國 立 交 通 大 學

電機學院 IC 設計產業研發碩士班

碩 士 論 文

考慮矩形/非矩形障礙物的 X 結構多層繞線器

Multi-layer Rectangular/Non-Rectangular Obstacles-Avoiding X-Architecture Router

研 究 生：石佳正

指導教授：陳宏明 博士

林佑政 博士

中 華 民 國 九 十 六 年 十 一 月

考慮矩形/非矩形障礙物的 X 結構多層繞線器

Multi-layer Rectangular/Non-Rectangular Obstacles-Avoiding X-Architecture Router

研究生：石佳正　　Student: Chia-Cheng Shih

指導教授：陳宏明 博士　　Advisor: Prof. Hung-Ming Chen

林佑政 博士　　Prof. Yu-Chung Lin

國 立 交 通 大 學

電機學院 IC 設計產業研發碩士班

碩士論文

A Thesis

Submitted to College of Electrical and Computer Engineering

National Chiao Tung University

in Partial Fulfillment of Requirements

for the Degree of

Master

in

Industrial Technology R & D Master Program on

IC Design

November 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年十一月

考慮矩形/非矩形障礙物的 X 結構多層繞線器

研究生：石佳正　　指導教授：陳宏明 博士
林佑政 博士

國立交通大學電機學院產業研發碩士班

摘要

隨著近年來深次微米技術的進步，許多之前被忽略的繞線問題一一出現，繞線的處理也就越來越重要。傳統繞線方式多以曼哈頓架構，相對於X 架構繞線方式會有較長的連線長度以及比較差的連線延遲，以多層繞線來說，一個打穿孔(via)的繞線延遲比一般曼哈頓繞線延遲來得嚴重，而且已經繞好的線及macro cell都可視為障礙物，因此在多層繞線階段我們必須將障礙物以及45度障礙物還有打穿孔的數目皆納入考量。

本論文提出在矩形與非矩形障礙物的條件下，分區以X結構繞線器並且使用極少的打穿孔進行多層繞線，主要的目的是要得到一個最小化繞線總長度以及最小化最大延遲的繞線結果。而為了能夠處理未來可能產生的ECO(Engineer Change Order)問題我們保留了之前繞線所建構的資訊，在原始設定作變更的時候能夠迅速的找出需要作變更的相關路徑並且快速得到重新繞線的結果。與傳統的ECO全部重新繞線的處理方式比較，能夠節省大量的時間跟資源。

Multi-layer Rectangular/Non-Rectangular Obstacles-Avoiding X-Architecture Router

Student: Chia-Cheng Shih

Advisor: Prof. Hung-Ming Chen
Prof. Yu-Chung Lin

Industrial Technology R & D Master Program of
Electrical and Computer Engineering College
National Chiao Tung University

Abstract

Due to emerging DSM effects, routing has been a very important topic in current design flow. Currently there are three issues. One, the traditional Manhattan routing has longer length and larger delay than X-architecture routing. Second, in multilayer routing, the delay of one via is much larger than the delay of Manhattan routing. Third, since a routed segment and macro cell should be considered as obstacles, we must consider the rectangle and non-rectangle obstacles, and consider the number of vias as well.

In this thesis, under the conditions of rectangle obstacles and non-rectangle obstacles, we use fewer vias and X-Architecture router by region to construct the multilayer routing trees. The main purpose is to obtain a routing tree of minimal wirelength and minimal delay. On the other hand, in order to solve one of the ECO (Engineering Change Order) problem, we keep the previous routing information to find the path, which needs to be modified, and thus the rerouting can be built quickly and efficiently. Compared with the traditional ECO method that reroutes all the nets again, we can save a lot of run time and resources.

誌謝

首先要感謝的人，是我的指導教授陳宏明老師和林佑政老師，沒有兩位老師不吝的指導，在我迷惑的時候替我指引我的目標跟方向，我是不可能有能力完成這篇論文的。

此外，特別要感謝的是台灣大學以及中原大學的研究團隊提供程式碼給我作為參考，也非常感謝中原大學的黃惠瑜同學和張紓萍同學在我有疑問的時候仍再百忙之中抽出時間並且不厭其煩的與我互動討論，讓我作研究的這段時間能夠更得心應手。

接下來要感謝的是實驗室同學兩年來一起的互相成長,使我這段時間獲益良多，最後要感謝的是我的家人，他們對我的支持、鼓勵是我研究過程中最大的動力。

石佳正

民國九十六年十一月 於新竹

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, scaling down device dimension or utilizing novel crystallization technologies provide the opportunity of applying much more devices to integrated circuit fabrication. However, it also brings a lot of physical characteristics which were neglected in the past. The problems contain wire congestion, delay, crosstalk, etc. Therefore, the researches about routing have drawn much attention in VLSI Physical Design.

Over the past few years, a considerable number of studies have been made on routing and these studies of routing could be mainly divided into three different aspects. First, the majority of the researches mainly gave priority to shorten delay time as a result of minimizing the total wirelength [7] [18] [6] [2]. Second, as the foregoing obstacles would block a routing fabrication, there are two ways to solve these obstacles at present. The first way is to obtain the routing result without considering the obstacles, and later to adjust the line among the obstacles [19]. Another way is to construct the routing tree considering the obstacles. It is called obstacle-avoiding rectilinear steiner minimal tree (OARSMT). There are many studies of OARSMT in the past, and the researches on single layer (2-D) have been drawn a lot of attention, such as [10] [13] [18] [5] [3]. Third, although a large number of studies have been made on 2-D routing tree, little is known about multi-layer routing tree. So far the

technology of IC design has been able to meet the requirements for SIP(System in Package) or SOP(System on Package) [12] [15] [17] [14] [16]. However, the technology of multi-layer EDA is not mature enough. The problem of large wirelength still exists, and therefore it is necessary to improve the multi-layer routing.

Since the delay time of a via is larger than that of Manhattan style, and the routing of X-Architecture results in better wirelength, our algorithm uses less via numbers, and completes the routing by X-Architecture. First, appropriate location of via each layer must be found, and we divide the chip to four regions. Second, the routing of each region is completed by adopting Delaunay Triangulation (DT) respectively. The spanning graph is applied to avoid the condition that the connection throughout the obstacle. Third, the shortest path is chosen from all connecting path, and the routing of the shortest path is completed by X-Architecture. The routings of all regions are constructed step by step until all of them are completed.

The routing strategy of avoiding non-rectangle obstacle is considerable additionally. It is discussed respectively that whether the pins are located on the routable region apart from the non-rectangle obstacle or not. As pin is inside the fictitious rectangle, we construct the spanning graph additionally. In order to deal with one of the ECO issues, we reserve the information constructed during routing process. Therefore, the rerouting result can be obtained quickly during ECO issues.

This thesis is divided into 4 chapters. After a brief introduction given in Chapter 1, we introduce the previous works and the problem description in Chapter 2. In Chapter 3, we give an explanation of our algorithm. After that, we show the experimental results in Chapter 4. Finally, we summarize our conclusion and future work in Chapter 5.

Chapter 2

Preliminary

In this chapter, we introduce some previous works and our problem description.

2.1 Previous Works

Because the technology of process progresses rapidly, there are more and more components can be put in the chip of same size. It will increase the complexity of routing problem and cause a lot of physical characteristics which were neglected in the past, hence routing becomes one of the key steps in physical design. There is a considerable number of studies have been made on routing, they can be classified into three main groups, we will introduce them separately in this section. To begin with, we illustrate the relevant research of the wirelength minimization. Then, we illustrate the relevant research of obstacle-avoiding routing. Finally, we introduce relevant research of multi-layer routing.

2.1.1 Researches on Wirelength Minimization in Routing

2.1.1.1 A Steiner Tree Heuristic [9]

[9] proposed 1-Steiner algorithms to solve the minimum rectilinear Steiner tree(MRST) question. The basic conception of the algorithm was to search Hanan Steiner candidate points sequentially. As 1-Steiner algorithms inserted one Steiner point each

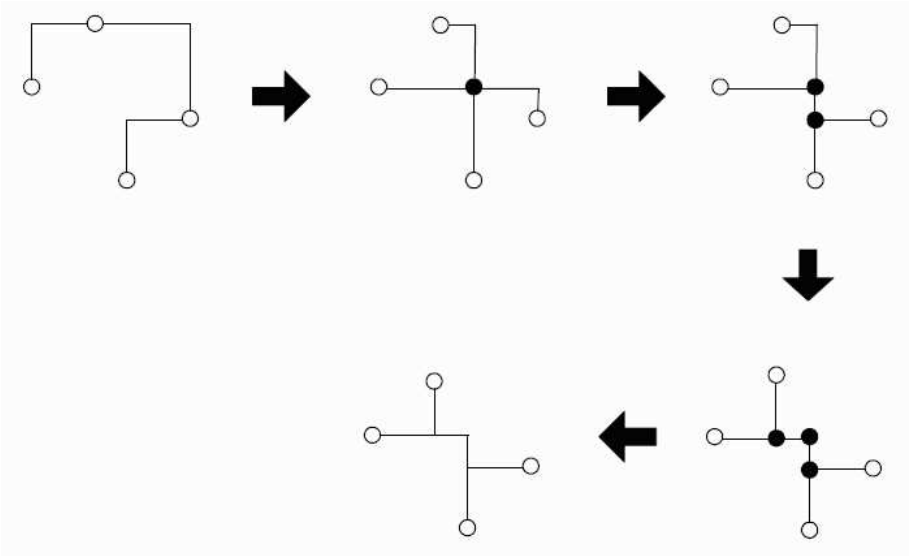


Figure 2.1: the procedure of 1-Steiner algorithms

time, it would update the total wirelength. Besides, it would bring a reduced wirelength by inserting Steiner point. And the algorithm would keep inserting the Steiner point until total wirelength was unable to decrease anymore. The example of carrying out 4 nodes algorithm is shown in Figure 2.1.

Compared with some previous MST-based methods, the total wirelength would be averagely improved about 10 percent. However, the time complexity of 1-Steiner algorithms is $O(n^2)$. This work addressed that the ratio of MRST, which was constructed by 1-Steiner algorithms, to the other MST was 2/3.

2.1.1.2 The X-Based Architecture Routing [6]

As technology advances into the nanometer territory, the interconnect delay has become a first-order effect on chip performance. To handle this effect, the X-architecture has been proposed for high-performance integrated circuits.In [6], the authors presented the first multilevel framework for full-chip routing using the X-architecture, such as in Figure 2.2. Since the optimal routing solution for each three-terminal net can be found easily, the authors used the delaunay triangulation

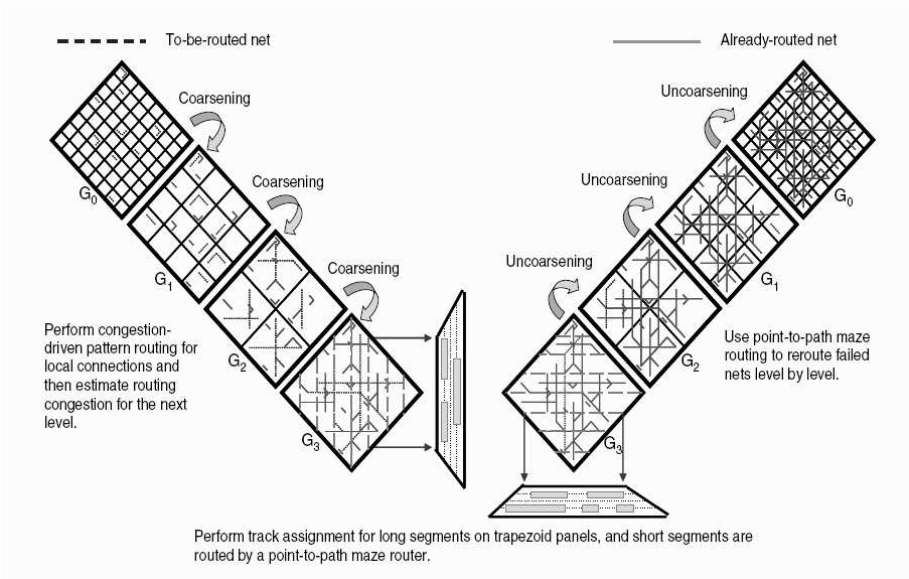


Figure 2.2: 45 degrees of multi-level routing systems

approach to divide all terminals into groups of three-terminal nets (see Figure 2.3 (a)). After that, the authors computed the optimal wirelength of all three-terminal nets, and sorted them by their wirelength (see Figure 2.3 (b)). Further, the authors iteratively picked up a group of three-terminal nets with the minimal wirelength, then routed and merged them to the X-Architecture Steiner Tree (XST) until it was constructed, such as in Figure 2.3 (c).

Compared with the multilevel routing for the Manhattan architecture, experimental results showed that the method of this work reduced wirelength by 18.7 percent and average delay by 8.8 percent with similar routing completion rates and via counts.

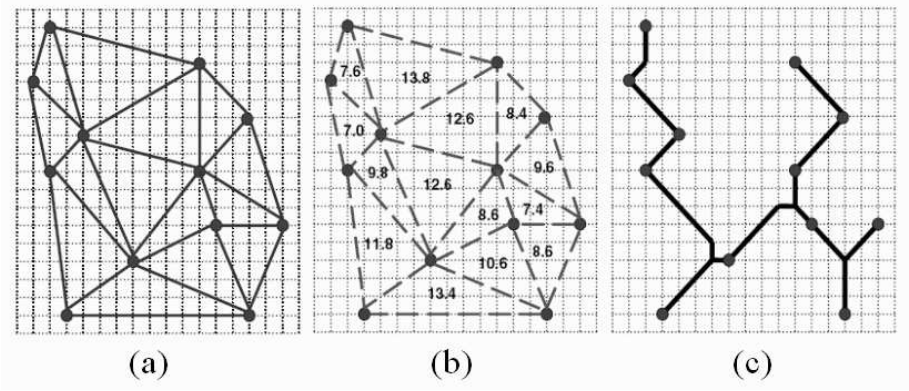


Figure 2.3: (a)Delaunay triangulation of terminals (b)Optimal wirelength of each triangle (c)XST.

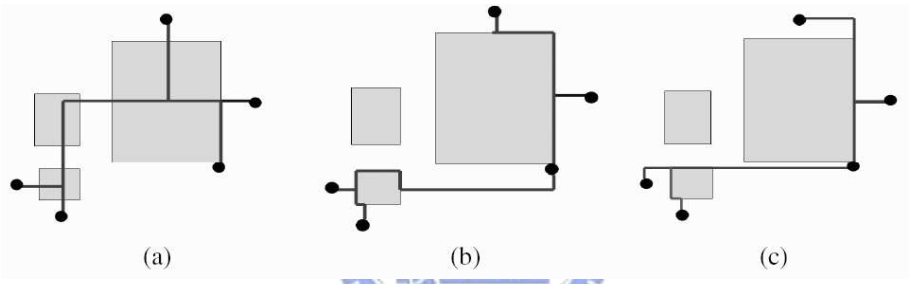


Figure 2.4: (a)complete the rectilinear routing without obstacles (b)remove the overlap line and reroute the obstacle-avoiding path(c)better obstacle-avoiding path.

2.1.2 Researches on Obstacle-Avoiding Routing Tree Construction

2.1.2.1 Rectilinear Steiner Minimal Tree among Obstacles [19]

Rectilinear Steiner minimal tree (RSMT) is a fundamental problem in VLSI/ULSI physical design. Most of the works do not take obstacles into consideration. But in fact, macro cells, IP blocks, and pre-muted nets are often regarded as obstacles in the routing phase, even in placement and floorplanning.

[19] studied RSMT problem among obstacles and presented an $O(mn)$ 2-step heuristic for multi-terminal tree construction. Where m is the number of obstacles and n is the number of terminals. Figure 2.4 shows one example: (a) Build a steiner

tree (b) move the overlap line to solve the overlap problem, but this way cause too many corners (c) actually, there is one better result with less corners.

2.1.2.2 Grid Graph Connection Approach [13]

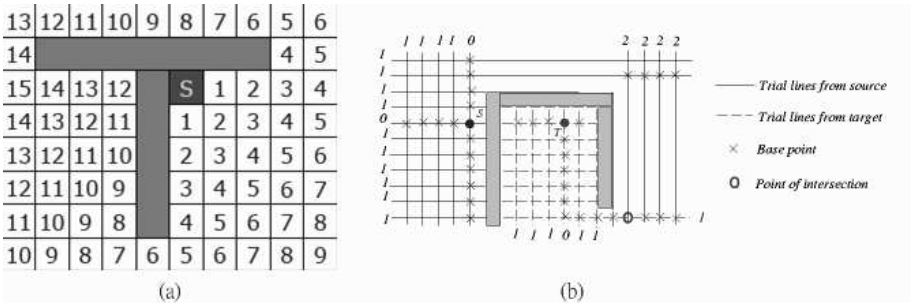


Figure 2.5: (a) The initial routing problem,Maze routing [10] (b) Line search is the variant of Maze routing [13]

Maze routing, first proposed in [10], finds a path from a source to a target on a layer by wave propagation as shown in Figure 2.5 (a). It can get an optimal solution at two-pins net. However, the time complexity and memory usage grow prohibitively huge as the routing area becomes larger. Further, there are some variants [13]. They decide several "escape points" to make the computation more efficient as shown in Figure 2.5(b), but they still incur unsuitable solution quality since they only handle the two-pins net.

2.1.2.3 Hanan Graph and Escape Graph Connection Approach [5]

[5] introduces a connected graph called the Escape Graph, which is similar to Hanan graph. The Hanan graph is built by pins and obstacle boundaries, extending the lines of pins and obstacle boundaries, then we can obtain the Hanan graph shown in Figure 2.6 (a). After removing the redundant lines which are blocked by obstacle from the Hanan graph, we can obtain the graph is called Escape graph shown in Figure 2.6 (b).

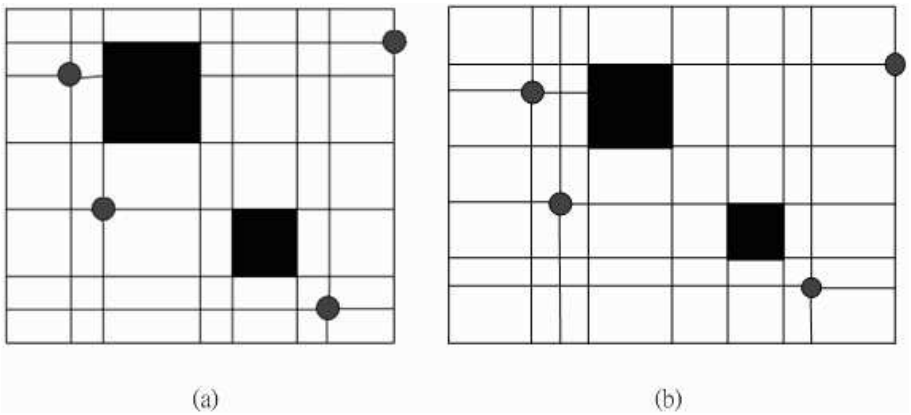


Figure 2.6: (a) Hanan graph consists of the pins and obstacle boundaries and the lines which extended by pins and obstacle boundaries. (b)Escape graph remove the extended edges which are blocked by obstacles.

2.1.2.4 Spanning Graph Connection Approach [18]

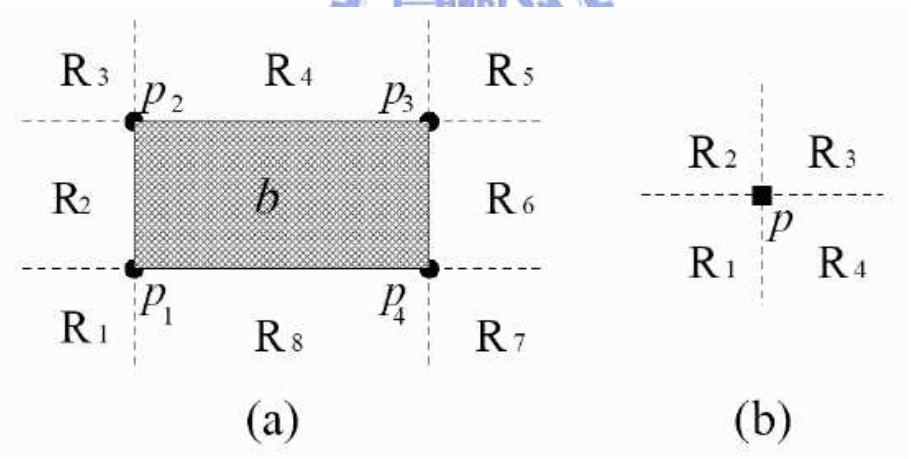


Figure 2.7: (a)Search regions of obstacle, every corner connect to neighbor three regions (b)search regions of pin, the pin connect to neighbor four regions.

Given n points on a plane, a Rectilinear Steiner Minimal Tree (RSMT) connects these points through some extra points called steiner points to achieve a tree with minimal total wire length. Taking blockages into account dramatically increases the problem complexity. It is extremely unlikely that an efficient optimal algorithm exists for Rectilinear Steiner Minimal Tree Construction with Rectilinear Blockages (RSMTRB). In [18], the authors proposed an efficient and effective approach to

solve RSMTRB. The connection graph they used in this approach is called spanning graph which only contains $O(n)$ edges and vertices. An $O(nlogn)$ time algorithm is proposed to construct spanning graph for RSMTRB. This approach can achieve a solution with significantly reduced wire length. The total run time increased is negligible in the whole design flow.

The step of the algorithm is as follows: Connect the blockage corner to the eight regions (Figure 2.7 (a)); connect pin location to the four regions (Figure 2.7 (b)); Search a minimal wirelength spanning tree (Figure 2.8). Figure 2.8 (a) shows they can get an initial circuit setting (include pins and obstacles), Figure 2.8 (b) builds the spanning graph, Figure 2.8 (c) chooses one result from the graph, Figure 2.8 (d)(e) find the other path smaller than the chosen one and replace the path until no path smaller than it, Figure 2.8 (f)finally transforms the path to rectilinear path.

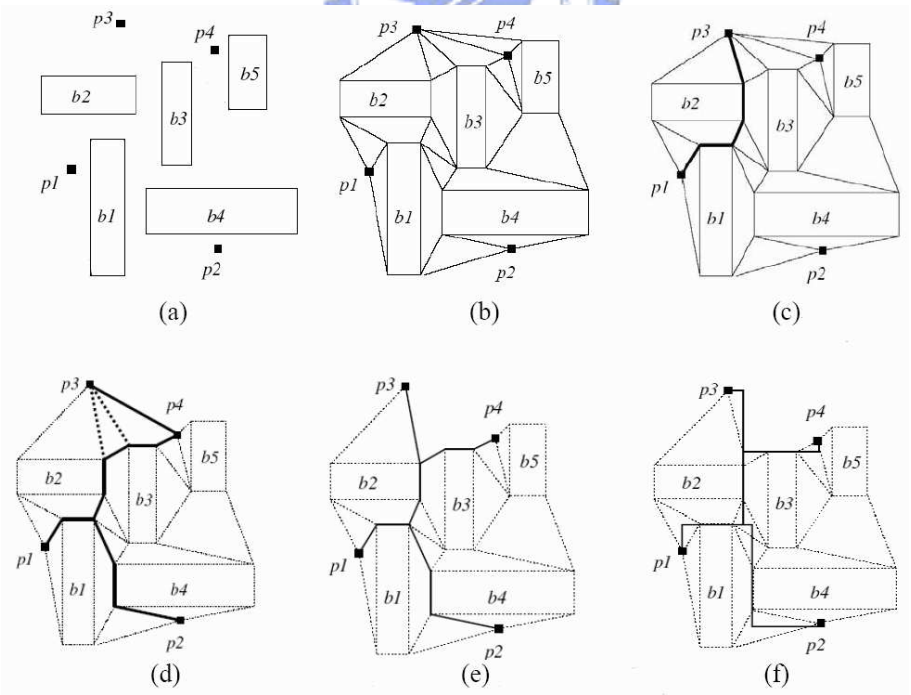


Figure 2.8: (a)Give an initial circuit (b)build the spanning graph (c)choice one path from the graph (d)find the shorter path to replace the choice one (e)until we receive the shortest path (f)transform the shortest path to rectilinear result.

2.1.3 Researches on Multi-layer Structure

2.1.3.1 Routing for System-On-Package [15]

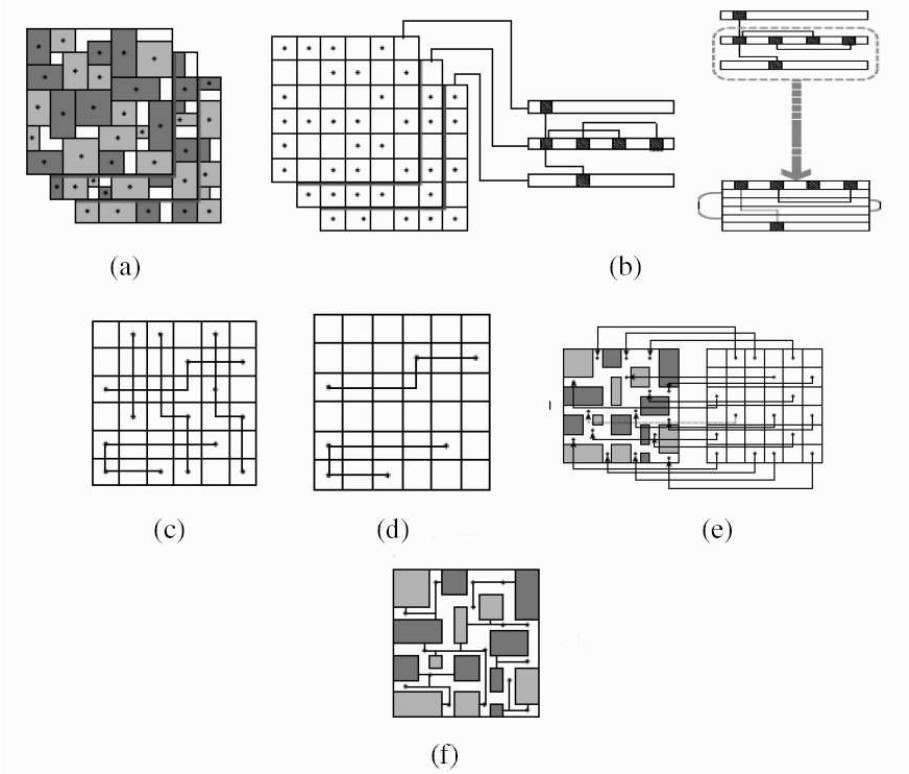


Figure 2.9: (a)pin distribution (b)net distribution (c)topology generation (d)layer assignment (e)channel assignment (f)pin assignment.

3D packaging via System-On-Package (SOP) is a viable alternative to System-On-Chip (SOC) to meet the rigorous requirements of todays mixed signal system integration. In [15], the author presented the first physical layout algorithm for 3D SOP that performs thermal-aware 3D placement and crosstalk-aware 3D global routing.

The 3D router, illustrated in Figure 2.9, is divided into the following steps: (a) coarse pin distribution, (b) net distribution, (c) topology generation, (d) layer assignment, (e) channel assignment, and (f) pin assignment. In the coarse pin distribution step, which is done before net distribution, the author find a coarse

location for the pins and use this information for the net distribution. After the net distribution, the detailed pin distribution step assigns finer location to all pins in each routing interval. A Steiner tree based routing topology for each net is constructed and a layer pair is assigned to it during the topology generation step. The conflict among the nets for routing resources is resolved and layer pairs are assigned during the layer assignment step. The channel assignment problem is to assign each pin in the pin distribution layers to a channel in the placement layers. The purpose of pin assignment is to finish connection between the pins in the routing channel and the pins along the block boundary. As the author focused on analyzing the pin location and obstacles each layer, therefore, the author obtained the better wirelength and fewer number of vias.

2.2 Basic Terminology Definitions

The **rectangle obstacle** and the **non-rectangle obstacle** are the obstacles on the layer, we usually regard all obstacle as the rectangle obstacle for easy calculation, in fact, besides the rectangle obstacle the obstacles may be non-rectangle. A **pin** is a vertex on a layer. A pin must not locate inside any obstacle, but it can be at the corner or on the boundary of an obstacle or inside the fictitious rectangle.

A **via** on layer z is an edge between (x, y, z) and (x, y, z+1). (x, y, z) and (x, y, z+1) must not locate inside any obstacle, but can be at the corner or on the boundary of an obstacle. see Figure 2.10(a), the illegal via is the via locate inside any obstacle , but it can be at the corner or on the boundary of an obstacle(2.10(b)).

The **routable region** is the region without intersecting with any obstacle, but the edge could be point-touched at the corner or line-touched on the boundary of an obstacle. Figure 2.11 shows the tree edges intersecting an obstacle(a), and the tree edges are point-touched and line touch at the obstacle boundary(b).

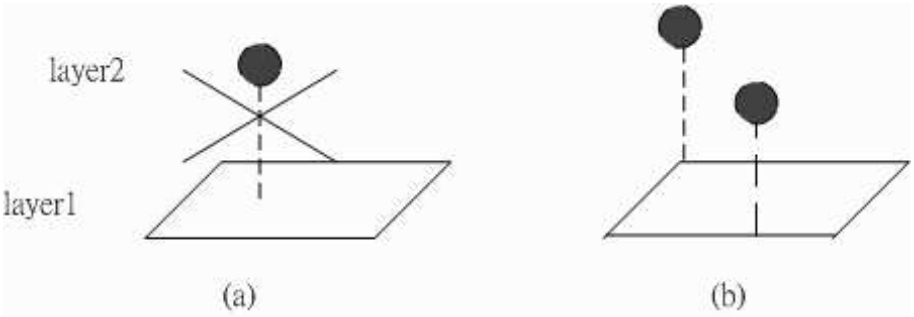


Figure 2.10: (a) A via on layer z is an edge between (x, y, z) and (x, y, z+1). (x, y, z) and (x, y, z+1) must not locate inside any obstacle. (b) It can be at the corner or on the boundary of an obstacle.

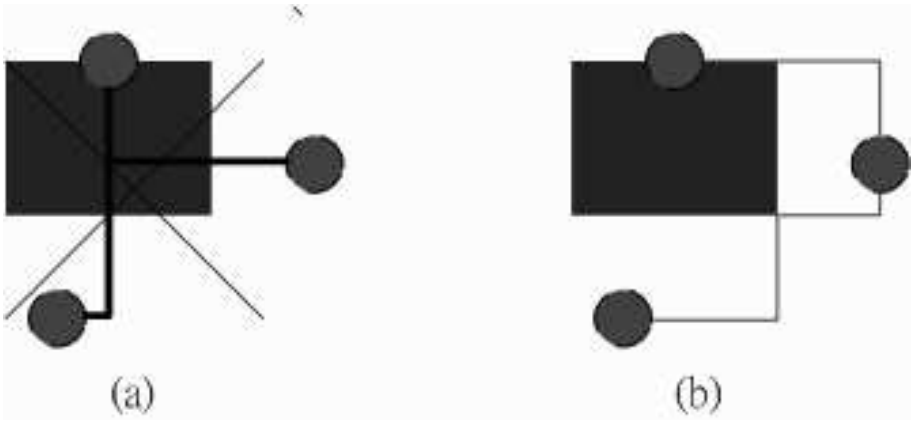


Figure 2.11: (a) the tree edges intersecting an obstacle (b) the tree edges are point-touched and line touch at the obstacle boundary.

2.3 Problem Description

It is well known that the traditional algorithm is generally used to make the connection between two pins with 2-D structure. Different from the traditional algorithm, we accomplish the connection between multiple pins with 3-D structure at the same time. First, we need to provide the algorithm with the initial source, plenty sinks, coordinates of the obstacle and number of the layers. Next, we avoid the unnecessary connection to achieve the connection between the initial source and sinks with our algorithm. At last, we obtain a routing tree with multi-layer and the minimal delay time.

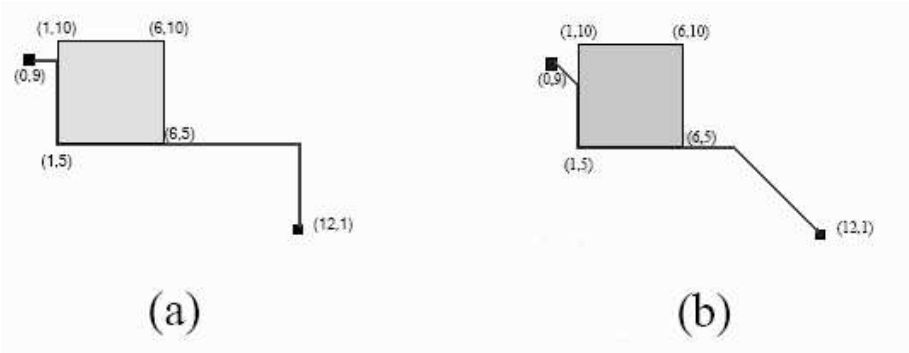


Figure 2.12: (a)The traditional Manhattan steiner path (b)the X-architecture routing path

Owing to the advantage of X-Architecture, we decide to adopt the method of X-Architecture to reduce the total wirelength. There are only vertical and horizontal connections in the traditional Manhattan structure. However, we allow 45 degree connections under X-Architecture. Therefore, we will efficiently obtain a better wirelength with X-Architecture than that with the traditional Manhattan structure, such as Figure 2.12. Besides, we will construct fictitious rectangles to solve the non-rectangular obstacles, and discuss the possible problem as the terminal inside the fictitious rectangle, such as Figure 2.13.

Moreover, we discuss some of the ECO(Engineering Change Order) issues [4]

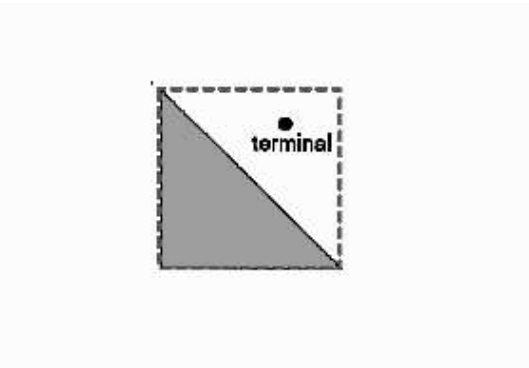


Figure 2.13: We change the non-rectangle obstacle to fictitious rectangle obstacle, there is a terminal inside the fictitious rectangle obstacle.

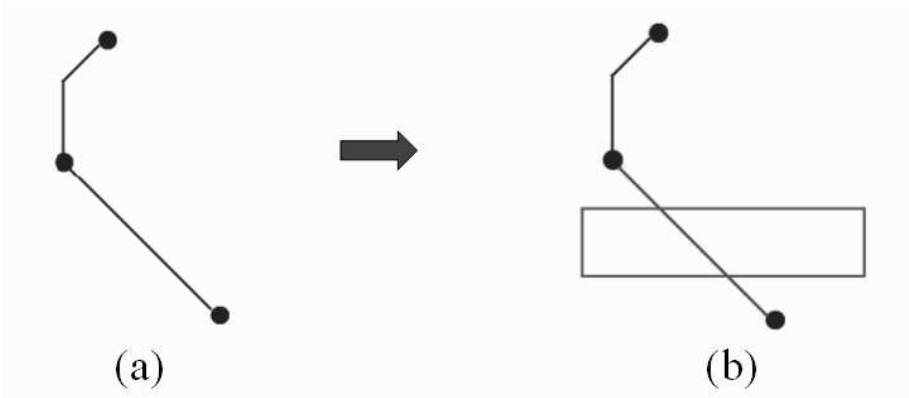


Figure 2.14: (a)Initial xroute routing result (b)Insert new obstacle and the obstacle overlap our initial routing path.

[11]. ECO routing is frequently requested in the later design stage for the purpose of delay and noise optimization. ECO routing is a very important design capability in advanced IC, MCM and PCB designs when additional routings need to be made at the latter stage of the physical design. ECO is difficult in two aspects. First, there are a large number of existing interconnects which become obstacles in the region. A hierarchical approach is not applicable in this situation, and we need to search a large, congested region thoroughly. Second, advances in circuit designs require variable width and variable spacing on interconnects. Thus, a new circuit change is a difficult problem to solve. An example is given as follows. As we accomplish the original routing, we need to insert new obstacles to change the initial setting. If the new obstacles overlap the connection of the original routing, we must dismantle the connection of the overlapped region and find the remaining space to reroute an obstacle-avoiding path between the two pins of the dismantled connection, such as Figure 2.14. In this thesis, we keep the information of the original routing, and therefore we can save the resources and time as rerouting the ECO circuit.

Chapter 3

Our Approach

Since constructing a routing tree is a NP-complete problem, the scholars hope to find an efficient algorithm to obtain a better routing. Besides the traditional Manhattan routing, the technology of X-Architecture is of great help in wirelength reduction. Therefore we can adopt the method of X-Architecture to improve the wirelength. In consideration of the obstacle-avoiding problem, it is necessary for the router to be able to deal with both the rectangular and non-rectangular obstacles. In multi-layer system, the delay time of one via is much larger than that of Manhattan length. According to this reason, we minimize the number of vias. Furthermore, we propose a method to save the resources and time as solving the ECO problem. On the basis of the above-mentioned thoughts, we present an algorithm, which is able to construct the X routing tree under the condition with both the rectangular and non-rectangular obstacles. Further, we minimize the delay time between the preliminary point and the extreme point.

For the multi-layer construction of the Steiner tree , our algorithm consists of the following steps : First, according to the locations of the individual pins, we figure out the central location. And then we project the central location into every layer. Second, we judge whether the location of via between layer l and layer(l+1) is applicable (l=1,2,3,...,n-1). Supposing the location of via between layer l and

layer(l+1) is not applicable, we switch the via to the suitable location. Third, as based on the location of via, we divide each layer into four regions. Afterward we connect all the pins in each divided region in the direction of each layer's central location. Furthermore, we use X-Architecture to modify the original results of the routing. The 3D algorithm is shown in Figure 3.1.

```
Algorithm: Rectangular/ Non-Rectangular Obstacles-Avoiding X-Architecture Router
           in Multilayer System
Input: A set of T= (t1, t2, t3, ... , tn), O= (o1,o2,o3, ... ,on), NO= (no1,no2,no3, ... ,non) with
       the source under a set of layers L= (l1,l2,l3, ... ,ln)
Output: A routing tree which minimizes total wire-length
begin
    for each layer
      Find the location of via
      Divide into four regions according to the via location
      for each region
        if there are two pins in the cluster
          Find the shortest path of two pins
        else
          Complete Delaunay Triangulation
          Find the MST
          if the MST edge through out the obstacle
            if there is non-rectangle obstacle
              Build fictitious rectangle then Spanning Graph
              Find the shortest path
            else
              Construct the Spanning Graph
              Find the shortest path
          else
    Xroute
end;
```

Figure 3.1: The pseudo code of our algorithm

3.1 The Location of Via

The delay time of one via is much larger than that of Manhattan length. According to this reason, we minimize the number of vias. In this step, we must decide the

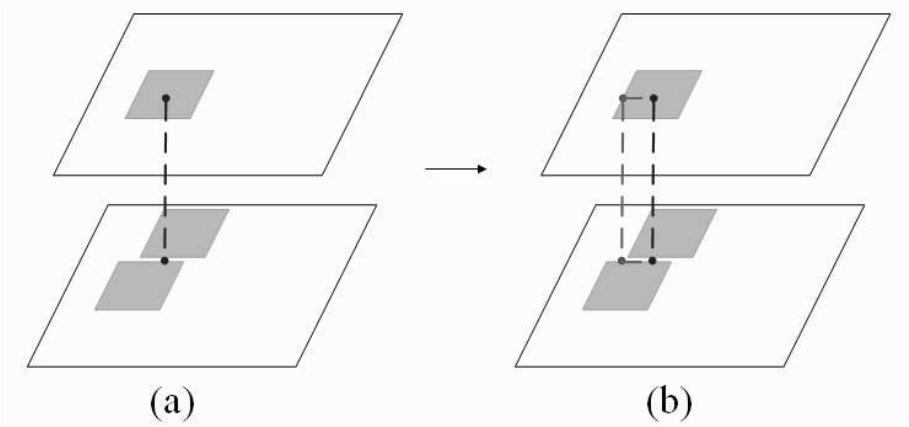


Figure 3.2: (a)Find the location of via (b)correct the location of via

location of via first, And the method, which we use to determine the location of via, is as follows:

First, according to the locations of the individual pins, we figure out the central location. And then we project the central location into every layer. Second, we find the total obstacles on layer l and layer (l+1). Later, we judge whether the location of via between layer l and layer(l+1) locates inside any obstacle or not. (l=1,2,3,...,n-1). Third, supposing the location of via between layer l and layer(l+1) is not applicable, such as Figure 3.2 (a). After finding out all the applicable locations, we separately calculate the lengths between the original central location and all the applicable locations. Then we pick out the suitable location closest to the original central location, as shown in Figure 3.2 (b). Based on the location of via, we divide each layer into four regions, shown in Figure 3.3.

3.2 The Method of Routing

In order to lower the complexity, we divide each layer into four regions according to the location of via in this thesis. Here we construct the routing of four regions on one layer in turn, and then we build the routing of the next layer in the same way,

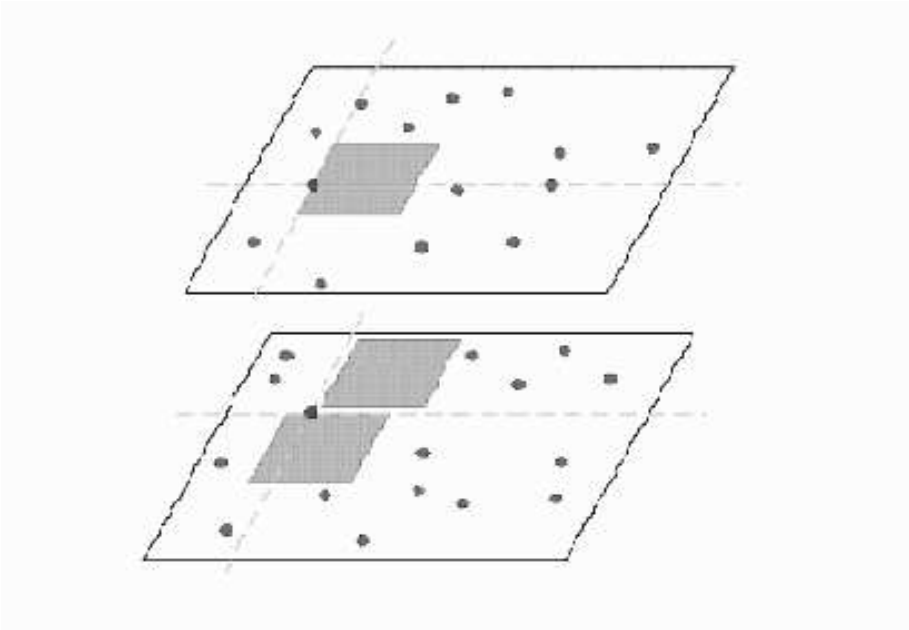


Figure 3.3: As based on the location of via, each layer is divided into four regions.

and so on until we complete the whole routing. For the routing of single region, we use the spanning graph to accomplish the regional connection and further adopt the Delaunay Triangulation (DT) [1] algorithm to figure out the shortest connection between three points. As we can get a better result by combining the spanning graph [18] and DT algorithm, the method of our routing is as follows:

First, we have to calculate the number of the points in single region. If the number is 1, we just do nothing. If the number is 2, we use the spanning graph to find the shortest path. If the number of the points is equal or greater than 3, we use the Delaunay Triangulation (DT) algorithm to obtain the result of the regional connection. With the result of the routing, which is brought by the DT algorithm, we use Kruskal algorithm to figure out the path of spanning tree. The Kruskal algorithm is to arrange all the edges and look for the shortest edge each time and put it in the tree. However, the edges are unable to lead into a loop. Thus we can obtain a shorter spanning tree.

In order to get a minimal spanning tree, we look for the unnecessary edges of the

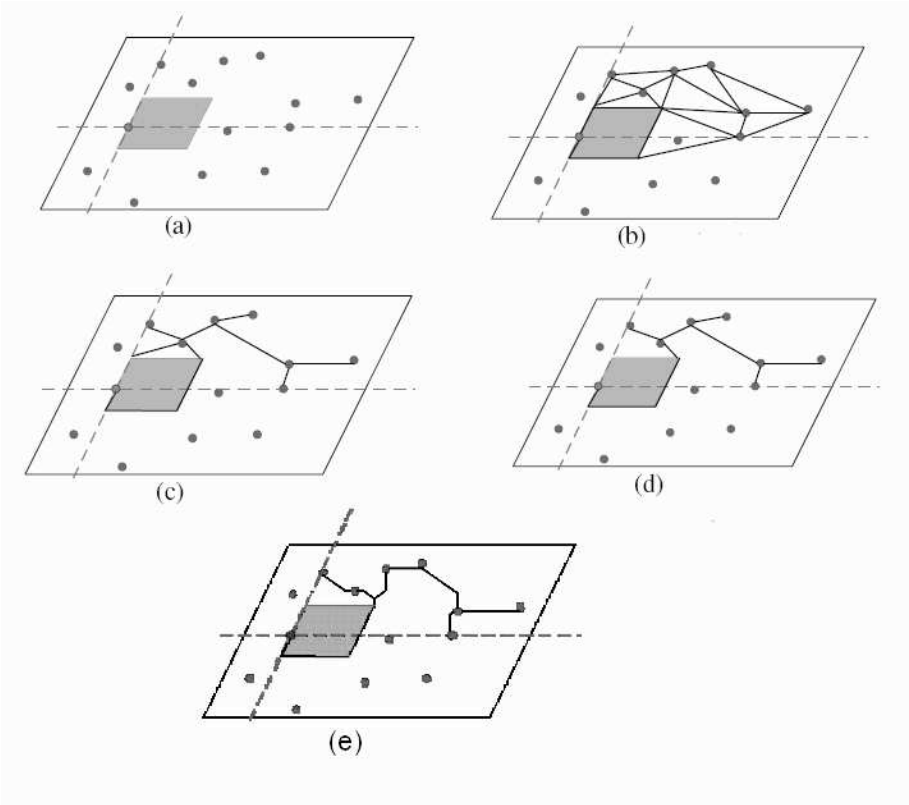


Figure 3.4: (a)The points in the divide region (b)original routing by using DT (c)find the shorter spanning tree from DT (d)remove the unnecessary edge (e)xroute result

spanning tree, which is mentioned before, and remove them. The unnecessary edges mean that the terminal point of the edge is in one corner of the obstacle. Figure 3.4 shows an example to explain the above-mentioned method. Figure 3.4 (a) shows the points in the divided regions. Figure 3.4 (b) displays the original routing in one divided region by using the DT algorithm. We get a shorter spanning tree with the result of the routing, which is brought by the DT algorithm, as shown in Figure 3.4. And we remove the unnecessary edges in Figure 3.4 (d). Finally, we transform the MST into X-Architecture, as shown in Figure 3.4 (e).

3.3 Dealing with the Edge Across Obstacles

Regardless of the range of the obstacles ,the Delaunay Triangulation (DT) algorithm only considers the locations of individual points, including pins and corners of obstacles. Since the DT algorithm invariably forms some inappropriate edges across the obstacles, we construct the spanning graph to adjust such edges of the MST. However, we may divide the obstacles into two types as rectangular and non-rectangular obstacles. If there are rectangular obstacles, the spanning graph can be generated directly. Nevertheless, if there are non-rectangular obstacles, we may not obtain the spanning graph immediately. Therefore, we discuss the two above-mentioned problems separately in this section.

3.3.1 Rectangular Obstacles

If the edge of the MST is on the location across the rectangular obstacle, we call the edge as the inappropriate edge. First, we pick any point of the inappropriate edge as a preliminary point, and then pick another point as a terminal point. At First, we pick any point of the inappropriate edge as a preliminary point, and then pick another point as a terminal point. Next, we construct the spanning graph between the two points, moreover, we figure out the shortest path from the spanning graph. Finally, we transform the shortest path into X-Architecture. Here is an example, as shown in Figure 3.5. Figure 3.5 (a) displays the result by using the DT algorithm in the region. In Figure 3.5 (b), we obtain the MST from the result, which is generated by the DT algorithm. Afterwards, we pick out the inappropriate edge, which is marked as a red one in Figure 3.5 (c). And in this example, the two points of the inappropriate edge are just located at the boundaries of the obstacle. In Figure 3.5 (d), we regard one point as a preliminary point, and another point as a terminal point to re-construct the spanning graph with the corners of the obstacle.

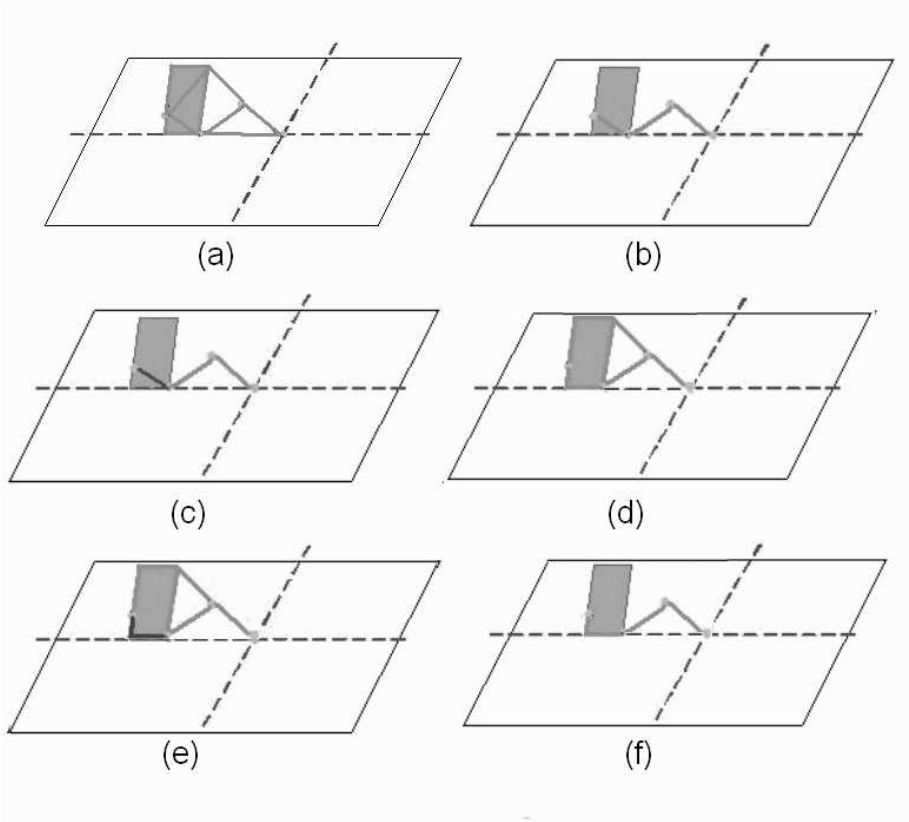


Figure 3.5: (a)The DT result of the region (b)find the MST from DT (c)find the illegal edge (d)build spanning graph to reroute the illegal edge (e)choice the shortest path (f)xroute the shortest path

Since the two points of the inappropriate edge are just located at the boundaries of the obstacle, we obtain two routing paths with the spanning graph. In Figure 3.5 (e), we select the shortest path to put into the tree. At last, we obtain a new minimal spanning tree, as shown in Figure 3.5 (f).

3.3.2 Non-Rectangular Obstacles

As the edge of the MST is on the location across the non-rectangular obstacle, we generate the additional edges to change the non-rectangular obstacle into a rectangular obstacle, which is called a fictitious rectangular obstacle. In this part, we give an example of taking an isosceles right triangle as a non-rectangular obstacle and two terminals.

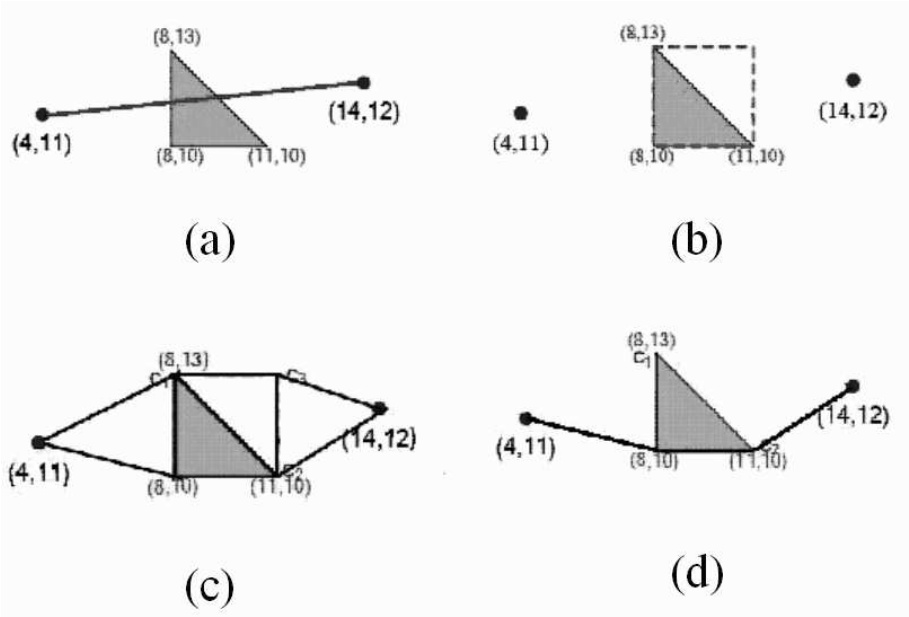


Figure 3.6: (a)The edge through out the obstacle is called illegal edge (b)Build the fictitious rectangle and two pins are outside (c)Generate the spanning graph of the fictitious rectangle (d)Find the shortest path from the spanning graph

First of all, we discuss the terminal location of the fictitious rectangular obstacle. If the terminal is outside of the fictitious rectangular obstacle, we use the spanning graph by regarding the fictitious rectangular obstacle as a normal rectangular obstacle. If the terminal is outside of the fictitious rectangular obstacle, we use the spanning graph to deal with the fictitious rectangular obstacle as a normal rectangular obstacle. However, if the terminal is inside of the fictitious rectangular obstacle, we need to construct the spanning graph within the routable region of the fictitious obstacle, and then generate the spanning graph outside of the the fictitious obstacle. The detailed method will be described in the following section.

3.3.2.1 Terminal is Outside The Fictitious Rectangle Obstacle

Figure 3.6 (a) shows the inappropriate edge of the MST across of the obstacle. After forming the fictitious rectangular obstacle, both the two pins (4,11) and (14,12) are just outside of the fictitious rectangular obstacle, as shown in Figure 3.6 (b). With

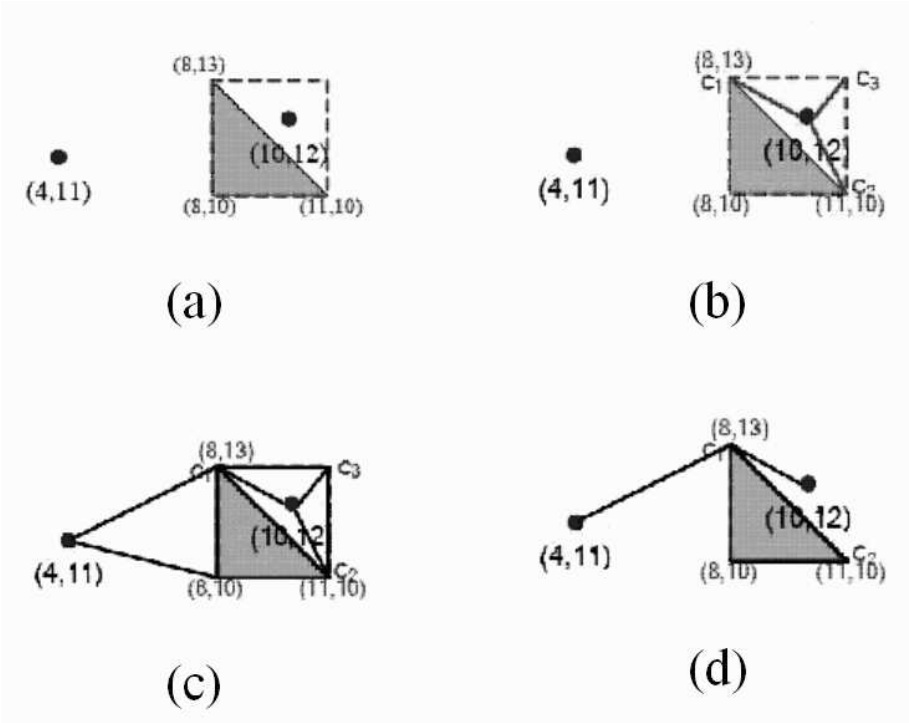


Figure 3.7: (a)Build the fictitious rectangle and pin (10,12) is inside the fictitious rectangle obstacle (b)Build the inside spanning graph of pin (10,12) and the fictitious (c)Generate the outside spanning graph of pin (4,11) and the fictitious (d)Find the shortest path from all path rectangle three corners

the fictitious rectangular obstacle and the two pins, we generate the spanning graph and put the triangular hypotenuse into the spanning graph, as shown in Figure 3.6 (c). After that, we pick out the shortest obstacle-avoiding path and obtain the final result. Figure 3.6 (d) shows that the path (4,11)-(8,10)-(11,10)-(14,12) is the result of this example. At last, we transform the shortest path into X-Architecture.

3.3.2.2 Terminal is Inside The Fictitious Rectangle Obstacle

After forming the fictitious rectangular obstacle, the pin (10,12) is just inside the fictitious rectangular obstacle, as shown in Figure 3.7 (a). First, we construct the spanning graph with the corners (c1 , c2 , c3) of the fictitious rectangular obstacle and the pin (10,12) , as shown in Figure 3.7 (b). Afterwards, we form the spanning graph with the fictitious rectangular obstacle and another pin (4,11) , and put the

triangular hypotenuse into the spanning graph, as shown in Figure 3.7 (c). After that, we pick out the shortest obstacle-avoiding path and obtain the final result. Figure 3.7 (d) shows that the path (4,11)-(8,13)-(10,12) is the result of this example. At last, we transform the shortest path into X-Architecture.

3.4 An ECO Problem: After Inserting New Obstacles

Roughly, there are three aspects of ECO issues : (1) resize the network, (2) location migration, (3) circuit changes. It is more difficult in circuit changes in particular of all mentioned above, since the non-use spaces must be found to be rerouted in our circuits. The new obstacles are inserted to change our original circuits, and the paths which are needed to be change are routed by using our strategies.

It wastes much time and resources to reroute all circuits, especially in only a little change or modifying many times by engineer. Rerouting all circuits several times will cost very much, therefore unnecessary routing work must be diminished to avoid rerouting all circuits when rerouting work needs to be implemented. It is considered that narrowing rerouting area such that rerouting in only partial area is implemented. However, still some unnecessary rerouting work are done while narrowing rerouting area is implemented. What we need to do first is to find out the influenced routing paths and to do necessary rerouting work of the changing paths.

In addition, whether the load of routing work is plenty or not, it takes much and unnecessary algorithm process time from the beginning to the end while rerouting work are implemented by original router. Therefore, we reserve the information constructed during routing process. The associated paths which need to be modified can be found quickly by applying the information while rerouting process. And then

only necessary approaches must be done to find out the modified routing paths. The rerouting results are outputted at last. In this way, we can save plenty of unnecessary time and get rerouting results more quickly.

Chapter 4

Experimental Results

Table 4.1: The working environment

	Type
Hardware	AMD XP 2500+,1.84 GHz,768 MB RAM
Operating system	Microsoft Windows XP
Software	Microsoft Visual C++ 6.0

We obtain 25 benchmark circuits from the CYCU, and we randomly choose seven case to test our algorithm. In addition, we also receive the source code about the multi-layer routing from CYCU. The source code is modified to fit in with our algorithm. There are four tables of the experiment data. The content of the first table is the comparable data between Rectilinear approach and our approach. The second one is the comparable data between the algorithm of CYCU and ours. The third one contains the routing results of non-rectangle obstacles. And the fourth one contains the rerouting results after inserting new obstacles.

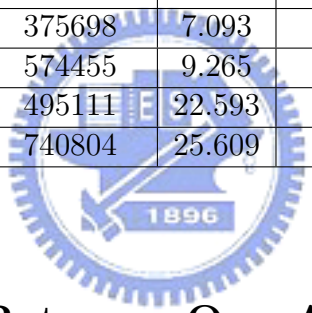
4.1 Comparison Between Our Approach and Manhattan-Wiring Approach

The Manhattan-wiring algorithm was modified from CYCU. The Manhattan-wiring algorithm and our algorithm apply the same method to insert via, thus the structures

of both are almost the same. The extreme difference of them is that the routing paths are implemented by different ways. The former uses the rectilinear path; and ours uses the X-architecture path. Table 4.2 demonstrates that the X-architecture has the better wirelength and almost the same run time.

Table 4.2: The comparison of wirelength and the run time between rectilinear and our xrouting result.

	Pin	Obs.	Manhattan		Xroute		Comparison	
			time	wirelength	time	wirelength	time(%)	wirelength(%)
case1	60	20	0.078	130800	0.078	120705	0%	7.72%
case2	140	20	0.21	217150	0.203	215768	3.33%	0.64%
case3	200	20	0.343	249390	0.359	245280	-4.66%	1.65%
case4	200	600	7.125	375698	7.093	356647	0.45%	5.07%
case5	1000	200	9.078	574455	9.265	555387	-2.06%	3.32%
case6	400	1000	22.39	495111	22.593	471873	-0.91%	4.69%
case7	2000	200	24.375	740804	25.609	726347	-5.06%	1.95%



4.2 Comparison Between Our Approach and [8]

All pins are split up in [8]. First, all pins are projected to the same layer, and then they are divided into four groups according the x and y coordinate. The center each group is found and the via is inserting to the point. According to the via point, each group is divided into four regions. The routing of each region is implemented in order until all of them are implemented. Table 4.3 shows that wirelength of our approach is better but the run time is worse a little than the CYCU algorithm.

4.3 The Skew Comparison

According to the two experiments above, we calculate the skew for comparison. Table 4.4 shows that skew of our approach is averagely better than other methods(the ratio of via = 200).

Table 4.3: The comparison of wirelength and the run time between [8] and our xrouting result.

	Pin	Obs.	[8]		Xroute		Comparison	
			time	wirelength	time	wirelength	time(%)	wirelength(%)
case1	60	20	0.078	132395	0.078	120705	0%	8.83%
case2	140	20	0.203	218820	0.203	215768	0%	1.39%
case3	200	20	0.312	271285	0.359	245280	-15.06%	9.59%
case4	200	600	6.56	379657	7.093	356647	-8.13%	6.06%
case5	1000	200	8.34	602840	9.265	555387	-11.09%	7.87%
case6	400	1000	19.015	524782	22.593	471873	-18.82%	10.08%
case7	2000	200	23.64	789107	25.609	726347	-8.33%	7.95%

Table 4.4: The comparison of skew.

	Pin	Obs.	Skew			Comparison	
			Manhattan	[8]	Xroute	Manhattan(%)	[8](%)
case1	60	20	23190	18050	20824	10.2%	-15.37%
case2	140	20	33820	33820	31407	7.13%	7.13%
case3	200	20	29150	28400	27136	6.9%	4.45%
case4	200	600	34573	33919	33348	3.54%	1.68%
case5	1000	200	42710	51800	39582	7.32%	23.59%
case6	400	1000	39479	41856	37922	3.94%	9.4%
case7	2000	200	76141	74893	71816	5.68%	4.1%
Average						6.39%	5%

4.4 Non-Rectangle Obstacle Avoiding Routing

Twenty rectangle obstacles, twenty-one pins, and two non-rectangle obstacles are included to test our algorithm (isosceles right triangles is used as the non-rectangle obstacles in our experiment). One pin is located exactly inside the routable fictitious rectangle region from the non-rectangle obstacles. The spanning graph of the inner pin is established and combined with other spanning graph. The shortest path is found at last. Table 4.5 shows the routing results include the rectangle and non-rectangle obstacles by our algorithm.

Table 4.5: The obstacle-avoiding routing result of rectangle obstacles and non-rectangle obstacles.

	Pin		Obstacles		Time	Wirelength
	original pin	inside pin	original obs.	non-rectangle obs.		
nrobcase	20	1	20	2	0.047	76095

4.5 Rerouting by Inserting Obstacle

Two ECO routing tests are implemented, one case includes twenty pins and twenty obstacles, and the other one includes two thousand pins and two hundred obstacles. One new obstacle is inserted into both cases respectively. And then the routing path is found and rerouted.

We insert a new obstacle, then we reroute all circuit and our eco approach separately. Table 4.6 shows the rerouting result of two test case of inserting new obstacles.

Table 4.6: The rerouting result of inserting new obstacle.

	Pin	Obstacles		Time		Wirelength	
		obstacles	new obs.	all reroute	eco reroute	all reroute	eco reroute
ecocase1	20	20	1	0.031	0.01	77663	77130
ecocase2	2000	200	1	24.172	0.422	726862	726370

Chapter 5

Conclusions and Future Works

The delay time of one via is much larger than that of Manhattan length. As X-Architecture can obtain the better wirelength, we construct the 3-D routing with X-Architecture router by using the fewest vias. The target is to obtain a 3-D routing tree with a minimal delay time and a minimal wirelength. With additional consideration for the non-rectangle obstacles routing and ECO problem, our router is able to deal with the probable problems under different conditions. According to our experimental results, we can obtain the better wirelength and skew than other methods. Moreover, our router also obtains the better wirelength during rerouting, and we can complete the rerouting fast.

In this work, non-rectangle obstacle issues applying isosceles right triangle and ECO issues as inserting of obstacles are discussed as well. In the future, we wish our routers can deal with more types of non-rectangle obstacles(eg. polygon) and different ECO issues(eg. changing of net).

Bibliography

[1] M. Berg, M. Krcveld, M. Overmars, and 0. Schwarzkapf. "Computational Geometry: Algorithms and Applicataons". In *2nd Edition, Springer-Verlag*, 2000.

[2] C. F. Chang and Y. W. Chang. "X-Route: An X-Architecture Full-Chip Multilevel Router". In *Proceedings IEEE International SOC Conference*, pages 229–232, 2007.

[3] S. P. Chang, H. H. Huang, Y. C. Lin, and T. M. Hsieh. "A Timing-Driven X-Architecture Router with Obstacles". In *VLSI Design/CAD Symposium*, 2007.

[4] J. Cong, Jie Fang, and K. Y. Khoo. "An implicit connection graph maze routing algorithm for ECO routing". In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pages 163–167, 1999.

[5] J. L. Ganley and J. P. Cohoon. "Routing a Multi-Terminal Critical Net: Steiner Tree Construction in Presence of Obstacles". In *Proceedings Internationl Symposium on Circuits and Systems*, pages 113–116, 1994.

[6] T. Y. Ho, C. F. Chang, Y. W. Chang, and S. J. Chen. "Multilevel Full-Chip Routing for the X-Based Architecture". In *Proceedings IEEE/ACM Design Automation Conference*, pages 597–602, 2005.

[7] Y. Hu, T. Jing, X. Hong, Z. Feng, X. Hu, and G. Yan. "An-OARSMan:Obstacle-Avoiding Routing Tree Construction with Good Length Performance". In *Pro-*

ceedings ACM/IEEE Asia and South Pacific Design Automation Conference, pages 7–12, 2005.

[8] H. H. Huang, Y. C. Lin, H. Y. Huang, and T. M. Hsieh. "Partition-based Routing Tree Algorithm with Obstacles". In *Proceedings of IEEE International Symposium on Integrated Circuits*, pages 196–199, 2007.

[9] A. B. Kahng and G. Robins. "A new Class of Steiner Tree Heuristics with Good Performance". In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, volume 11, pages 893–902, 1992.

[10] C. Y. Lee. "An Algorithm for Connections and Its Application". In *IRE Transactions on Electronic Computer*, pages 346–365, 1961.

[11] Y. L. Li, J. Y. Li, and W. B. Chen. "An Efficient Tile-Based ECO Router Using Routing Graph Reduction and Enhanced Global Routing Flow". In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 345–358, 2006.

[12] S. K. Lim. "Physical design for 3D system on package". In *IEEE Design and Test of Computers*, volume 22, pages 532–539, 2005.

[13] K. Mikami and K. Tabuchi. "A Computer Program for Optimal Routing of Printed Circuit Conductors". In *Proceedings of IFIP Congress*, volume 2, pages 1475–1478, 1968.

[14] J. M. Minz, E. Wong, M. Pathak, and S.K. Lim. "Placement and Routing for 3-D System-On-Package Designs". In *IEEE Transactions on Components and Packaging Technologies*, volume 29, pages 644–657, 2006.

[15] J. R. Minz, E. Wang, and S. K. Lim. "Thermal and Crosstalk-Aware Physical Design for 3D System-On-Package". In *Proc. of IEEE Electrical Performance of Electronic Packaging*, pages 824–831, 2005.

[16] M. Pathak and S. K. Lim. "Thermal-aware Steiner Routing for 3D Stacked ICs". In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pages 205–211, 2007.

[17] R. Ravichandran, J. Minz, M. Pathak, S. Easwar, and S. K. Lim. "Physical layout automation for system-on-packages". In *IEEE Electronic Components and Technology Conference*, 2004.

[18] Z. Shen, C. C. N. Chu, and Y. M. Li. "Efficient Rectilinear Steiner Tree Construction with Rectilinear Blockages". In *Proceedings IEEE International Conference on Computer Design*, pages 38–44, 2005.

[19] Y. Yang, Q. Zhu, T. Jing, and X.L Hong. "Rectilinear Steiner Minimal Tree among Obstacles". In *Proc. of IEEE ASIC 5th Intlernational Conference*, pages 348–351, 2003.

作者簡歷

石佳正，民國六十八年十二月出生於台南市。民國九十一年六月畢業於私立義守大學電子工程學系，並於民國九十四年九月進入國立交通大學電機學院 IC 設計產業研發碩士班就讀，從事 VLSI 實體設計方面相關研究。民國九十六年十一月取得碩士學位，碩士論文題目為『考慮矩形/非矩形障礙物的 X 結構多層繞線器』。