiles to the routing area. Tile plane enumeration can be used to measure the level of tile fragmentation after a fixed number of net routing. The time required for an enumeration operation is considerably smaller than detailed net routing, so the time for detecting whether a tile plane is overfragmented, and for the RGR, is also quite small as compared to the time saving achieved by the successive net routings following RGR.

This paper considers a congestion metric using via resource for ECO routing. The following discusses the wire- and via-resource congestion metrics. Bottom metals and vias in Fig. 13(a) and (b) exhibit conventional and modern via design rules, say, VA_C and VA_M, respectively. S_{m2m} denotes the space rule between two adjacent metals, S_{v2v} represents the space rule between two vias, E_{m2v} is the enclosure rule of metal over via, and W_v denotes the via width. In VA_C, the metal enclosing the via has a larger width than the metal wire width, and the metal enclosing values over the via in the x and y directions are the same. Additionally, the space rule between two metals enclosing two vias generally overrules the space rule between two vias, that is, a legal space between two metals enclosing two vias guarantees a legal space between two enclosed vias, and the inequality $2 \times E_{m2v} + S_{m2m} \geq S_{v2v}$ always holds. In VA_M, the metal enclosing values differ in the x and y directions. For instance, a horizontal (vertical) metal has a lower vertical (horizontal) enclosing value than its horizontal (vertical) enclosing value, while the metal height

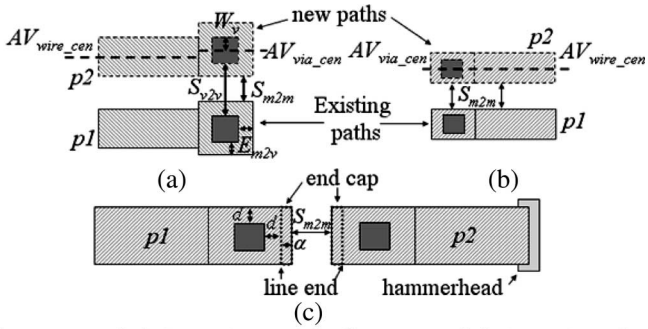


Fig. 13. (a) Paths $p1$ and $p2$ apply the conventional via design rule. Path $p2$ is placed as close to $p1$ as possible. (b) Paths $p1$ and $p2$ apply the modern via design rule. Path $p2$ is placed as close to $p1$ as possible. (c) End caps are attached in the line end to act as a hammerhead for line-end shortening correction.

(width) is the same as the metal wire width. Fig. 13(c) shows the details of a modern via. The metal vertical and horizontal enclosing values over the via are d and $d+\alpha$, respectively. The region of width α, outlined by a dotted border, is called an end cap and is inserted in the line end for optical proximity correction (OPC), which counteracts lithography distortions resulting from subwavelength manufacturing. OPC can apply hammerheads for line-end shortening correction, as shown in the right line end of path $p2$ in Fig. 13(c). If two line ends are face-to-face adjacent with a minimum space, as shown in Fig. 13(c), no room exists for hammerhead insertion. The end cap can then prevent OPC failure during routing. VA_M has become popular as the semiconductor technology advances to 0.18 μm and below.

The bottom metals and vias outlined by solid lines in Fig. 13(a) and (b) are originally placed paths, and the top metals and vias indicated by dotted lines are the new and legal metals and vias, which are placed as close as possible to the existing path. $\mathrm{AV}_{\mathrm{wire\_cen}}$ denotes the bottommost center line of a new legal metal wire, and $\mathrm{AV}_{\mathrm{via\_cen}}$ represents the bottommost center line of a new legal via. For VA_C, $\mathrm{AV}_{\mathrm{wire\_cen}}$ is clearly closer to the existing path than $\mathrm{AV}_{\mathrm{via\_cen}}$, as shown in Fig. 13(a), demonstrating that via resources can be considered a lower bound of wire resources. For two GCells in a high-vacancy density design, the GCell with higher via resources may contain less wire resources since the GCell with less via resources may contain more available wiring regions, most of which cannot afford enough space for vias. This phenomenon is probably more frequent in high-vacancy density designs because more available wiring regions induce a higher possibility of the appearance of wire-only regions. The misestimate of GCell routability resulting from only using via resources in high-vacancy density designs directs detailed routers to pass through the GCell that consumes more vias. This is why the high-vacancy density design $C1$ consumes over 50% more vias than the optimal paths in two tests. On the other hand, for low-vacancy designs, via resources are more important than wire resources in providing space for a zigzag path. Consequently, congestion metric considering dynamic weighting on wire resources and via resources based on the vacancy density of the routed design can further improve the quality of the proposed ECO routing for the designs using the conventional via design rule. Most real VLSI applications have low-vacancy density for economic reasons, namely to achieve more dies per wafer. For VA_M, $\mathrm{AV}_{\mathrm{wire\_cen}}$ and $\mathrm{AV}_{\mathrm{via\_cen}}$ are located equally horizontally as a result of having equal metal enclosure width and metal wire width, as shown in Fig. 13(b). Therefore, the via resource as well as wire resource congestion metrics can accurately estimate the routability of a GCell.

VIII. Conclusion

Tile-based routers are widely employed for gridless routing; however, no significant performance improvement has been reported since it was first proposed. In this paper, a novel RGR that nearly doubles the speed of tile propagation without sacrificing routing quality is first proposed. The RGR reduces the node complexity of the routing graph and tile fragmentation by removing redundant tiles and aligning neighboring tiles in an attempt to merge adjacent block tiles. The proposed tile-based router with RGR achieves superior routing performance and routing quality for three ECO routings compared with a commercial place and route tool. Furthermore, a new ECO routing design flow with RGR and enhanced ECO global routing flow via extended routing and GCell restructuring that prevents routing failure in a routable design is proposed. Compared with the traditional tile-based router, the runtime can be reduced by around 86%.

Acknowledgment

The authors would like to thank the anonymous reviewers whose valuable suggestions help shape and clarify this paper.

References

[1] J. Cong, L. He, C.-K. Koh, and P. Madden, "Performance optimization of VLSI interconnect layout," *Integr. VLSI J.*, vol. 21, no. 1/2, pp. 1–94, Nov. 1996.

[2] T. Ohtsuki, "Gridless routers—New wire routing algorithms based on computational geometry," in *Proc. Int. Conf. Circuits and Syst.*, May 1985, pp. 802–809.

[3] K. L. Clarkson, S. Kapoor, and P. M. Vaidya, "Rectilinear shortest paths through polygonal obstacles in O(n(log n)) time," in *Proc. 3rd Annu. Symp. Comput. Geometry*, 1987, pp. 251–257.

[4] Y. Wu, P. Widmayer, M. Schlag, and C. Wong, "Rectilinear shortest paths and minimum spanning trees in the presence of rectilinear obstacles," *IEEE Trans. Comput.*, vol. C-36, no. 1, pp. 321–331, Mar. 1987.

[5] S. Zheng, J. S. Lim, and S. Iyengar, "Finding obstacle-avoiding shortest paths using implicit connection graphs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 15, no. 1, pp. 103–110, Jan. 1996.

[6] J. Cong, J. Fang, and K. Khoo, "An implicit connection graph maze routing algorithm for ECO routing," in *Proc. Int. Conf. Comput.-Aided Des.*, Nov. 1999, pp. 163–167.

[7] ——, "DUNE: A multilayer gridless routing system with wire planning," in *Proc. Int. Symp. Phys. Des.*, Apr. 2000, pp. 12–18.

[8] ——, "DUNE—A multilayer gridless routing system," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 5, pp. 633–647, May 2001.

[9] M. Sato, J. Sakanaka, and T. Ohtsuki, "A fast line-search method based on a tile plane," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1987, pp. 588–591.

[10] A. Margarino, A. Romano, A. De Gloria, F. Curatelli, and P. Antognetti, "A tile-expansion router," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. CAD-6, no. 4, pp. 507–517, Jul. 1987.

[11] R. E. Lunow, "A channelless, multilayer router," in *Proc. 25th ACM/IEEE Des. Autom. Conf.*, 1988, pp. 667–671.

[12] L. C. Liu, H.-P. Tseng, and C. Sechen, "Chip-level area routing," in *Proc. Int. Symp. Phys. Des.*, Apr. 1998, pp. 197–204.

[13] C. Tsai, S. Chen, and W. Feng, "An H-V alternating router," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 11, no. 8, pp. 976–991, Aug. 1992.

[14] J. Dion and L. M. Monier, "Contour: A tile-based gridless router," Western Res. Lab., Palo Alto, CA, Res. Rep. 95/3.

[15] Z. Xing and R. Kaog, "Shortest path search using tiles and piecewise linear cost propagation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 2, pp. 145–158, Feb. 2002.

[16] J. K. Ousterhout, "Corner stitching: A data-structuring technique for VLSI layout tools," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. CAD-3, no. 1, pp. 87–100, Jan. 1984.

[17] J. Cong, J. Fang, and Y. Zhang, "Multilevel approach to full-chip gridless routing," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, San Jose, CA, Nov. 2001, pp. 396–403.

[18] J. Cong, M. Xie, and Y. Zhang, "An enhanced multilevel routing system," in *Proc. IEEE Int. Conf. Comput.-Aided Design*, San Jose, CA, Nov. 2002, pp. 51–58.

[19] Y.-W. Chang and S.-P. Lin, "MR: A new framework for multilevel full-chip routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 5, pp. 793–800, May 2004.

[20] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D. T. Lee, "A fast crosstalk- and performance-driven multilevel routing system," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, San Jose, CA, Nov. 2003, pp. 382–387.

Yih-Lang Li (M'01) received the B.S. degree in nuclear engineering and the M.S. and Ph.D. degrees in computer science from the National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1987, 1990, and 1996, respectively.

He was a Software Engineer and an Associate Manager with Springsoft Corporation from 1995 to 1996 and 1998 to 2003, where he was involved in the development of verification and synthesis tools for custom-based layout. In February 2003, he joined the faculty of the Department of Computer Science, National Chiao Tung University (NCTU), Hsinchu, where he is currently an Assistant Professor. His research interests include physical synthesis, parallel architecture, and very large scale integration (VLSI) testing.

Jin-Yih Li received the B.S. and M.S. degrees in computer and information science from the National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 2002 and 2004, respectively.

Since 2005, he has been with the Design Automation Department, Taiwan Semiconductor Manufacturing Company Ltd., Hsinchu.

Wen-Bin Chen received the B.S. degree in electrical engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C., in 2002 and the M.S. degree in computer and information science from the National Chiao Tung University, Hsinchu, Taiwan, in 2005.

He is currently with Global Unichip Corporation, Hsinchu, where he works on the physical synthesis of system-on-chip (SoC) design.