國立交通大學

電子工程學系 電子研究所

碩 士 論 文

晶片-封裝-印刷電路板的介面設計之演算法

The Algorithms for Chip-Package-Board Interfacing

研 究 生：徐欣吳

指導教授：陳宏明 教授

中 華 民 國 一〇〇 年 七 月

晶晶片-封裝-印刷電路板的介面設計之演算法

The Algorithms for Chip-Package-Board Interfacing

研 究 生：徐欣吳　　Student：Hsin-Wu Hsu

指導教授：陳宏明　　Advisor：Hung-Ming Chen

國 立 交 通 大 學

電子工程學系 電子研究所

碩 士 論 文

A Thesis

Submitted to Department of Electronics Engineering and

Institute of Electronics

College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electronics Engineering

July 2011

Hsinchu, Taiwan, Republic of China

中華民國一００年七月

晶片-封裝-印刷電路板的介面設計之演算法

學生: 徐欣吳　　　　指導教授: 陳宏明 教授

國立交通大學　電子工程學系　電子研究所　碩士班

摘　　要

現今的晶片(Chip)、封裝(Package)和印刷電路板(Printed Circuit Board)，是藉由介面(Interfaces)的整合，把各自獨立的設計再加以組合成為單一個系統。隨著輸入輸出阜(Input/Output Port)的數量增加，介面也隨之變得更加複雜，然而，很少電子設計自動化工具(Electronic Design Automation tools)可以提供有效的支援。由介面所衍生的問題通常與許多設計決策(Design Decisions)有關，例如一個新產品從構思到實際推入市場所用的時間(Time-To-Market)以及生產效率；然而，這類問題的判斷不容易被公式化，因此，迫切地需要一些實際且有效的方法，推進晶片/封裝/系統的設計方便性。

另一方面，在當今的設計流程中，晶片設計廠必須與封裝廠反覆地重做重新分佈層的繞線(Re-Distribution Layer Routing)，因此，從晶片設計廠的角度而言，必須與封裝廠發展協同設計、以及採用好的重新分佈層繞線器(RDL Router)，以達成快速地實現介面的目的。

我們提出的晶片-封裝-印刷電路板的介面設計之演算法包含兩個部份。

第一個作品是適用於繞線空間極擁擠、單層繞線無法達到 100%可繞度(Routability)的情形下，採用假單層繞線概念的重新分佈層繞線器。套用於工業界的測試案例上，我們的方法可以達到 100%的可繞度且同時可縮小繞線使用道

的面積，表現甚至優於當今最先進的商用重新分佈層繞線器。第二個作品包含產生封裝針腳與金屬線規畫的方法，針對晶片-封裝-印刷電路板的共同設計，可避免曠費時日的繞線過程，直接產生金屬線規畫，用以估計封裝大小、信號完整性(Signal Integrity)、及可繞度分析。我們的方法可以實現晶片、封裝、系統設計的快速收斂，藉由這兩個作品，設計廠可以大幅減少設計的困難度及加快上市的時間。

Algorithms for Chip-Package-Board Interfacing

Student: Hsin-Wu Hsu Advisor: Prof. Hung-Ming Chen

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

ABSTRACT

Chip, package, and board are nowadays designed separately and then combined into one system by their interfaces. The interfaces have become much more complex as the number of input/output (IO) pins increases. Few commercial EDA tools provide effective support. Since the problems caused by interfaces involve many design decisions such as time-to-market (TTM) and productivity, and it is not easy to formulate, some practical and efficient interfacing methods are strongly in need to facilitate chip/package/system designs.

On the other hand, iterative re-works with package houses and RDL trial routing exist in conventional design flow. Accordingly, from design houses' point of view, co-design with package houses and good RDL router must be developed to enable fast implementation of RDL. Our proposed algorithms for chip-package-board interfacing contain two parts.

The first work is RDL routing on pseudo single-layer which targets at congested cases where 100% routability cannot be achieved within single layer. Our approach can achieve 100% routability and minimize the area for 2-layer routing on a real industrial case, outperforming a state-of-the-art commercial RDL router. The second work contains the methodologies which can generate package pin-out and wire planning for chip-package-board co-design. It provides wire planning without time-consuming routing process to estimate package size, signal integrity, and routability. Our approaches can enable fast re-spin between chip, package, and system design houses. Through these two works, design houses can greatly reduce the design efforts and time-to-market.

誌　　　　謝

感謝陳宏明老師，您陪伴我從大學到研究所的路上，不僅有學術上、生活上的指導，更給我許多表現的機會。

謝謝仁傑學長及創意電子的主管 Hock Chen，你們在研究上給的指引，幫助我堅定地往前走。

感謝實驗室的學長姐、學弟妹，與你們在這些日子的相處、共同努力、享樂，都是我最重要的能量來源。特別謝謝長毅、孟伶、篤雄，沒有你們的集思廣益，就沒有今天這份成果。

謝謝我的家人、親戚們，你們是最大的功臣，持續地支持我專心致力於學業。

謝謝每個朋友，欣吳的成就歸功於你們。

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Contributions

Chip, package, and board are nowadays designed separately and then combined into one system by their interfaces. The process of separate designing induces an information gap. If companies in cooperation have, for example, a meeting per week, and then the information gap exists until next meeting. The interfaces have become much more complex as the number of input/output (IO) pins increases. With the advent of flip-chip ball grid array (FCBGA) packaging, the number of IOs has reached up to more than a thousand. Such complexity makes it more difficult, time-consuming, and uncertain to design. Even worse, not many commercial EDA tools provide effective support. It is because the nature of separation between chip, package, and system design houses. The problems caused by interfaces involve many design decisions such as time-to-market (TTM) and productivity, and it is not easy to formulate. Therefore, some practical and efficient interfacing methods are indeed in need to facilitate chip/package/system designs.

Among all packaging technologies, FCBGA is the best choice in electrical performance and IO count. In such packaging architecture, the interface between chip and package is re-distribution layer (RDL). A typical design flow is shown in Fig. 1.1. Implementation of RDL is composed of floorplanning, assignment, and routing for

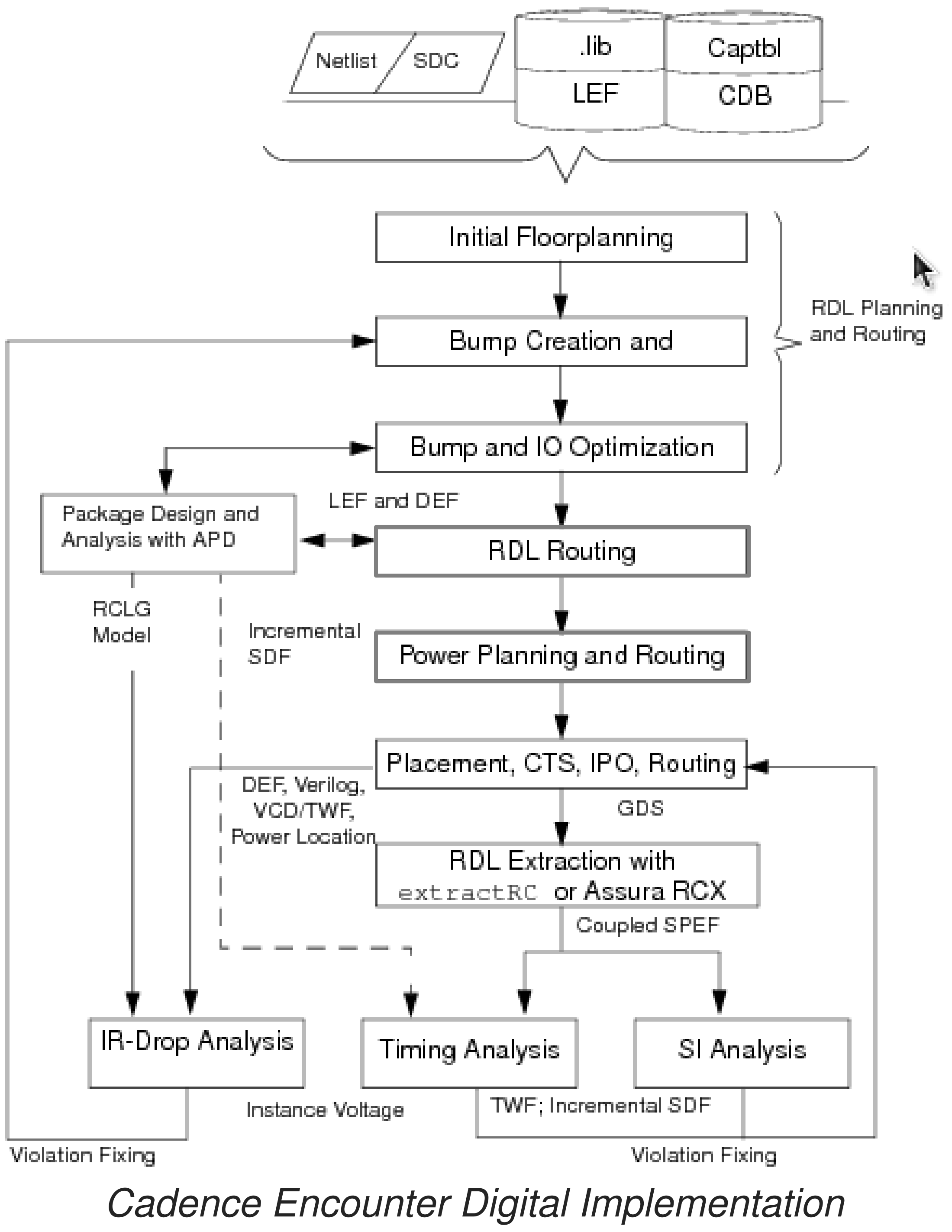


Figure 1.1: Typical chip design flow. In the implementation of RDL, floorplanning, assignment, and routing for bump pads and IO pads are in early stage of design flow. Iterative re-works with package houses and RDL trial routing are needed.

bump pads and IO pads. They first decide the area of die, estimate the number of IOs, floorplan the bump/IO pads, and then send this initial planning to package houses and RDL designers to evaluate the feasibility. If a failure is reported from either package houses or RDL designers, and then the initial planning has to be modified and check the feasibility again. These iterative re-works with package houses and RDL trial routing are then inevitable. The physical design in later stages such as placement and routing (P&R) cannot begin before the iterative re-works converge. Accordingly, from chip design houses' point of view, co-design with package houses and good RDL router must be developed to enable fast implementation of RDL. We hereby propose algorithms for chip-package-board interfacing.

The proposed RDL router targets at congested cases where 100% routability cannot be achieved within single layer. The concept of pseudo single-layer routing is introduced. It is to employ a small area, which is less critical in performance, from another metal layer. With some techniques such as regional layer allocation and assignment of movable pins, the problem is solved by classic channel routing algorithms. Our approaches can achieve 100% routability and minimize the area for 2-layer routing on a real industrial case.

Another proposed co-design methodology can generate package pin-out and wire planning for chip-package-board co-design. It provides wire planning without time-consuming routing process to estimate package size, signal integrity (SI), and routability. Some heuristics are used to generate an initial via/ball assignment that its corresponding routes are monotonic. Two post optimization schemes are proposed to further minimize the cost. Our approaches can enable fast implementation re-spin between chip, package, and system design houses.

1.2 Organization of This Thesis

The remainder of this thesis is organized as follows. The algorithms for RDL routing on pseudo single-layer are proposed in Chapter 2. The idea of routing on pseudo single-layer is defined, and then the modeling of routable region is discussed, and then our successive channel routing algorithm is addressed. The methodologies that generate package pin-out and wire planning for chip-package-board co-design are proposed in Chapter 3. Pin-out designation method for wire planning and pin-out optimization is addressed. Lastly, we draw the conclusions and list some future works in Chapter 4.

Chapter 2

RDL Routing on Pseudo Single-Layer

2.1 Overview

Re-distribution layer (RDL) is the special metal layer used to implement flip-chip assembly. Sometimes RDL is so congested that the capacity for routing is insufficient. Routing therefore cannot be completed within a single layer even for manual routing. [2] proposed a routing algorithm that uses two layers of RDLs. It can be used to solve the cases that require the whole area of two RDLs. But it is often that the required routing area is a little more than one layer. This problem can be overcome by adopting the concept of pseudo single-layer. With the algorithms for routing on successive channels, the area of 2-layer routing can be minimized and the routability is 100%. Comparisons of routing results between manual design, the commercial tool, and the proposed method are addressed. We show that, on a real industrial case that originally required fully manual design, the proposed method can finish RDL routing automatically and effectively.

2.1.1 Introduction to RDL Design

As the demand for more IO count increases, traditional packaging such as wire bonding is not effective to support thousands of IOs. Flip-chip assembly is now

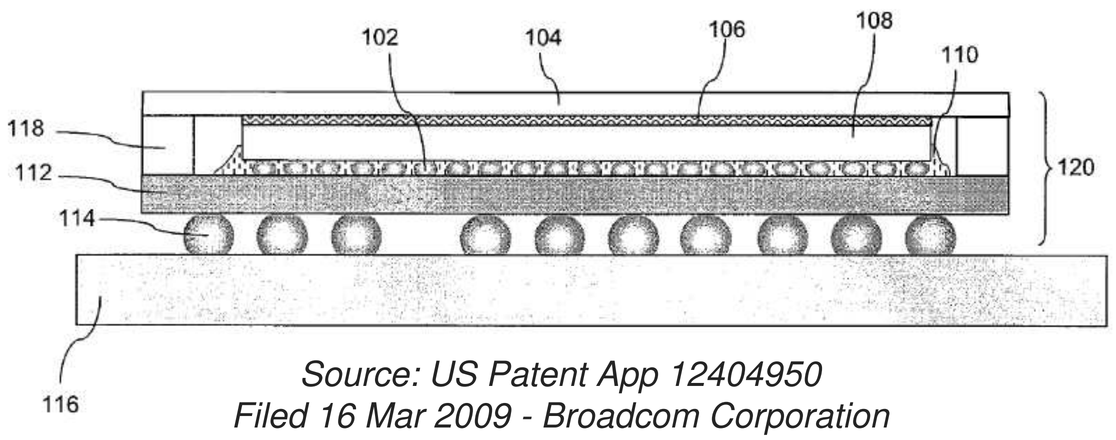

102
104
106
108
110
118
112
114
120
116
Source: US Patent App 12404950
Filed 16 Mar 2009 - Broadcom Corporation

Figure 2.1: Flip-chip ball grid array (FCBGA) package. In this example, a die(108) is coupled to flip-chip bumps(102). Bumps(102) and package balls(114) are molded on a package substrate(112). Printed circuit board(116) (PCB) is the carrier of chip(120).

commonly used to replace wire bonding in high end products because it reduces chip area while supporting more IOs. It can also greatly reduce inductance, allow high-speed signals, and carry heat better. For high IO count chips, a general purpose packaging method is flip-chip ball grid array (FCBGA). As shown in Fig. 2.1, the structure of FCBGA is composed of a die coupled to flip-chip bumps coupled to a package substrate coupled to package balls.

Due to FC application, RDL is used on top of core metals which makes the IO pads of die available in other locations. It enables bonding out IO pads to other locations such as bump pads. Bumps are usually placed in a grid pattern and serve as the interface of flip-chip. Each bump possesses two pads, one on the top, one in the bottom, attached to RDL and package substrate respectively, as shown in Fig. 2.2. Fig. 2.3 shows an RDL serves as a layer connecting I/O pads and bump pads.

Both RDL routing and bump assignment are additional implementation tasks for design houses to migrate designs from wire bonding to flip-chip. Bump assignment is to assign each bump to a specific IO pad. Since IO pads are put on peripheral

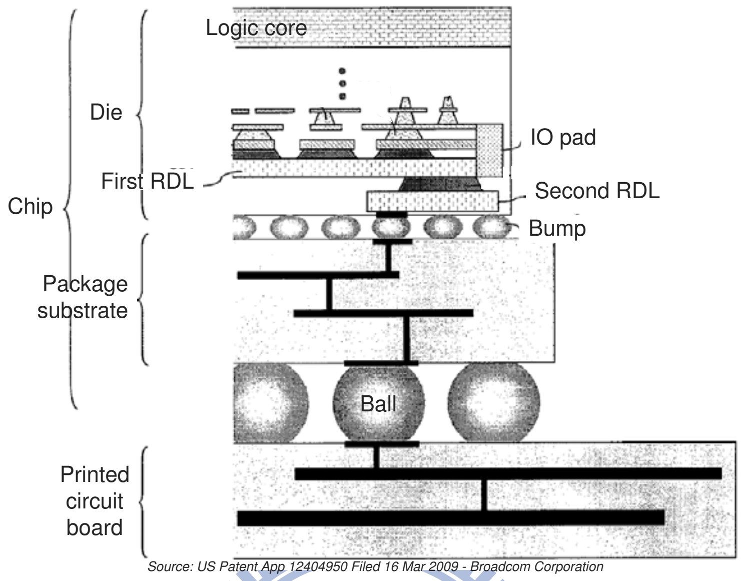


Figure 2.2: First RDL(314) connects with top metal layer(222a) and second RDL(312) by via. Second RDL connects with first RDL and bump(102) by via(316) and bump pad respectively. Both first/second RDL can couple to IO pad(226) on the periphery.

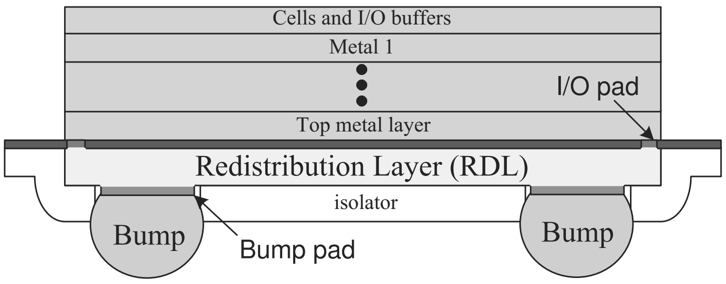


Figure 2.3: Cross-section of flip-chip. An redistribution layer (RDL) is an extra layer above top metal layer, which couples to IO pads and bump pads.

of die for most designs, the flylines and signal routing look like nets escaping from center to boundary of chip.

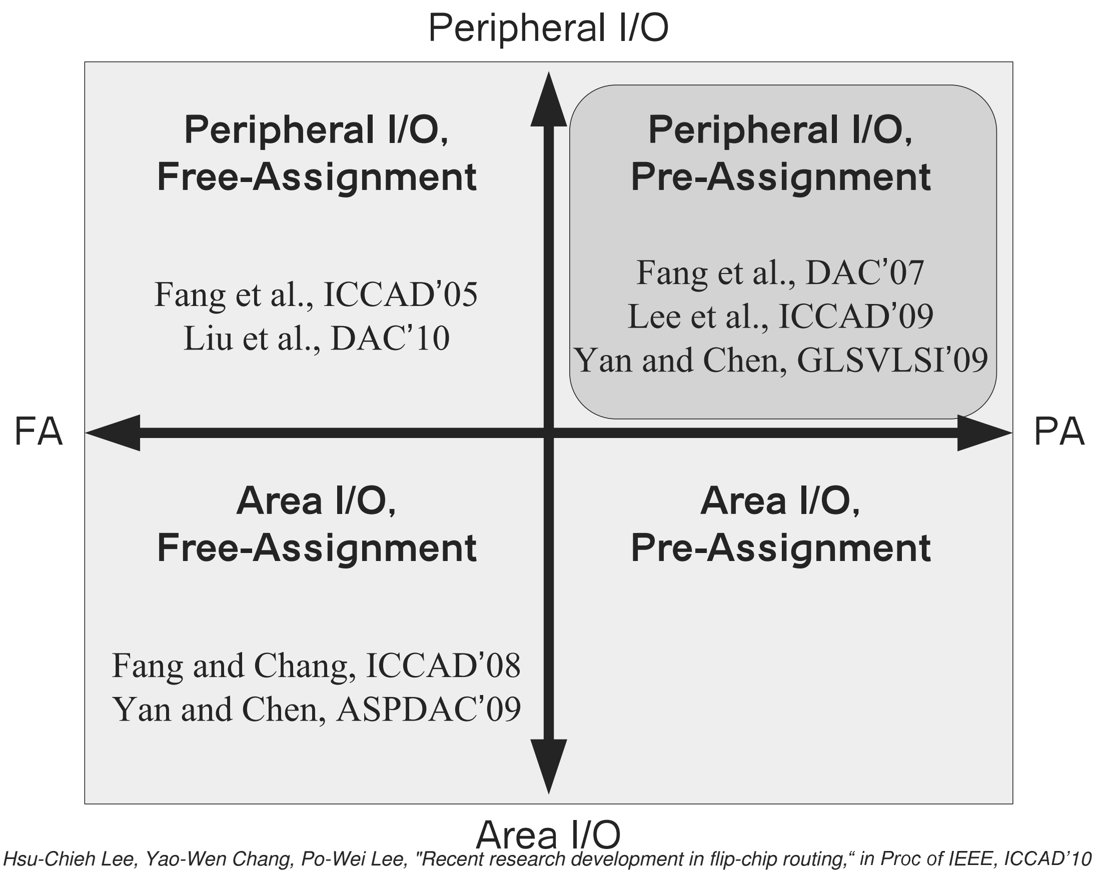
For our designs under consideration, there are two layers of RDL. Metal 9 (M9) is used to implement power/ground (PG) mesh and power routing. Metal 10 (M10) is used to route all signal nets.

2.1.2 Previous Works

Previous works can be classified into four types based on the flip-chip structures and pad assignment methods. A research map proposed by [3] is shown in Fig. 2.4. Two pad assignment methods, free-assignment (FA) and pre-assignment (PA), represent whether the $bumppad - IOpad$ mapping is given as input. For FA problems, each IO pad is free to assign any bump pad, so an IO pad is more likely to connect to the bump pads close to it. For PA problems, algorithms focus on solving complex crossings, so it is more difficult than FA but more convenient for designers.

Two flip-chip structures, area-IO (AIO) and peripheral-IO (PIO), represent patterns of IO placement. AIO and PIO problems are to place IOs in the central area and on the peripheral of die respectively. PIO is more popular today mainly because of its implementation simplicity even though AIO is theoretically better in performance. An example of PIO is shown in Fig. 2.5.

PIO-FA problems are solved by [4] using network flow algorithms such as minimum cost maximum flow (MCMF) algorithm. PIO-PA problems are solved by [5, 6, 8] using some heuristics and integer linear programming (ILP) respectively. Under the routing model of [6, 8, 7], their ILP method guarantees an optimal solution. AIO-FA problems considering signal skew are solved by [9, 10] using MCMF. Recently, the problems regarding unified AIO, which means an RDL containing both AIO-FA and AIO-PA problems, are solved by [11]. Some works take differential



Hsu-Chieh Lee, Yao-Wen Chang, Po-Wei Lee, "Recent research development in flip-chip routing," in Proc of IEEE, ICCAD'10

Figure 2.4: Research map of previous works. Free-assignment (FA) and pre-assignment (PA) are 2 methods for pad assignment. Peripheral-IO (PIO) and area-IO (AIO) are 2 flip-chip structures. In this work, we focus on the problem of PA and PIO.

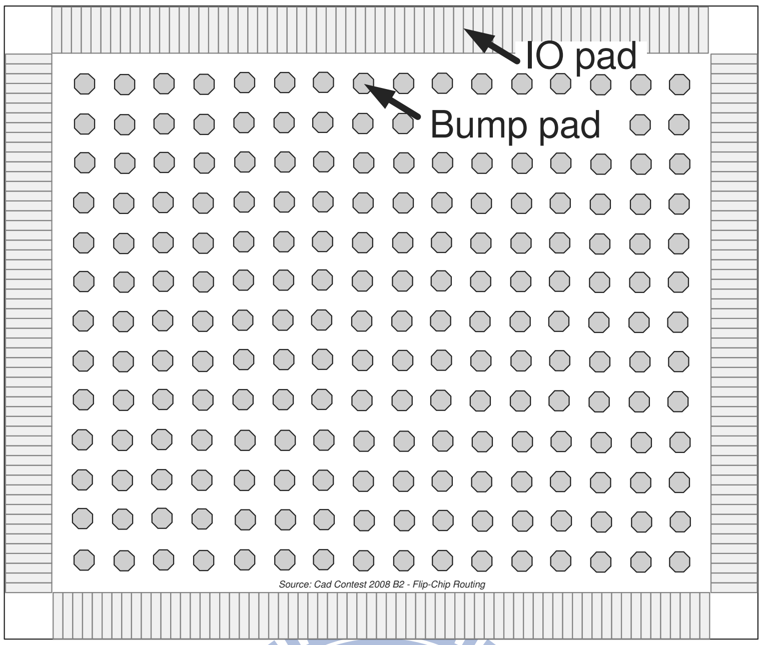


Figure 2.5: Top view of RDL. There are bump pads that are in a grid pattern and IO pads on the periphery. This is an example of real scale design.

pairs or floorplanning into account. In this work, we focus on the problem of PA and PIO, related previous works are listed in the top-right quadrant of Fig. 2.4.

Most previous works focus on single-layer routing. They limit routes within one metal layer, on which every net must be routed. The common objective is wire-length minimization. Their optimization schemes are performed under a prerequisite that routability is 100%. They are successful for each type of RDL routing problems, providing that a solution exists within single layer.

2.1.3 Routing on Pseudo Single-Layer

The number of signal nets to be routed is, however, generally huge for RDL. Bump pads are large in area and are seen as obstacles in routing stage. An example of real scale design is shown in Fig. 2.5. Fig. 2.6(a) shows an example of congested RDL where six nets, $netA, netB, ..., netE$, are shown in flylines. Such designs are

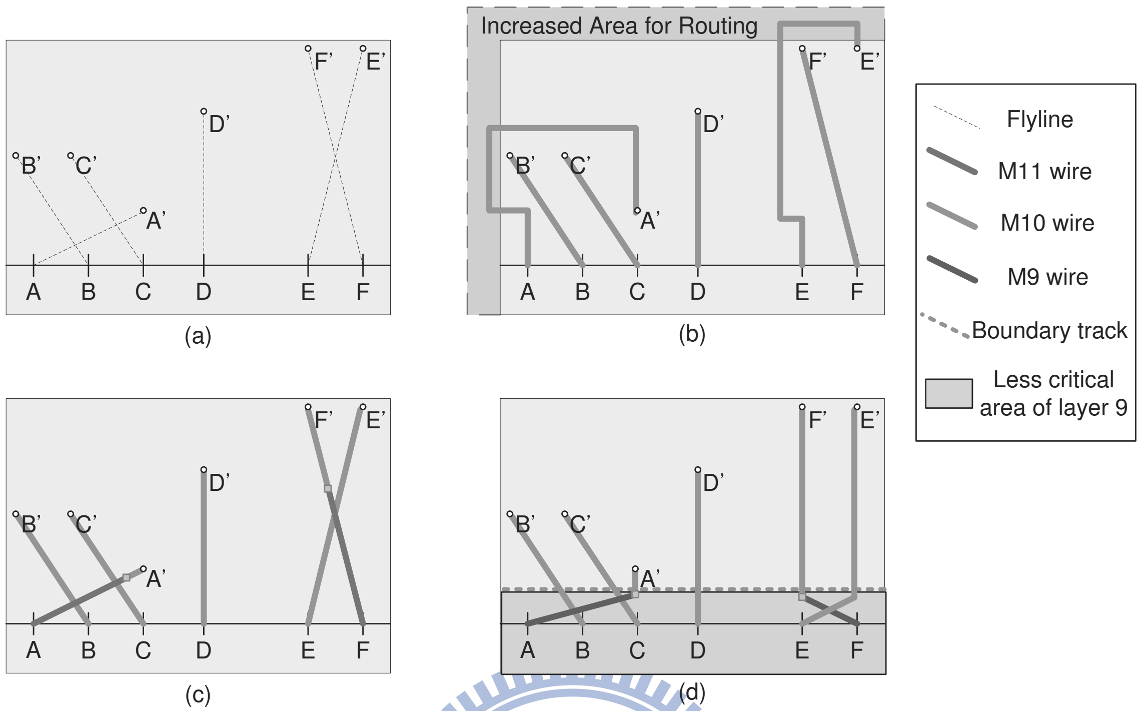


Figure 2.6: Solutions for congested RDL routing. An RDL is congested if it has so many signal IO nets that a single RDL cannot provide sufficient capacity for routing. In (a), 2 trivial solutions are shown in (b) and (c). Solution (b) increases the area of RDL (M10), while solution (c) adds an extra RDL (M11). We proposed another solution, pseudo single-layer, which is a compromise in-between. It is to employ a small region of an existing RDL (M9), as shown in (d).

so congested that 100% routability cannot be achieved within single layer (ex: layer 10), so we must consider two trivial solutions. One is to increase the area of RDL (ex: layer 10), which is equivalent to increasing the die-size, as shown in Fig. 2.6(b). Another is to add an extra layer of RDL (ex: layer 11), as shown in Fig. 2.6(c). However, neither of the solutions above is acceptable for cost concern.

We hereby introduce the concept called $Pseudo\ Single-Layer\ Routing$. It is to borrow a small area from another existing metal layer (ex: layer 9). This is practical and cost-effective provided that the area is less critical in performance. In the example shown in Fig. 2.6(d), some area of layer 9 (the pink area) is borrowed to complete routing. Here we assume that the area is defined as the place between a boundary track (the dotted grey line) and the border of die. The idea of pseudo

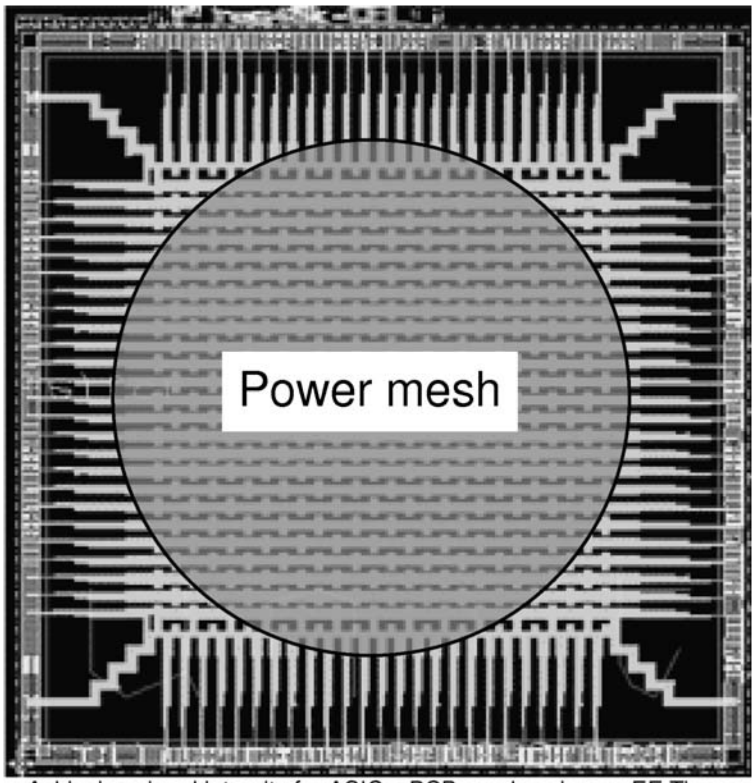

Power mesh
Achieving signal integrity for ASICs, PCBs and packages-EE Times

Figure 2.7: Example of a wide metal PG mesh on an ASIC. The traces running out to the PIOs are for the signal bumps while the metal in the center of the chip is for PG.

single-layer routing avoids cost problems and realize congested routing. While previous works focus on pure single-layer routing, the concept of pseudo single-layer uses 2-layer routing within a small area. This is applicable to RDL due to the routing style in M9.

M9 is traditionally used to connect power/ground (PG) from IO pads to core. Some different styles of PG nets such as rings, stripes, and meshes are therefore designed. Fig. 2.7 shows an example of PG mesh. The most important function of M9 is to evenly distribute power to every logic gate in the core. So the peripheral area of M9 is relatively less important than the central area. This is key observation enables signal nets to treat peripheral area of M9 as routable area.

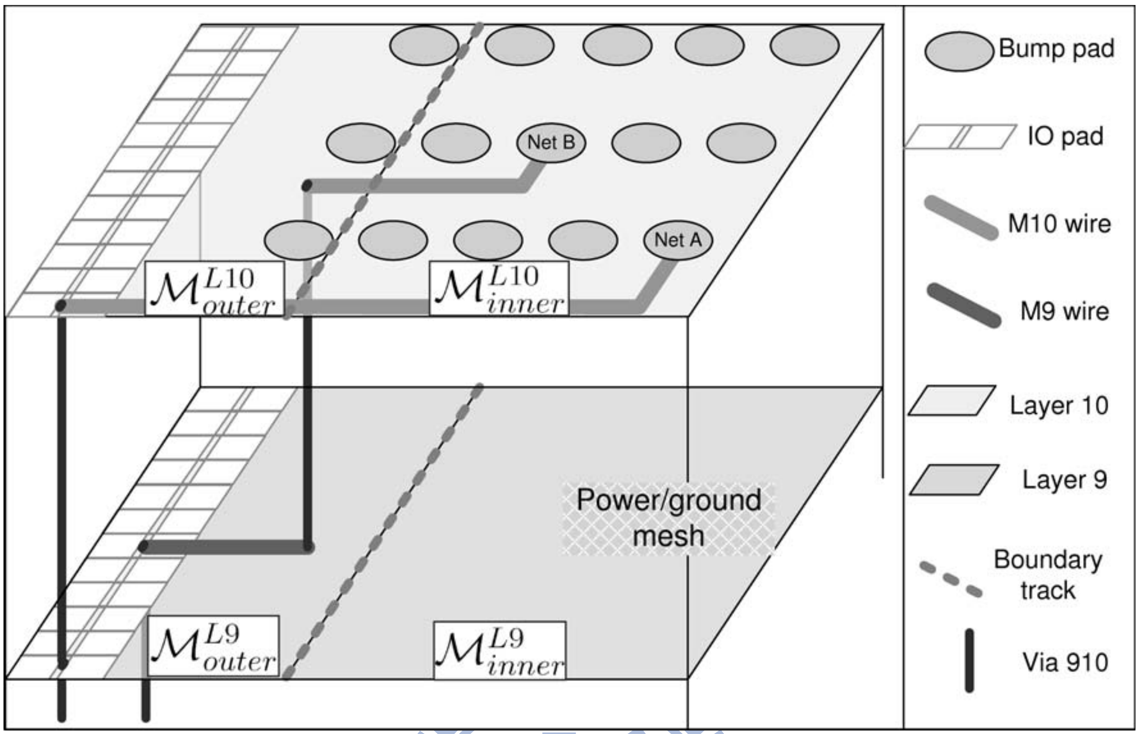

Net B
Net A
$\mathcal{M}^{L10}_{outer}$
$\mathcal{M}^{L10}_{inner}$
Power/ground mesh
$\mathcal{M}^{L9}_{outer}$
$\mathcal{M}^{L9}_{inner}$
Bump pad
IO pad
M10 wire
M9 wire
Layer 10
Layer 9
Boundary track
Via 910

Figure 2.8: Our routing model. First and second RDLs are layer 9 and layer 10 respectively. PG mesh are placed in $\mathcal{M}^{L9}_{inner}$. Routable region is $\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L10}_{inner} \cup \mathcal{M}^{L9}_{outer}$.

2.1.4 Problem Formulation

The problem of RDL routing is to connect net $\mathcal{N}_i$ between the bump pad $\mathcal{B}_i$ and the input/output pad $\mathcal{IO}_i$. First and second RDLs are layer 9 and layer 10 respectively, as shown in Fig. 2.8. We define the area as inner/outer region with respect to the boundary track. Regarding layers and inner/outer, the two RDLs are partitioned into four area: $\mathcal{M}^{L10}_{outer}$, $\mathcal{M}^{L10}_{inner}$, $\mathcal{M}^{L9}_{outer}$, $\mathcal{M}^{L9}_{inner}$.

Definition of Term:

- Routable region: $\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L10}_{inner} \cup \mathcal{M}^{L9}_{outer}$
- Outer region: $\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L9}_{outer}$
- Inner region: $\mathcal{M}^{L10}_{inner} \cup \mathcal{M}^{L9}_{inner}$

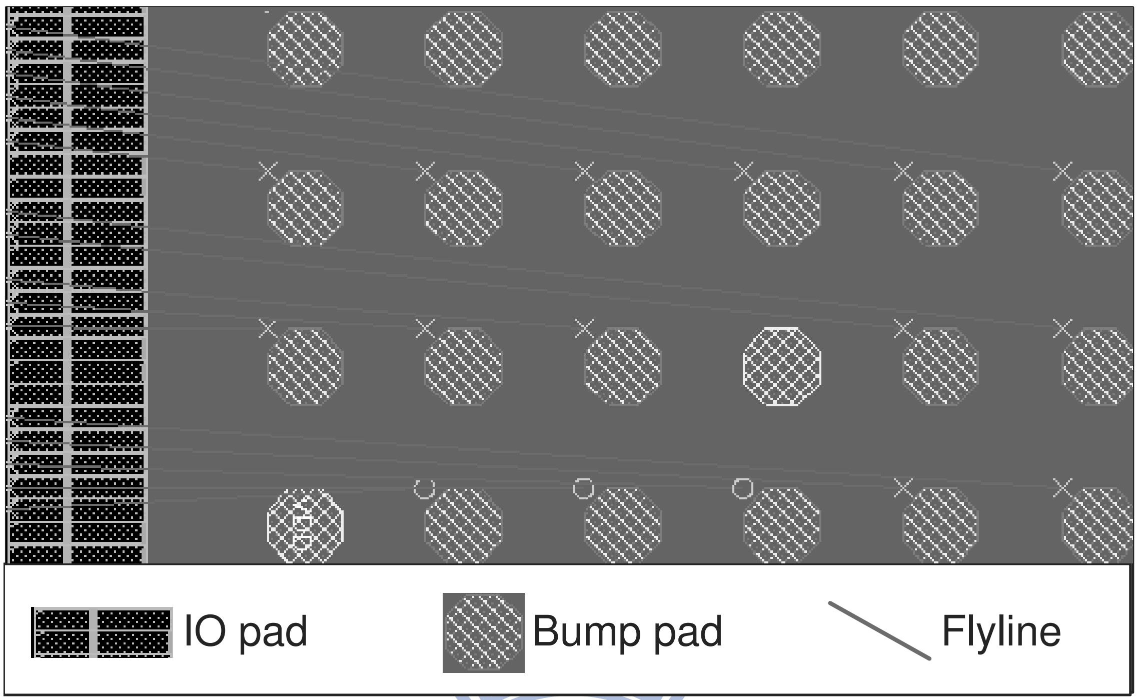


Figure 2.9: Physical locations of bump pads and IO pads are given. The bump-IO mapping is shown in flylines.

The pseudo single-layer RDL routing problem is to physically connect $\mathcal{B}_i$ and $\mathcal{IO}_i$ of net $\mathcal{N}_i$ in routable region and to minimize the outer region. The problem formulation is as follows:

Input:

- Given physical locations of bump pads $\mathcal{B}_i$ and IO pads $\mathcal{IO}_i$
- Given $\mathcal{B}_i - \mathcal{IO}_i$ mapping of net $\mathcal{N}_i$

Output:

- Single-layer routing in $\mathcal{M}^{L10}_{inner}$
- Two-layer routing in $\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L9}_{outer}$

Objective:

- Minimize the area of outer region

The whole area of one layer RDL is divided into four sectors by two diagonals, west, north, east, and south. In the following descriptions, we focus on west region only. In our implementation, north, east, and south regions are counterclockwise rotated 90, 180, 270 degrees respectively.

2.2 Defining Channels for Congested RDL Routing

The modeling of routable region as channels is described in this section. Firstly, some analyses of the region are derived. Secondly, an abstraction from physical layouts to tracks and pins is presented. Lastly, to efficiently utilize routable spaces, regional layer allocation is proposed.

2.2.1 Constraints and Considerations

Here we address the constraints and considerations observed from manual routes. The example shown in Fig.2.10 is a real situation from industrial case. There are few crossings due to a well devised bump-IO assignment. There is sufficient capacity for horizontal wires because the row can support more capacity than six wires. In Fig.2.10(a), all nets of the row can be routed from bump pads to IO pads. However, capacity for vertical wires is insufficient because 24 nets (4 rows $\times$ 6 bumps per row) travel through the horizontal cut line, in Fig.2.10(b), only 12 flylines (out of 24 nets) are plotted. The capacity allowed is small so that there is no room for 24 vertical wires.

In Fig. 2.8, there is only limited area for routing in layer 10 ($\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L10}_{inner}$). For example, when $\mathcal{N}_A$ is considered for routing, other bumps are seems as obstacles. If we restrict nets to route within its own horizontal channel, the route from $\mathcal{B}_A$ has

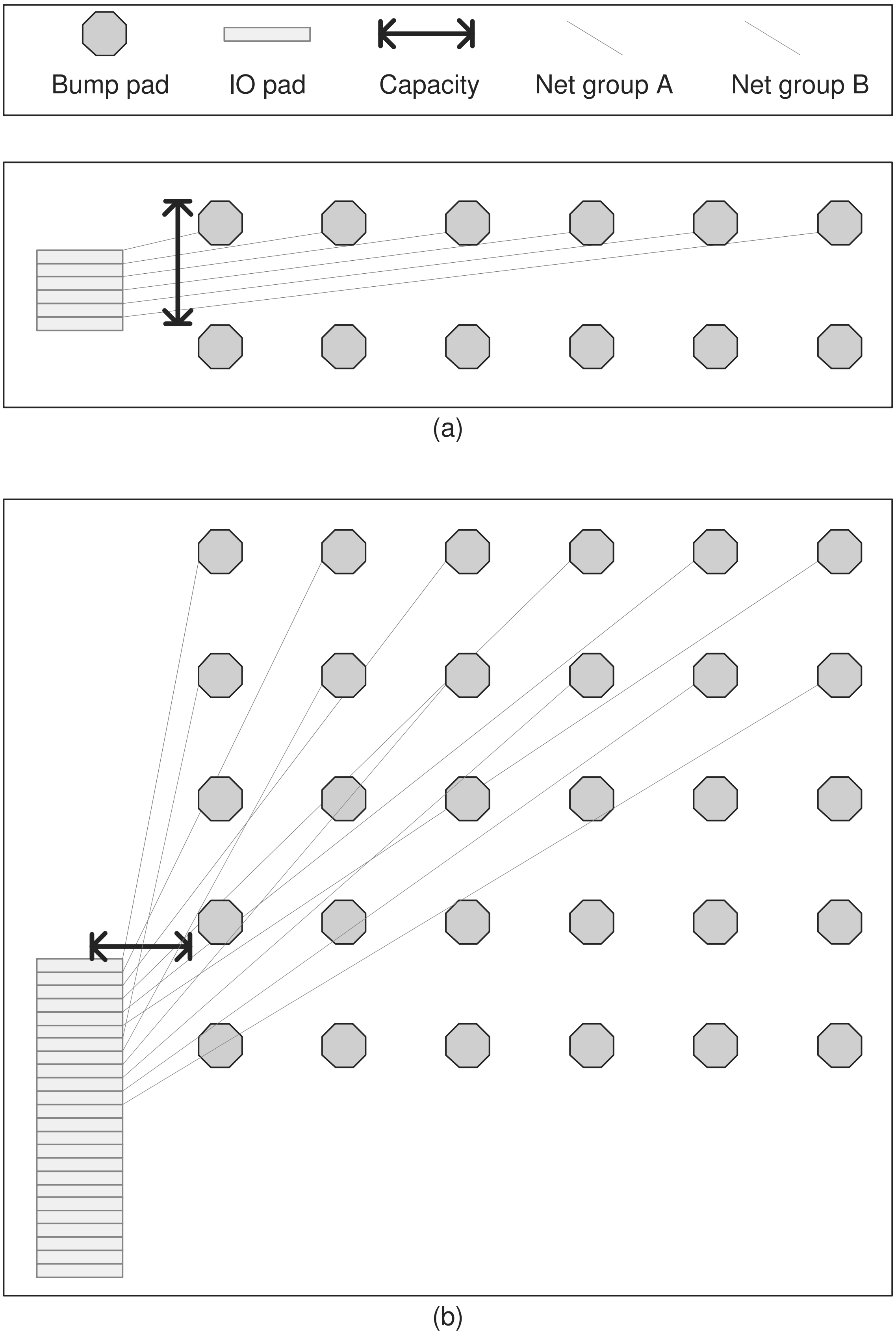


Figure 2.10: Capacity constraints for horizontal wires (a) and vertical wires (b).

no choice but go left (toward I/O pads) directly. Note that we can only use inner region of M10. The area for routing in layer 9 is along the peripheral region ($\mathcal{M}^{L9}_{outer}$). It is small because it is borrowed from routing space of PG nets. It is clean because signal nets have higher priority than PG nets in the design flow shown in Fig. 1.1. The only implicit constraint is that there must be some spaces for PG IOs to go to central area and to connect with PG mesh. This is important so some horizontal spaces in M9 must be reserved for PG routes.

2.2.2 Modeling of RDL Routing

In west region, bump pads are on the right and IO pads are on the left, as shown in Fig. 2.10(b). The placement of IO pads is in a vertical column. Wires of the innermost (rightmost) bumps generally escape horizontally to reach their corresponding IO pads. We define horizontal direction as x-axis and vertical direction as y-axis. Modeling of RDL routing is critical to the entire routing process. Abstraction from physical layout to our model can enable a direct application of classic channel routing algorithm. This is the major advantage of our model and will be explained in the next section.

A route from an IO pad or a bump pad is composed of more than one wire segment. It is assumed that wires can go either horizontally or vertically, although 45-degree wires are allowed in RDL. Thus, a wire segment can contain information in only one direction if it starts at a given point. For example, if a horizontal wire starts at $(3, 5)$, then it must end at $(3 + dx, 5)$. This wire only possesses one information, dx. A track is a vertical line where x-coordinate is fixed. A pin is a point on a track. To locate a pin, we only have to locate a y-coordinate because the x-coordinate is inherited from its track. Based on this definition, minimizing the x-coordinate of boundary track is equal to minimizing the area outer region, which is our objective in problem formulation. Recall that the objective is to minimize the area of outer region, and the boundary track is the cut line between inner and outer region. The x-coordinate of IO track is the x-coordinate of every IO pad because IO pads are assumed to be in a column and uniform in size. Nets must travel through tracks on their left to reach IO pads. Pins are the points where wires travel across tracks. In the example shown in Fig. 2.11, there are IO track, boundary track, and 6 bump tracks (Ly1, Ly2, Ry1, Ry2, Ry3, Ry4).

2.2.3 Regional Layer Allocation for Effective Capacity Utilization

For outer region, two layers are both available. To effectively utilize routing capacity, two solutions for routing resource allocation are applied, shown in Fig. 2.12. Regional layer allocation is proposed as follows. One layer is for horizontal wires and the other one is for vertical wires. Horizontal wires are used to connect from IO pads to the assigned track. They cannot be blocked by any obstacle. Thus, the allocation must have horizontal wires routed with M9 based on the observation that M10 is full of bumps. Another advantage is that PG wires go horizontally to inner region using M9. This allocation can leave some spaces for PG wires. Vertical wires are therefore used to connect two pins on a track. An example of layer assignment is shown in 2.12(b). One net blocks the PG net (arrowed-dotted) and another net is not routed (dotted).

2.3 Successive Channel Routing Algorithm

The proposed routing algorithm divide the routing process for a net into three steps. *Step*1 is to route from bump pad to a pin. *Step*2 is to decide which track is to use. *Step*3 is to route from IO pad to the pin. Solving the problem in Fig. 2.12(c) is similar to channel routing problem. In channel routing problem there are channels and two sequence of pins. In Fig. 2.12(c) , the vertical area between bumps are seen

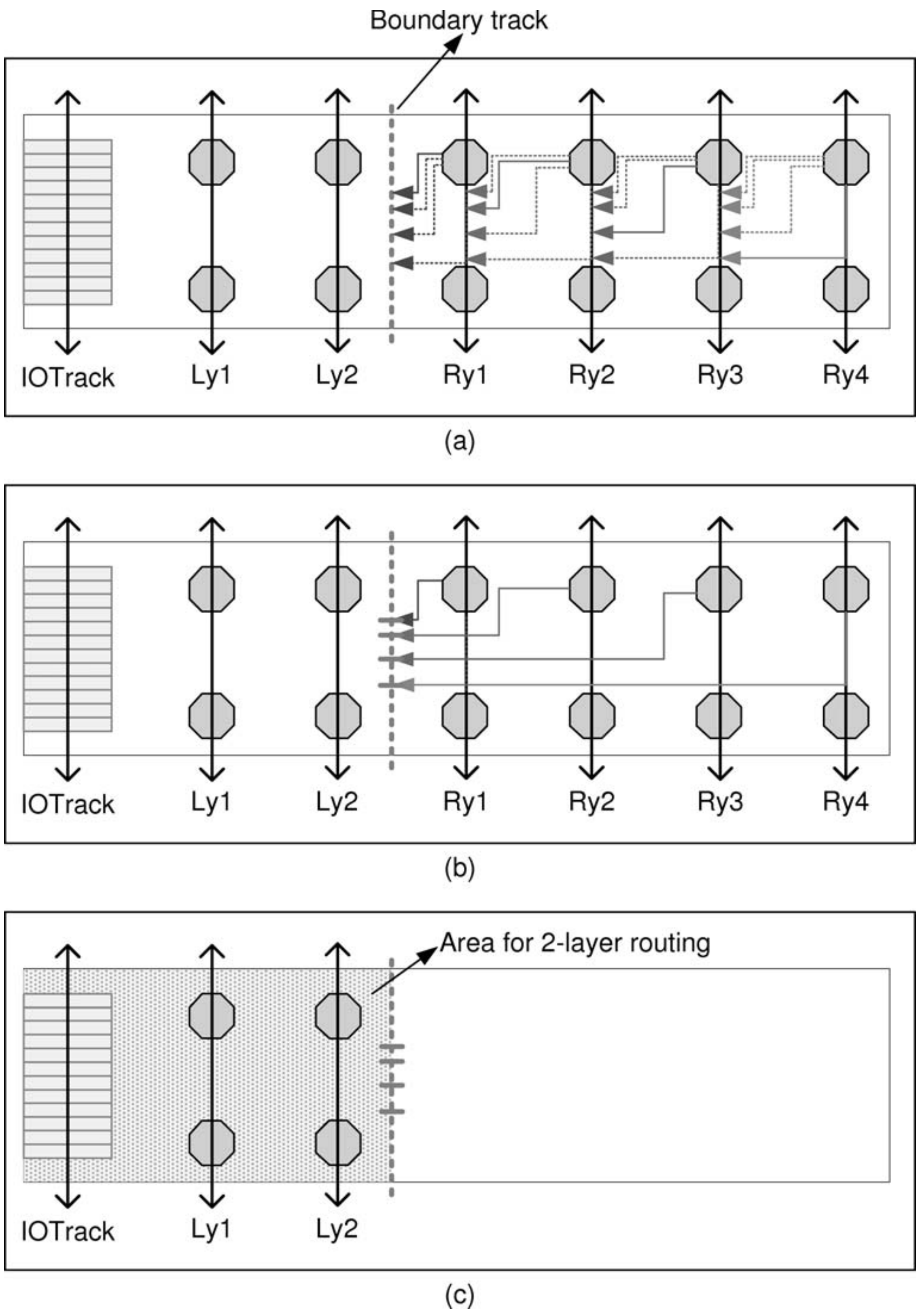


Figure 2.11: Modeling of layout. There are IO track, boundary track, and 6 bump tracks (Ly1, Ly2, Ry1, Ry2, Ry3, Ry4). In (a), on the right of boundary track, each bump track has some possible paths (dotted lines), and one of paths is routed (solid lines). In (b), routes are extended onto the boundary track. In (c), area for 2-layer routing ($\mathcal{M}^{L10}_{outer} \cup \mathcal{M}^{L9}_{outer}$) contains two bump tracks and is enclosed by IO track and boundary track.

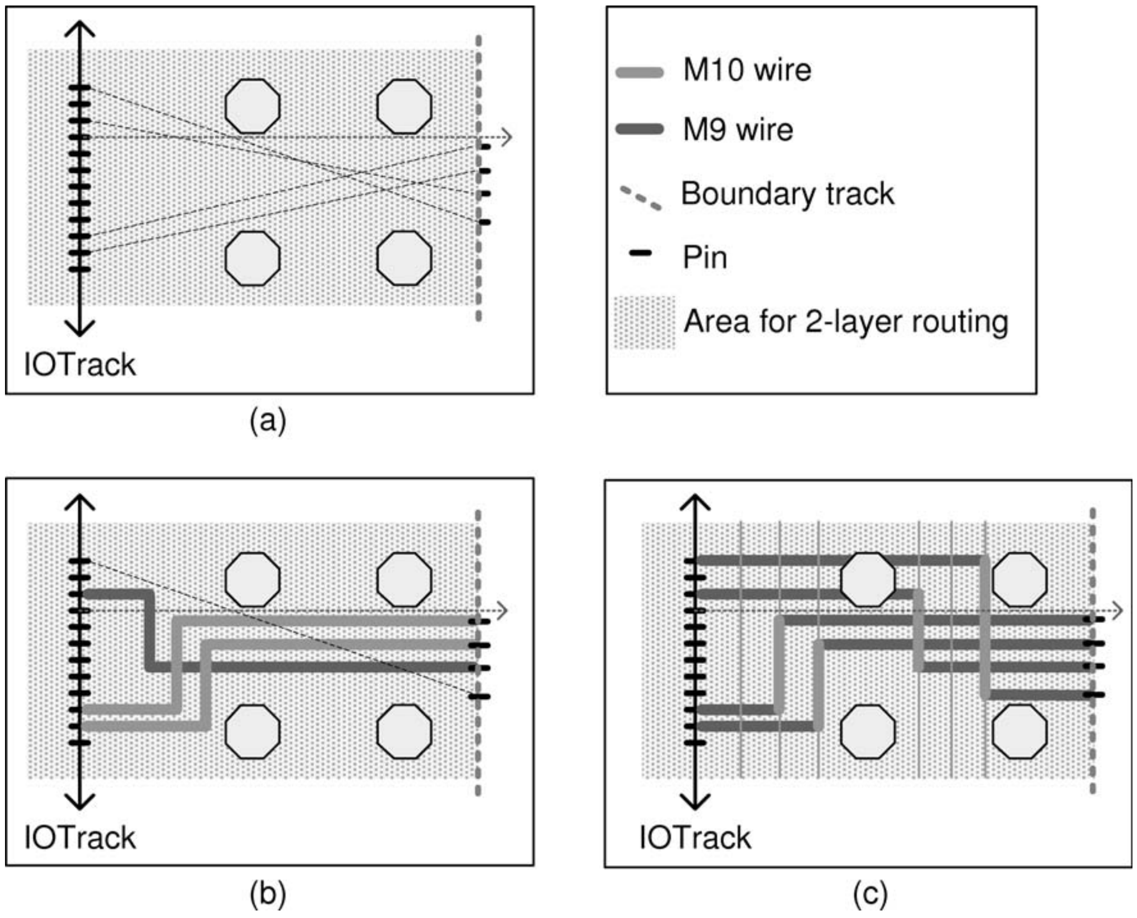


Figure 2.12: Solutions for routing resource allocation. The example shown in (a) contains 4 nets. Layer assignment, as shown in (b), has some drawbacks and limitations. To resolve them, regional layer allocation is proposed, as shown in (c). One PG net arrowing toward the right is to indicate that some spaces must be reserved for PG nets in M9.

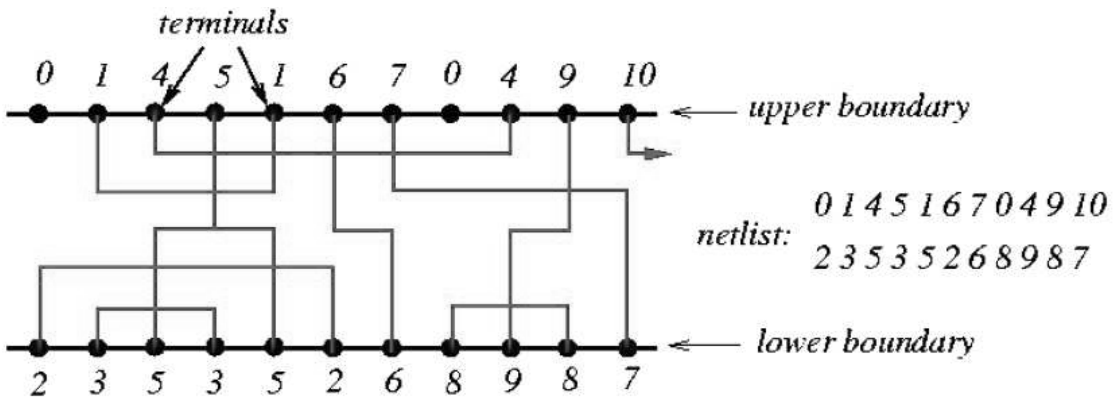


Figure 2.13: Classic channel routing. An example of channel routing which is obtained by left-edge algorithm (LEA).

as channels; IO track and boundary track are seen as two sequence of pins

2.3.1 Classic Channel Routing Algorithm Review

Here the classic channel routing algorithm is reviewed, left-edge algorithm (LEA) is used for routing and to find minimum number of tracks. LEA composes of 3 steps: 1. Build vertical constraint graph (VCG) 2. Place vertical segments. For the nodes who have no ancestor, place them, then update VCG. 3. Repeat track by track. This is a well-known method [21], we simply put an example and its results in Fig. 2.13 for reference. Recall that the results obtained from LEA is optimal if no vertical constraint exists.

2.3.2 Channel Routing on Successive Routing Channels

Abstraction from tracks to channel routing problem is shown in Fig. 2.14. First, all bump pins are collected into one track called virtual track. Then LEA is applied between IO track and the virtual track. Then minimum number of required tracks is obtained. For all tracks from left to right, they defined as $IOtrack$, $track_{1,2,3}$, $Ly1$, $track_{4,5,6}$, and $Ly2$ respectively. For the track $track_i$, $i = 1 \sim 6$, they are allocated 2 layers. $Ly2$ is the boundary track and is defined as boundary of inner/outer region.

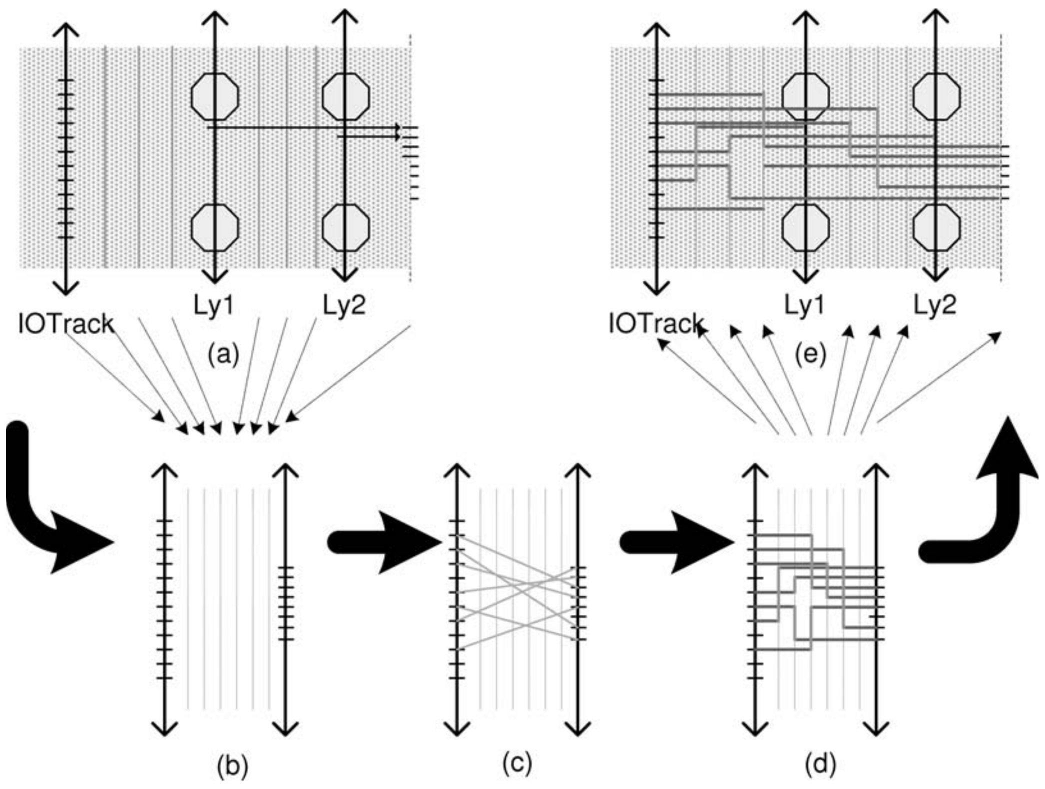


Figure 2.14: Channel routing on successive channels. Abstraction from physical layout to channel routing is shown in (a). For the example shown in (b), the results shown in (c) are obtained by applying LEA and physically routed.

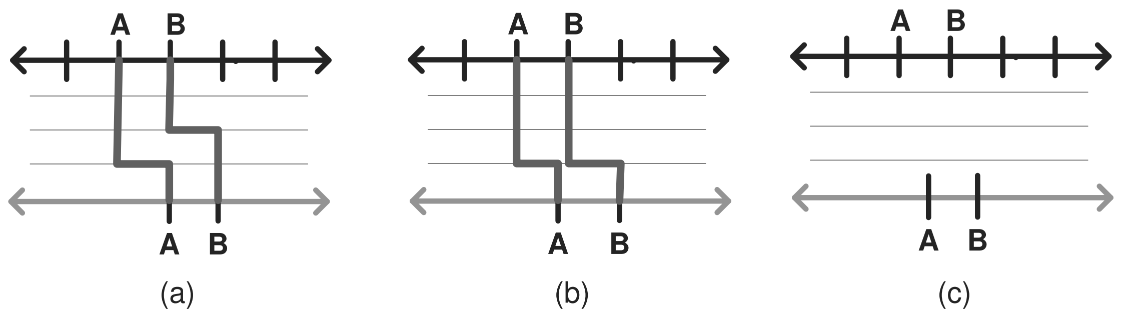


Figure 2.15: Impact of pin locations. A vertical constraint (VC) exists between $\mathcal{N}_A$ and $\mathcal{N}_B$ in (a). It can be relieved by a slight shift of pin_A, as shown in (b). The idea of staggered pins is to stagger all pins, as in (c), on each track so that no VC exists.

The results are mapped onto layout once the results are obtained.

2.3.3 Assignment of Movable Pins on Pin-Tracks for VC Reduction

In *Step*1, pins are movable along track within certain distance. As shown in Fig. 2.15, pin_A is movable if capacity the bottom track is sufficient. We can assign locations of the pins once they are movable. The impact of locations of pins is huge. As shown in Fig. 2.15(a), minimum number of tracks for this case is two because the pin_B on the top track (IO-pin) and the pin_A on the bottom track (bump-pin) have vertical constraint. However, if pin_A is moved to its left, then minimum number of tracks is reduced to 1, as shown in Fig. 2.15(b). So the assignment of movable pins directly determines the number of required tracks. If they are well assigned, vertical constraints can also be greatly reduced.

Based on the observations from design parameters, the idea of *staggered pins* is proposed to assign movable pins. Observations in TSMC 90nm process:

1. Wire width = 10um

2. Wire spacing = 2.5um

3. IO pad height = 30um

There is enough capacity to place one horizontal wire at each border of IO pads. So bump-pins can be staggered from IO-pins, as shown in Fig. 2.15(c). Besides, once all pins are staggered, there does not exist any vertical constraint.

2.4 Experimental Results

The proposed algorithms are performed on a real industrial case. The whole chip is divided into four sectors: W, N, E, and S. Each sector contains more than a hundred signal bumps.

We implemented the algorithm in tool command language (Tcl). The data are fetched from the design in Encounter Digital Implementation (EDI). This preprocessing generates the input of our algorithm. For each sector, our algorithm can generate the results and dump scripts of commands in less than 5 seconds. By sourcing these scripts in EDI, wires are physically routed. All results are clean in design rule checking (DRC). Our router is now under examination for internal use of the company in collaboration. The experimental results are shown in Fig. 2.16 and Fig. 2.17 [1]

2.5 Summary

We introduce algorithms for RDL routing on pseudo single-layer in this chapter. The designs under consideration are so congested that even manual routing cannot find solutions within single-layer. The concept of pseudo single-layer is then proposed. We have shown that it provides an acceptable solution other than adding an extra metal layer or increasing the die-size.

We addressed regional layer allocation, assignment of movable pins, and layout abstraction. These techniques transform the RDL routing problem into classic chan-

[1] Due to non-disclosure agreement (NDA), only partial of the results are shown.

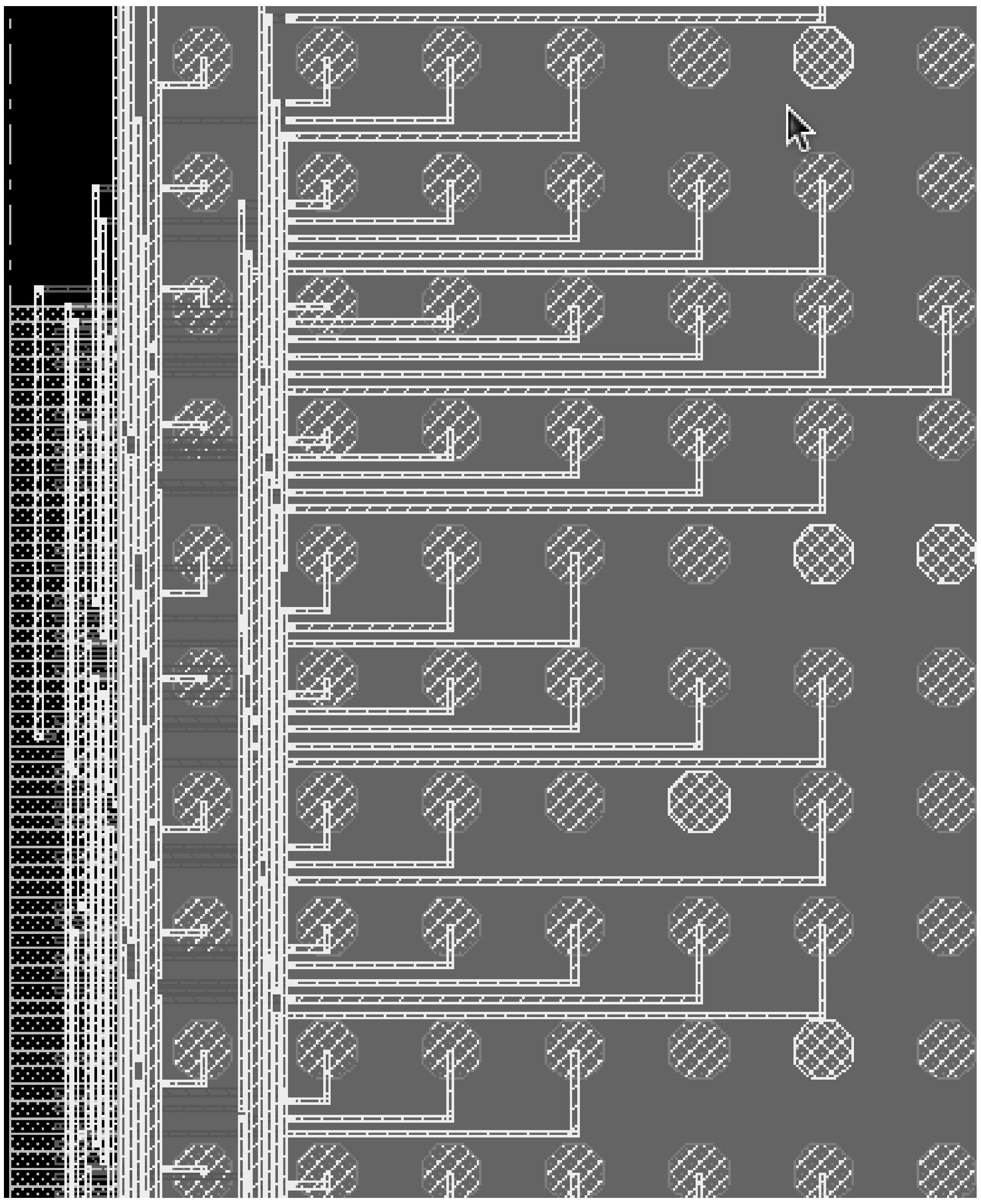


Figure 2.16: Partial results of sector-W, M10 and M9 wires are in yellow and red respectively.

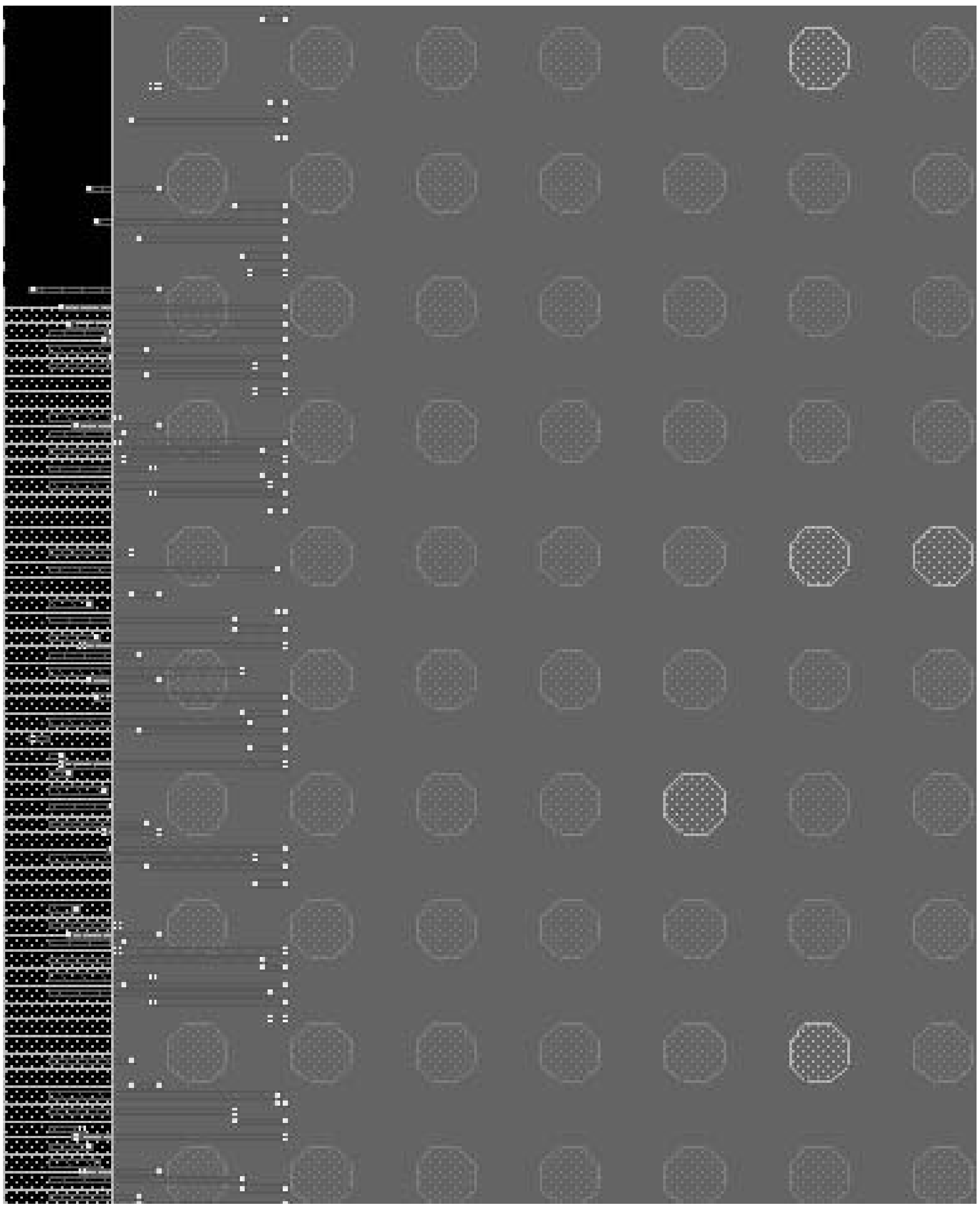


Figure 2.17: Only M9 wires in Fig. 2.16 are shown.

nel routing problem. By simply applying left-edge algorithm (LEA), 100% wires are routed and the area of 2-layer routing is minimized. Recall that LEA minimizes the number of routing tracks.

Comparisons of routing results from manual design, commercial tool, and the proposed method are provided. For the real industrial case, commercial tools cannot even initiate routing process. Manual routing can reach 100% routability but the designers in collaboration spent about one month to achieve that. However, our proposed method can finish RDL routing automatically and very effectively.

Chapter 3

Fast Package Pin-out and Wire Planning in Chip-Package-Board Codesign

Slow turn-around between design, package and system houses has been one of the primary concerns in semiconductor business. This is seriously lagging the development time of the system due to time-consuming interface design between chip, package and board. In order to enable chip-package-board codesign to speedup the design process, we propose an approach to address this issue by fast estimating the resources we use during designing the interface, which includes the package pin-out designation and corresponding wire planning in package and board.

We model the problem as an interval intersection problem. Due to the special need in pin-out rules, we have developed an algorithm to resolve the problem. We then use some optimization techniques to further improve the objectives such as global wire congestion and length deviation. Our results show that we can perform a very efficient estimation considering those important objectives, even outperforming one recent related work.

3.1 Overview

Nowadays larger gaps are emerging between *chip*, *package*, and *board* designs. More resources are spent on reaching a consensus between these three interfaces. Chip-Package-Board codesign targets at better system performance and shorter design cycles. It efficiently facilitates achieving a convergent solution. Fig. 3.1 shows the example of the whole platform: signals starting from I/O pads travel through many interfaces including RDL bumps, package balls, and printed circuit board (PCB). In modern VLSI designs, more than a thousand I/O pins are usually required for communicating each other. Due to the demand for more I/Os, ball grid array (BGA) packaging has become a major interface between chip and PCB. Trade-offs between system performance and cost are therefore determined by BGA pin-out designation (also called ballout).

[12] proposed an efficient approach to automate pin-out designation for package-board codesign. Their frameworks consider signal integrity (SI), power delivery integrity (PI) and routability (RA) in pin-out block design, and achieve close-to-minimum package size while providing good signal quality. However, more requirements should be further fulfilled in pin-out designation, in addition to the performance metrics mentioned above, to facilitate the routing works between chip, package and PCB.

3.1.1 Previous Works

Regarding the flip-chip designs, it is generally classified into two regimes. One is called peripheral-array I/O (PIO) where bumps are placed along the chip boundary. The other one is called area-array I/O (AIO) where bumps are placed in central area of the chip [13]. Since AIO accommodates much more number of bumps than PIO, it is more suitable for modern VLSI designs.

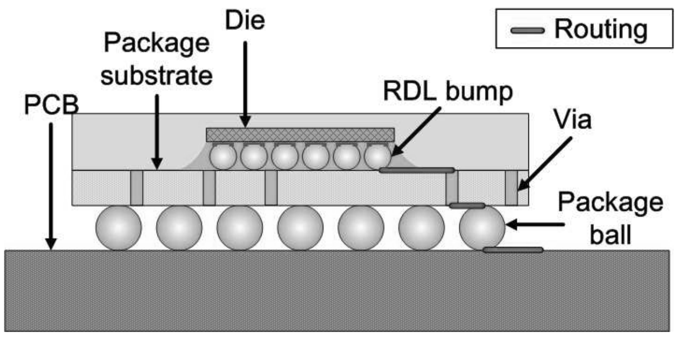


Figure 3.1: The cross section of the platform: signal trace traveling through three interfaces including RDL bumps, package balls, and PCB.

For AIO flip-chip designs, some sophisticated redistribution layer (RDL) routing methods are developed to connect the peripheral I/O pads with area-array bump pads. According to the pre-assigned order of I/O pads, Fang *et al.* applied the network-flow-based [13] and the integer-linear-programming-based [14] RDL routing algorithms for designing area-array ICs. Each of these two-stage techniques not only completes 100% routability but also reduces the total RDL wirelength and the signal skews compared with an industrial heuristic algorithm. Consequently, in order to preserve the optimized results in RDL routing, the pin-out designation must follow the ordered I/O pin sequence while designing package.

On the other hand, considering the PCB routing problem, it can also be divided into two categories. One is escape routing, which routes nets from pin terminal (ball) to component boundaries. The other one is area routing, which routes nets between component boundaries [15]. For area routing, the planar-fashioned bus routing is always preferred to control and match impedance for each high-speed signal. One approach regarding automatic bus planner for PCB was published very recently in [16]. On testing a state-of-the-art industrial circuit board, their bus planner achieves 98.5% routing completion and simultaneously assigns routing layers and nets.

However, the basic requirement of this bus planner is ordered escape routing which routes nets from balls to component boundaries with a given order. Without ordered escape routing, it is not guaranteed that the planar bus routing between components can be done [17]. To achieve the ordered escape routing, the given I/O pin sequence must be carefully considered when designating the package pin-out.

3.1.2 Our Contributions

The common approach usually takes weeks to rearrange pin-out, rework package substrate and PCB layout, as shown in Fig. 3.2, each modification of interfaces can bring costly iterations. For chip core designers, the iterations of modifying I/O pads and RDL bumps with system designers takes at least one month. We hope to have a fast estimation on the resources we can use in package and board, to skip long turn-around time and iterations between design house, package house and system house. This thesis proposes a feasible pin-out designation which considers the ordered pin sequence in both die side and package side. These ordered pin sequences are passed to die RDL routing and PCB area routing, which are optimized by using previous works [13, 14, 15, 16, 17]. In other words, core designers can specify the preferred I/O pad ordering; system designers can specify the preferred bump pin-out designation. Our method can efficiently analyze if the preferences from both sides accommodate each other, before performing RDL routing and substrate routing. Thus the flow can be replaced by the proposed methodology, as shown in Fig. 3.3.

The rest of the thesis is organized as follows. Section II defines the problem of wire planning for 2-layer package design and PCB escape routing considering the ordered pin sequence. Section III describes the package ballout and wire planning approach; Section IV shows the optimization for various objectives to further strengthen our methodology. Section V shows the experimental results, followed by conclusion in Section VI.

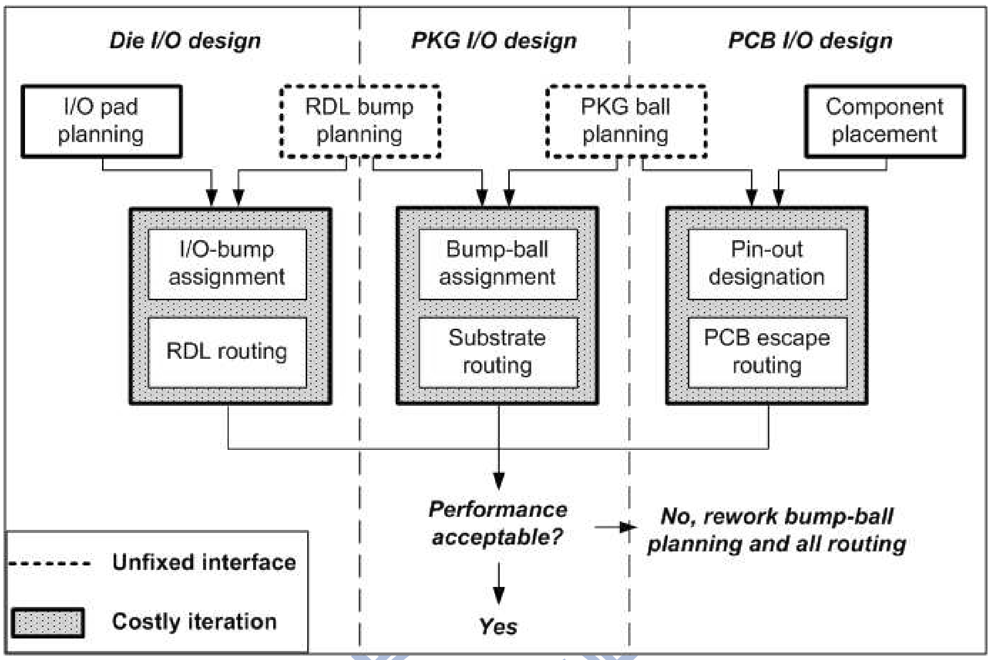


Figure 3.2: Conventional flow suffers from costly rework and slow turn-around.

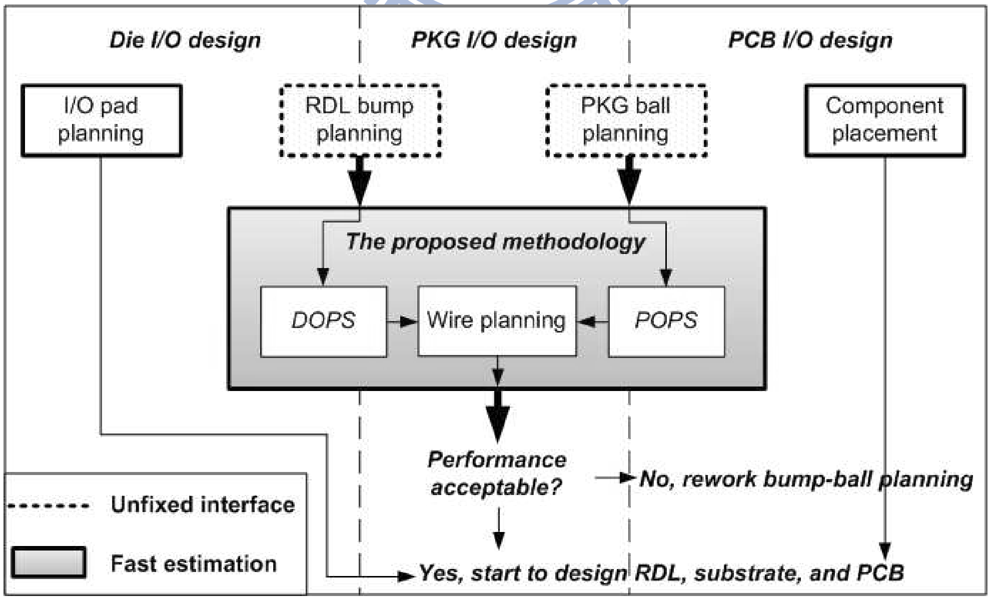


Figure 3.3: The proposed flow enables fast chip-package-board codesign respin.

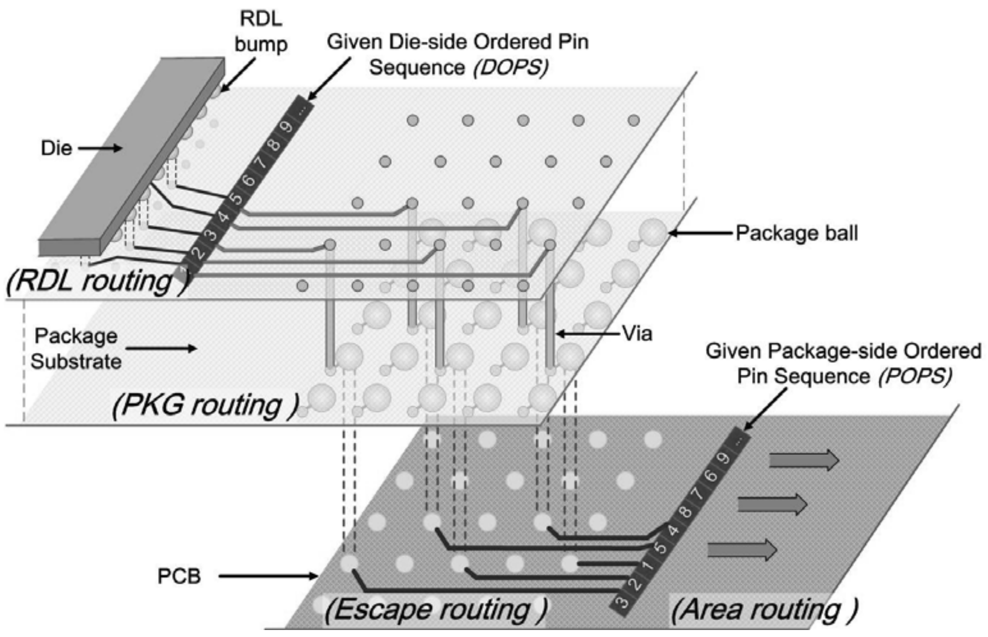


Figure 3.4: Wires/traces are routed in the 2-layer BGA package model.

3.1.3 Problem Formulation

Fig. 3.4 shows our 2-layer BGA package model. Die-side ordered pin sequence ($DOPS$) and package-side ordered pin sequence ($POPS$) are the orders of I/O pins in both sides. $DOPS$ serves as input of RDL bump assignment. The corresponding RDL routing can be optimized by applying the network-flow-based [13] or the integer-linear-programming-based algorithms [14]. $POPS$ is regarded as input of PCB bus planner. The corresponding PCB area routing with planar bus can be well solved by [16].

In this 2-layer package model, each net (denoted as n_i), starting from $DOPS$, is connected to a via (denoted as v_i) on layer-1. Each via connects to exactly one ball (denoted as b_i) on layer-2. Each ball then connects to $POPS$ using PCB escape routing. In our initial assignment, v_i and b_i are tied up as one pin (denoted with p_i)[1] and will be loosened in the post-optimization.

[1]In this work, we consider one signal net n_i through package and board (with assignment v_i and b_i) as pin (p_i). Therefore we use pin and net interchangeably.

The ballout/pin-out designation problem is to assign n_i to v_i and b_i and to generate the corresponding wire planning from given $DOPS$ and $POPS$. The formal definition is defined as follows:

Input:

- Given two sequences:
  - Die-side ordered pin sequence ($DOPS$).
  - Package-side ordered pin sequence ($POPS$).

Output:

- Ballout/Pin-out designation for 2-layer BGA package.
- The corresponding wire planning (monotonic global routing) for package design and PCB escape routing.

Objectives:

- Minimize package size (can be seen as the total number of columns used, referred to Fig. 3.5).
- Minimize wire congestion.
- Minimize wirelength variation/deviation for each routing layer.

There are 6 rows and 5 columns in the example shown in Fig. 3.5. The row number counts from top to bottom and the column number counts from left to right. For example, the locations of p_1 and p_4 are (row 3, col 5) and (row 1, col 1) respectively. Note that each route on package layer-1 is composed of two segments: 1^{st} layer routing and 1^{st} layer plating lead. A plating lead is redundant for operation and is usually used to reduce fabrication cost [18]. Plating leads are not plotted in

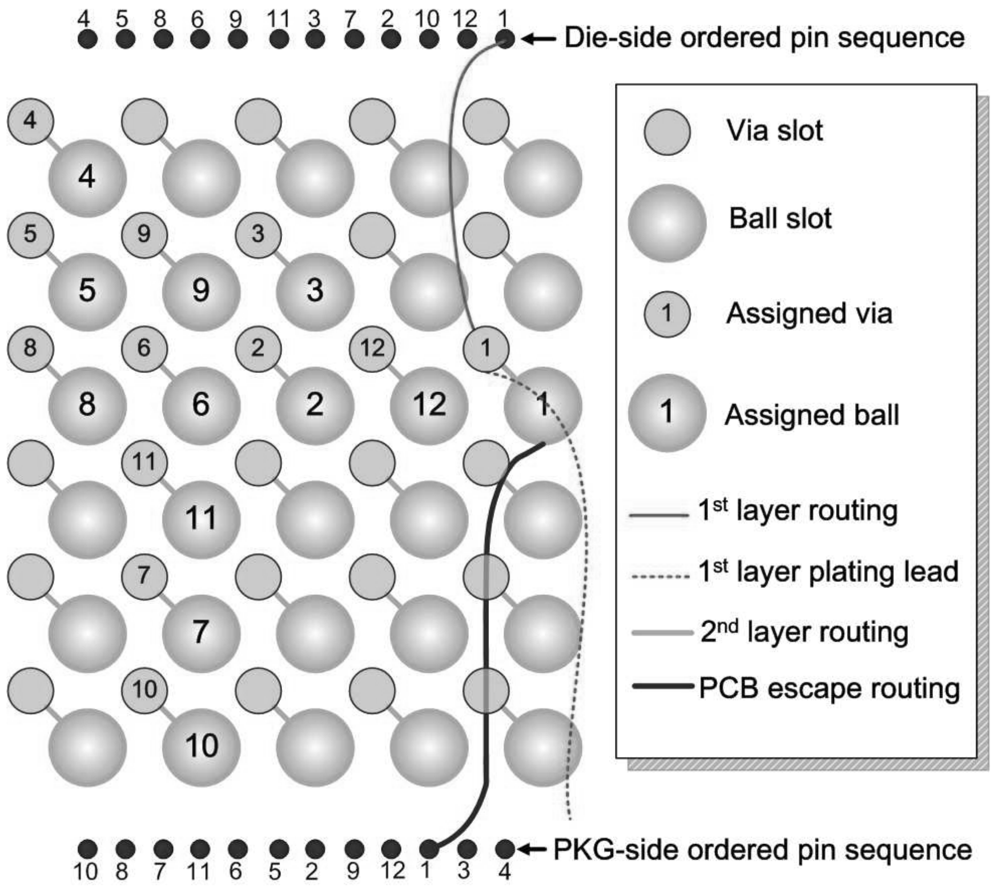


Figure 3.5: Problem Illustration. Each net is designated to a via and a ball; the corresponding wire planning of n_1 is plotted. v_i and b_i are tied up to find initial solution. The numbers inside via and ball slots are initial solution for these two ordered pin sequence.

the following figures, however, they are evaluated as normal wires in cost evaluation (shown in Section IV.C).

3.2 Package Pin-out and Corresponding Wire Planning

3.2.1 Monotonic Global Routing in Wire Planning

A route is called monotonic if it only intersects every horizontal grid line once. Fig. 3.6 shows eight routing scenarios of 2-layer package routing and PCB escape routing, only (a) and (e) are not monotonically assigned. [18] [19] have shown that the routing on the package layer-1 are monotonic when vias are monotonicly assigned. Following the same idea, when balls are designated to be monotonic, the PCB escape routing is monotonic. In Fig. 3.6, three nets (n_1, n_2, and n_3) are assigned in eight different patterns. $DOPS$ connects via on the first layer of package and $POPS$ connects balls on PCB. $DOPS$ is given as 1, 2, 3 and $POPS$ is given as 2, 1, 3, in which the order of n_1 and n_2 are reverse. They are called *intersected nets* since their flylines intersects with each other.

Based on these scenarios, we can define the general rule of thumb for designating package pin-out and completing the monotonic global routing. Take Fig. 3.6(a) as an example, p_1 (pin/net 1; with via v_1 and ball b_1) is on the left of p_2, this order is consistent with $DOPS$ but inconsistent with $POPS$. When the designated column number of n_1 and n_2 is not in the same order as in $DOPS$ or $POPS$, that will cause the routing intersection during package layer-1 routing or PCB escape routing. To route without intersection, as shown in Fig. 3.6, different pin-out assignments will produce different routing results. These results are summarized as follows: (row_i means the row number of p_i)

- ***Case 1*** ($row_1 = row_2$, $column_1 \neq column_2$):

In this case, p_1 and p_2 are located at the same row. To solve the routing intersection, the package layer-1 routing or PCB escape routing must be non-monotonic (see Fig. 3.6(a) and (e)).

- ***Case 2*** ($row_1 \neq row_2$, $column_1 \neq column_2$):

  In this case, the assignment of p_1 and p_2 can produce the monotonic routing. However, these pins will possibly be routed through more than one routing track. [2] (see Fig. 3.6(b), (d), (f) and (h)).

- ***Case 3*** ($row_1 \neq row_2$, $column_1 = column_2$):

  In this case, p_1 and p_2 are located at the same column. For both the package layer-1 routing and PCB escape routing, the routing results not only are monotonic but also use only one routing track. (see Fig. 3.6(c) and (g)).

According to these scenarios, the pin-out designation rules are defined below:

- ***Rule 1: To achieve monotonic routing***

  1. *the pins corresponding to intersected nets must not be assigned at the same row, or*
  2. *the designated column number of these pins corresponding to intersected nets must be in the same order as in both DOPS and POPS.*

- ***Rule 2: To minimize the routing space***

  1. *the pins corresponding to intersected nets must be assigned at the same column, and*
  2. *the designated row number of these pins corresponding to intersected nets must be adjacent.*

[2]For the package first layer routing, routing track is the routing space between two column of vias. For PCB escape routing, that is the space between two column of balls.

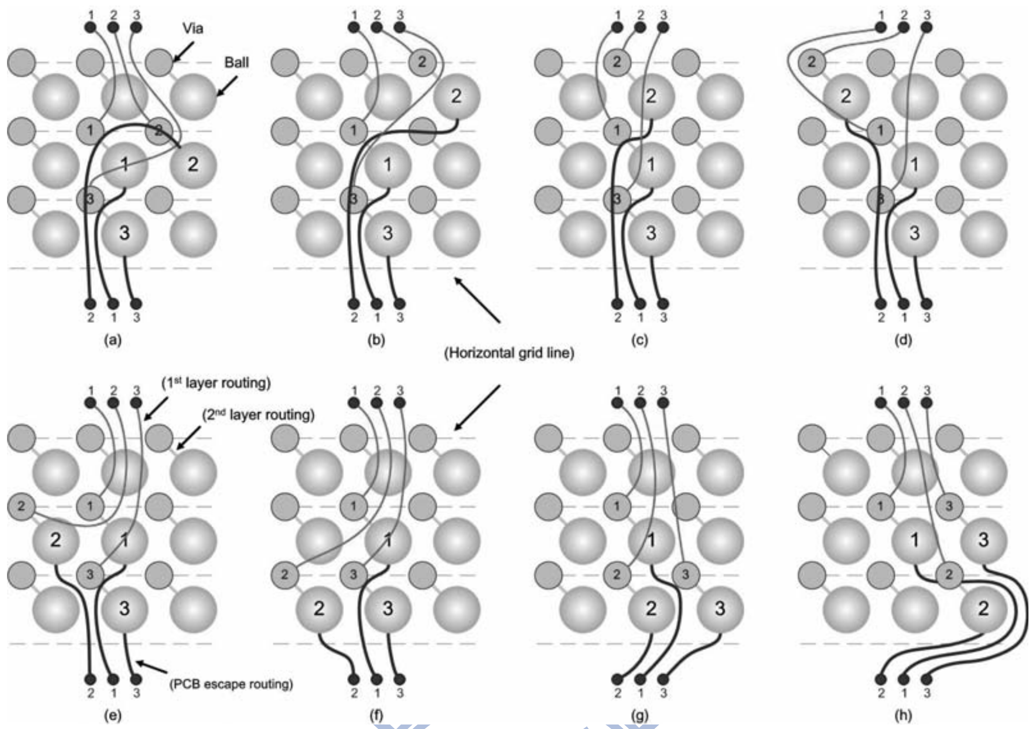


Figure 3.6: Routing scenarios which are produced by the pins corresponding to intersected nets designated with different ways.

In order to designate package pin-out efficiently and achieve the monotonic global routing for package design and PCB escape routing, the proposed methodology is to find the intersected relationship between nets by using an intersection graph. The pin-out assignment based on the rule of thumb listed above can be satisfied by applying the proposed intersecting relationship analysis.

3.2.2 Pin-Out Designation Methods for Wire Planning

Interval diagram, which analyzes the intersection relationship of nets, is used to produce intersection graph. In [20], they proposed a method using inversion table to analyze the orderings of two sequences.

The interval diagram shown in Fig. 3.7 shows the intervals of nets. For each net n_i, its corresponding interval I_i is composed of a start point s_i and a destination point d_i. s_i and d_i are represented by a small solid circle and an arrow respectively and they are determined by the index of n_i in $DOPS$ and $POPS$ respectively.

Interval-Scan Algorithm, which transforms the interval diagram into the intersection graph, is described below. Intersection graph is defined as G_I=(V, E_I) and plotted in Fig. 3.8. V= $\{vt_i | vt_i$ represents interval $I_i\}$. Two vertices are connected by an edge if and only if their corresponding nets intersect each other.

Interval-Scan Algorithm:

```
 1. $i \leftarrow 1$, $j \leftarrow 2$
 2. **Repeat:**
 3. select net $n_i$ from $DOPS$ and scan $I_i$
 4.   **Repeat:**
 5.   select net $n_j$ from $DOPS$ and scan $I_j$
 6.   **IF** ($I_i$ and $I_j$ go same direction and $I_i$ covers up $I_j$)
 7.   **Then** build an edge between $vt_i$ and $vt_j$
 8.   **IF** ($I_i$ and $I_j$ go opposite direction and $I_i$ overlaps $I_j$)
 9.   **Then** build an edge between $vt_i$ and $vt_j$
10.   increase $j$
11.   **Until** $I_i$ has scanned every $I_j$
12. increment $i$
13. **Until** all nets are selected
```

We have the following lemma:

Lemma 3.2.1. *If there exists an edge between two vertices, then the child should be placed at the parent's row + 1.*

Once we have the intersection graph, the initial pin-out designation can be produced by using a simple algorithm based on the pin-out designation rules. The detailed processes are shown below:

Initial Pin-Out Designation Algorithm:

```
1.  $i \leftarrow 1$, $j \leftarrow 2$
2.  select net $n_i$ in $DOPS$, assign $row_i = 1$, $column_i = 1$
3.  Repeat:
4.  select net $n_j$ in $DOPS$
5.  IF an edge exists between $vt_j$ and $vt_i$, Then
6.    assign $row_j$ based on Rule 2
7.    assign $column_j$ based on Rule 1
8.  ELSE
9.    assign $row_j = row_i$
10.   assign $column_j$ based on Rule 1
11. $i \leftarrow j$
12. increment $j$
13. Until all pins in $DOPS$ have been assigned
```

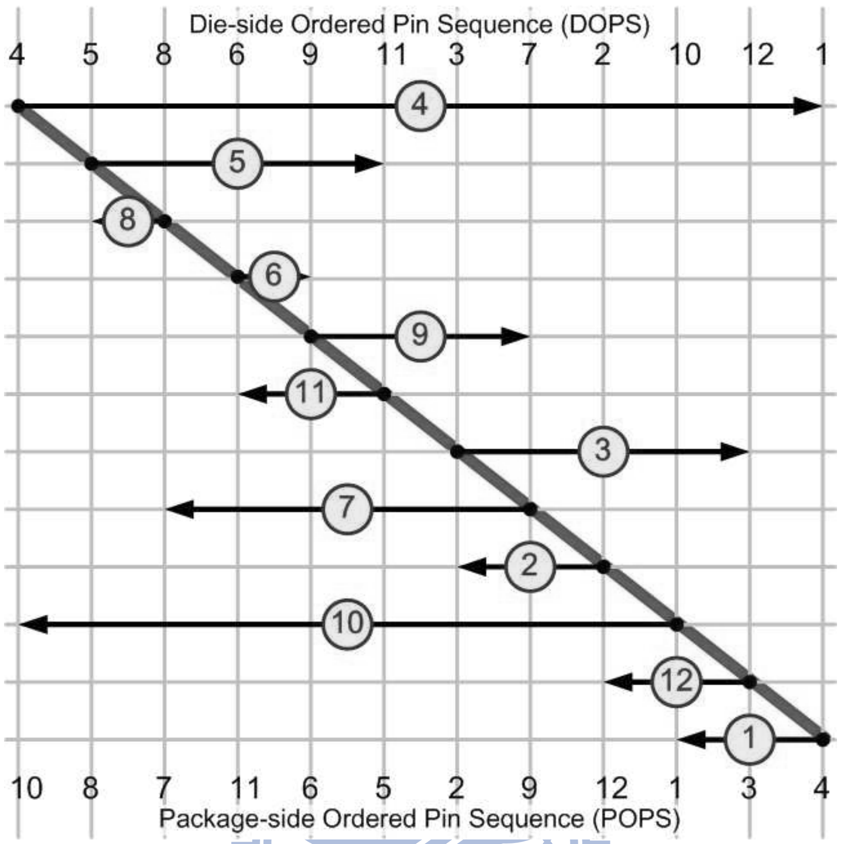

Die-side Ordered Pin Sequence (DOPS)
4 5 8 6 9 11 3 7 2 10 12 1
4
5
8
6
9
11
3
7
2
10
12
1
10 8 7 11 6 5 2 9 12 1 3 4
Package-side Ordered Pin Sequence (POPS)

Figure 3.7: Interval Diagram.

Fig. 3.5 shows an example of initial assignment. By using this designation algorithm, we can obtain the monotonic global routing for package design and PCB escape routing. In the next section, we propose the pin-out optimization methods considering the ways to minimize the package size, routing congestion and wirelength difference on each routing layer which are critical concerns in chip-package-board codesign.

3.3 Pin-Out Optimization

3.3.1 Optimization for Individual Objectives

Optimization scheme targeting at three individual objectives is discussed in this section: package size (PS), wire congestion ($Cong$), and sum of length difference on

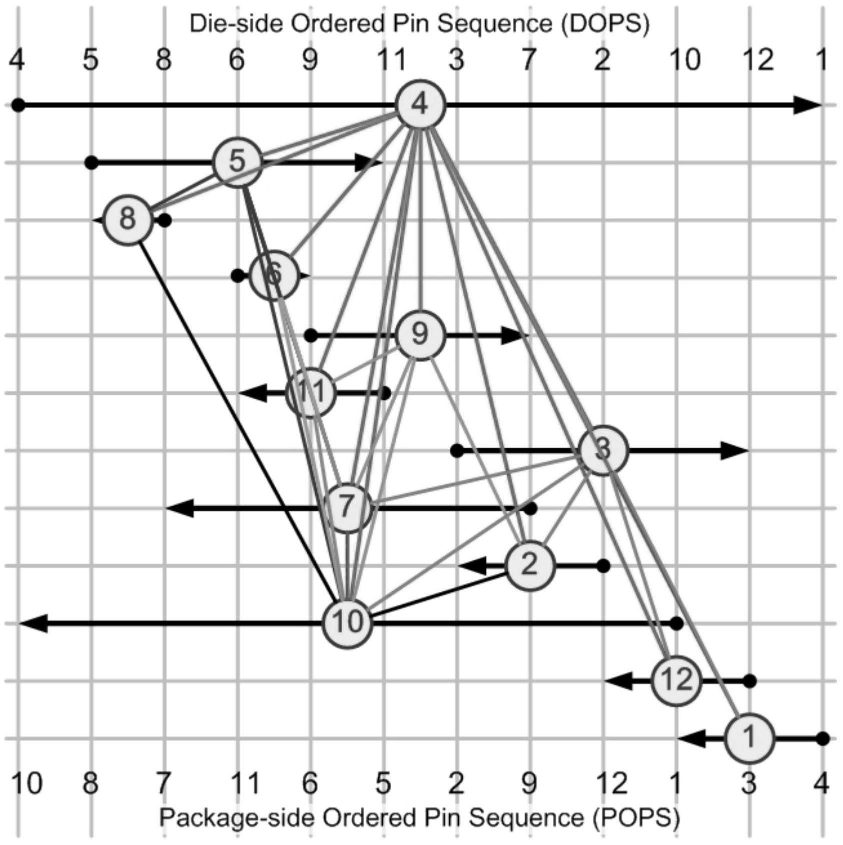


Figure 3.8: Intersection Graph.

each routing layer ($Diff$).

$Cong$ minimization can be achieved using two simple methods. One is to equally distribute routing channels: when pins are constrained to be at the same row, intuitively, averaged routing channel distribution minimizes wire congestion. For example, when there are 4 columns and 2 pins, it is best to assign p_1 and p_2 to $column_2$ and $column_4$ respectively. The other is to change the column number of pins. As shown in Fig. 3.9, moving p_3 out of the original row relieves $Cong$ a lot. It is trivial that focusing on $Cong$ minimization possibly enlarges pin-block size, however, the proposed general optimization scheme can still find the assignments which decrease $Cong$ while preserving PS.

$Diff$ minimization is to minimize the variation in wirelength for each layer. Length differences for each layer should be considered separately because the electrical characteristics on package are different from that on PCB. Note that the sum of routes are longer on the package layer-1 because plating leads are always required in packaging process. Rather than minimizing all wires altogether, to minimize the longest wires is sufficient because they often dominate $Diff$.

PS minimization can be obtained if the pins are moved in a certain order iteratively. The order is generated by post-order traversal of intersection graph. For instance, it is 8, 10, 7, 11, 6, 5, 2, 9, 12, 1, 3, 4, for the tree in Fig. 3.8 and Fig. 3.10(a). The pins move sequentially in this order and obey the direction priority (go left $>$ go bottom-left $>$ go down). Fig. 3.10 (b)-(f) illustrates each step of the procedure in minimizing PS. For each step, only one pin moves once at a time as long as there exists an empty via/ball slot, and all pins move in order. Each pin stops moving if it touches the boundary thus the number of rows cannot be increased. In the initial solution, the number of rows is minimum, which is determined by the depth of intersection tree shown in Fig. 3.10(a). In order to preserve monotonic assignment, pins which cross each other cannot be placed in the same row. Thus, to minimize

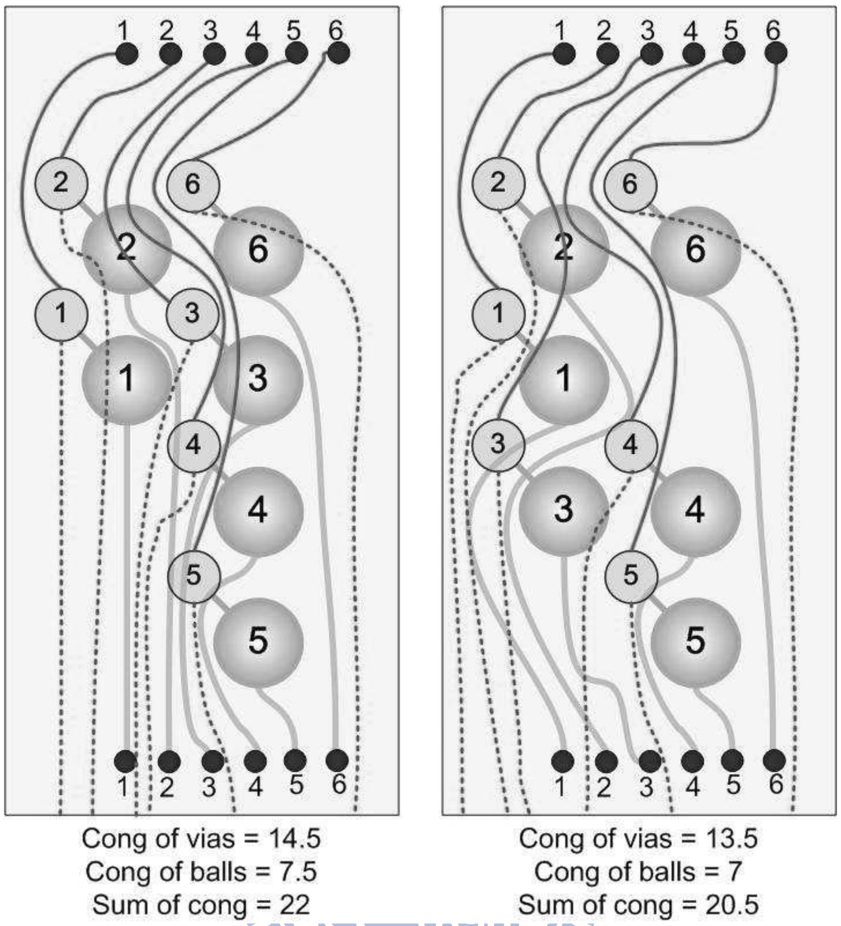


Figure 3.9: Optimization for wire congestion.

package size is actually to minimize the number of columns.

3.3.2 Unified Cost Optimization

The proposed optimization scheme is to: 1)select one pin/via/ball which costs most, 2)search its legal neighbors, 3)perform operations between the pin/via/ball and its legal neighbors. It is important to honor the legality in order to keep the assignment monotonic. The costs of via grid array (cVGA) and ball grid array (cBGA) are summed together. So an optimization step which merits cVGA may demerit cBGA.

We use the following heuristics to find better solutions in moving pin/via/ball.

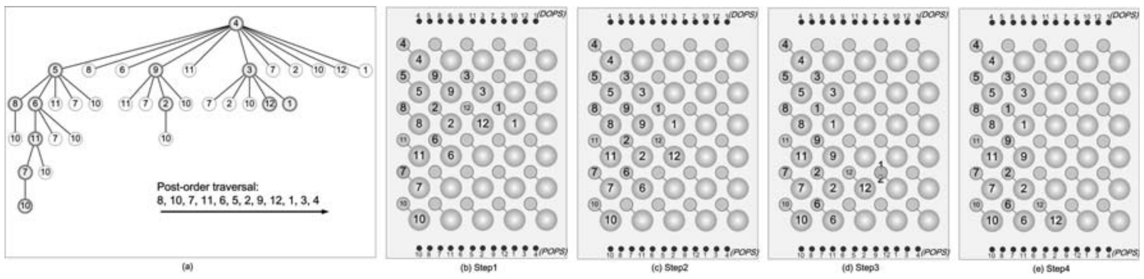


Figure 3.10: Optimization for package size. (b)-(f) show the status of movement. The number of columns used is decreased from 5 to 3.

Note that $Cong$, $Diff$, and PS are normalized to fairly compare with each other.

- Greedy Method: Starting from initial solution, only down-hill searches are accepted. The method keeps moving the most expensive pin/via/ball to its less expensive neighbors. It is useful when there is one pin/via/ball which contributes a lot in cost. However, it is not suitable when there is a group of pins/vias/balls which should be optimized simultaneourly.

- Lowest partial cost (LPC) Method: A serial of moves are firstly accepted to escape local optimal. Moves are performed whose accumulated sum of costs is minimum. The first move is relatively important because the quality of all following moves depends on it.

The optimization scheme is conducted in two stages: tie-up optimization followed by loose optimization. Each pair of v_i and b_i are tie-up as p_i to search for a global optima in the first stage, and they are loosened to search for local optima in the second stage. Fig. 3.11 shows an optimization step for n_3. p_3 attempts to decrease cost by exploring its neighbors in (a), while v_3 and b_3 search to decrease their own cost separately.

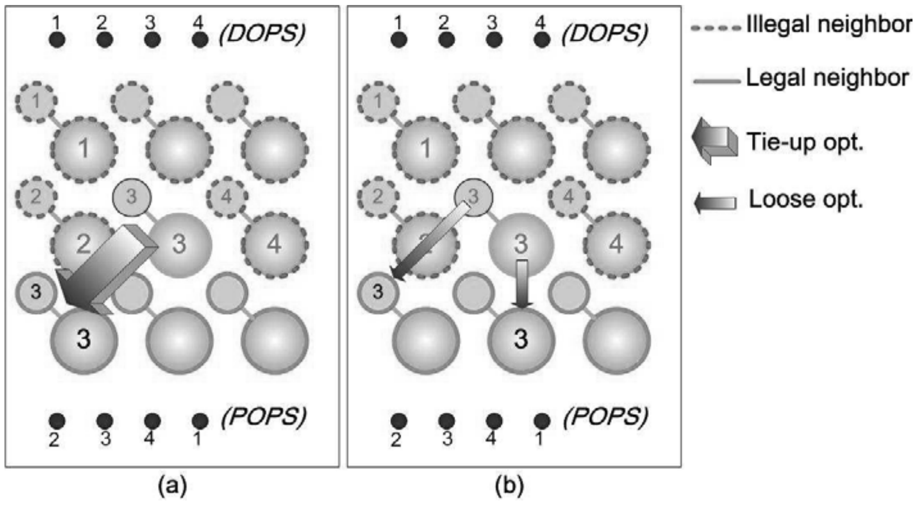


Figure 3.11: Post Optimization. We first tie up v_3 and b_3 for global search, then we loosen this constraint so that they can separately find other local solutions to reduce the cost.

3.3.3 Cost Evaluation

In the unified cost optimization, the cost function is defined as follows:

$$Cost_{vi/bi} = \alpha \times Cong_{vi/bi} + \beta \times Diff_{vi/bi} + \gamma \times PS_{vi/bi} \tag{3.1}$$

where $Cost_{vi/bi}$ indicates the cost of a via v_i or a ball b_i, and α, β, and γ are user-defined parameters. Each via/ball has a cost composed of $Cong$, $Diff$, and PS. And total cost is the sum of all vias/balls. Here we define the cost of three objectives separately.

The cost of wire congestion of two via/ball is defined as:

$$Cong_{vi/bi} = \frac{\#wire_l}{\#channel_l} + \frac{\#wire_r}{\#channel_r} \tag{3.2}$$

where $\#wire_l$ denotes the number of wires which lie on the left, $\#channel_l$ denotes the number of routing channels on the left, shown in Fig. 3.12.

The cost of length difference is defined as:

$$Diff_{vi/bi} = dist(v_i/b_i) - dist(avg) \tag{3.3}$$

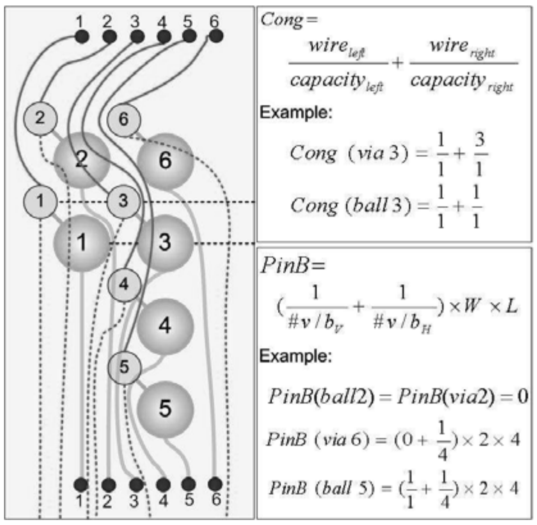

1 2 3 4 5 6
2 6 2 6 1 3 1 3 4 4 5 5
1 2 3 4 5 6
$Cong = \frac{wire_{left}}{capacity_{left}} + \frac{wire_{right}}{capacity_{right}}$
Example:
$Cong\ (via\ 3) = \frac{1}{1} + \frac{3}{1}$
$Cong\ (ball\ 3) = \frac{1}{1} + \frac{1}{1}$
$PinB = (\frac{1}{\#v/b_V} + \frac{1}{\#v/b_H}) \times W \times L$
Example:
$PinB(ball2) = PinB(via2) = 0$
$PinB\ (via\ 6) = (0 + \frac{1}{4}) \times 2 \times 4$
$PinB\ (ball\ 5) = (\frac{1}{1} + \frac{1}{4}) \times 2 \times 4$

Figure 3.12: Cost evaluation for via and ball.

where $dist(v_i/b_i)$ denotes the Manhattan distance of the via/ball and $dist(avg)$ denotes the averaged Manhattan distance of all vias/balls. Since the monotonic assignment can guarantee monotonic routing, using Manhattan distance to estimate wirelength is sufficient.

The cost of PS is defined as:

$$PS_{vi/bi} = \begin{cases} 0 & \text{if } \#v/b_V = 0 \\ 0 & \text{if } \#v/b_H = 0 \\ \frac{1}{\#v/b_V} + \frac{1}{\#v/b_H} \times W \times L & \text{otherwise} \end{cases} \tag{3.4}$$

where $\#v/b_V$ and $\#v/b_H$ denote the number of vias/balls which lie on the vertical and horizonal boundary respectively, W and L denote number of columns and rows of package size respectively, shown in Fig. 3.12.

3.4 Experimental Results

Our algorithm is implemented using C++ on a 3.0GHz Intel Xeon Quad Core Processor 5160 PC under the Linux operation system. In the experiments, α, β, and γ are all set to 1. And in the following tables, $cVGA$ denotes the total cost of Via Grid Array, $cBGA$ denotes the total cost of Ball Grid Array, and Sum is the sum of $cVGA$ and $cBGA$.

$$cVGA = \sum_{i=0}^{n} Cost_{vi} \tag{3.5}$$

$$cBGA = \sum_{i=0}^{n} Cost_{bi} \tag{3.6}$$

$$Sum = cVGA + cBGA \tag{3.7}$$

Two optimization schemes, *Greedy* and LPC, are implemented and tested in two modes: *tie-up* and $full$. *tie-up* indicates that, for n_i, v_i and b_i are tied-up as p_i to optimize simultaneously. $full$ indicates that, after having conducted *tie-up*, p_i is loosened to perform optimization for vias and balls separately.

As shown in Table 3.1, the costs of the proposed initial solution are all 3.00 because the costs of three objectives ($Cong$, $Diff$, and PS) are normalized to their initial solution. The proposed initial solution is generated by Initial Pin-Out Designation Algorithm proposed in Section III. Both [19] and the proposed methodology adopt the same 2-layer package model. However, [19] assigns its initial solution randomly and performs optimization for cVGA only. We test the proposed methodology in 5 industrial cases in which the largest pin count is 40 (for bench2-5, on one side). The proposed initial solution improves 54% in average, compared to [19]. The initial solutions can be futher improved by 22% in average after applying *Greedy* method in $full$ mode.

Table 3.2 shows the results of two optimization schemes *Greedy* and LPC. Similiar behaviors are observed for most of the results. They improve the initial solutions by 16% in tie-up mode. This can be further optimized in full mode to give a final improvement of 20-22% in average. Note that the initial assignment of bench1 is shown previously in Fig. 3.5. The execution time for all experiments is less than 1 second.

Table 3.1: Compare [19] and the proposed post-optimization scheme with the proposed initial solution.

	The proposed initial solution			[19] (opt. for cVGA only)				Greedy Method full mode			
	cVGA	cBGA	Sum	cVGA	cBGA	Sum	Imp.%	cVGA	cBGA	Sum	Imp.%
bench2	3.00	3.00	6.00	4.43	3.42	7.85	-31%	1.94	1.95	3.90	35%
bench3	3.00	3.00	6.00	7.20	3.91	11.12	-85%	2.11	2.11	4.23	30%
bench4	3.00	3.00	6.00	6.09	3.28	9.37	-56%	2.84	2.60	5.45	9%
bench5	3.00	3.00	6.00	5.80	2.73	8.53	-42%	2.71	2.51	5.23	13%
avg.	3.00	3.00	6.00	5.88	3.34	9.22	-54%	2.40	2.30	4.70	22%

The results of bench3 are plotted as shown in Fig. 3.13. (a) shows the proposed initial assignment in which all wires are planned monotonically. (b) and (c) shows the results of post-optimization using *Greedy* and *LPC* methods respectively. They have similar patterns with slightly different assignment. The cost of (b) is lower than (c) by 6% because greedy method can find better solution in loose mode. This shows that the cost of BGA benefits more than that of VGA from loose optimization. (d) is one of the experimental results of [19], which shows smaller in package size, however more congested and has larger variation in wirelength.

3.5 Summary

In order to address the long-existing problem in slow turn-around between design, package and system houses, we define a new subproblem in helping the fast estimation of wire planning in chip-package-board codesign. Core designers can specify the preferred I/O pad ordering; system designers can specify the preferred bump pin-out designation. We can efficiently analyze if the preferences from both sides accommodate each other, before performing RDL routing and substrate routing. We also consider the essential concerns in package design, such as routing congestion, package size and length deviation among all nets. The results show the effectiveness and efficiency.

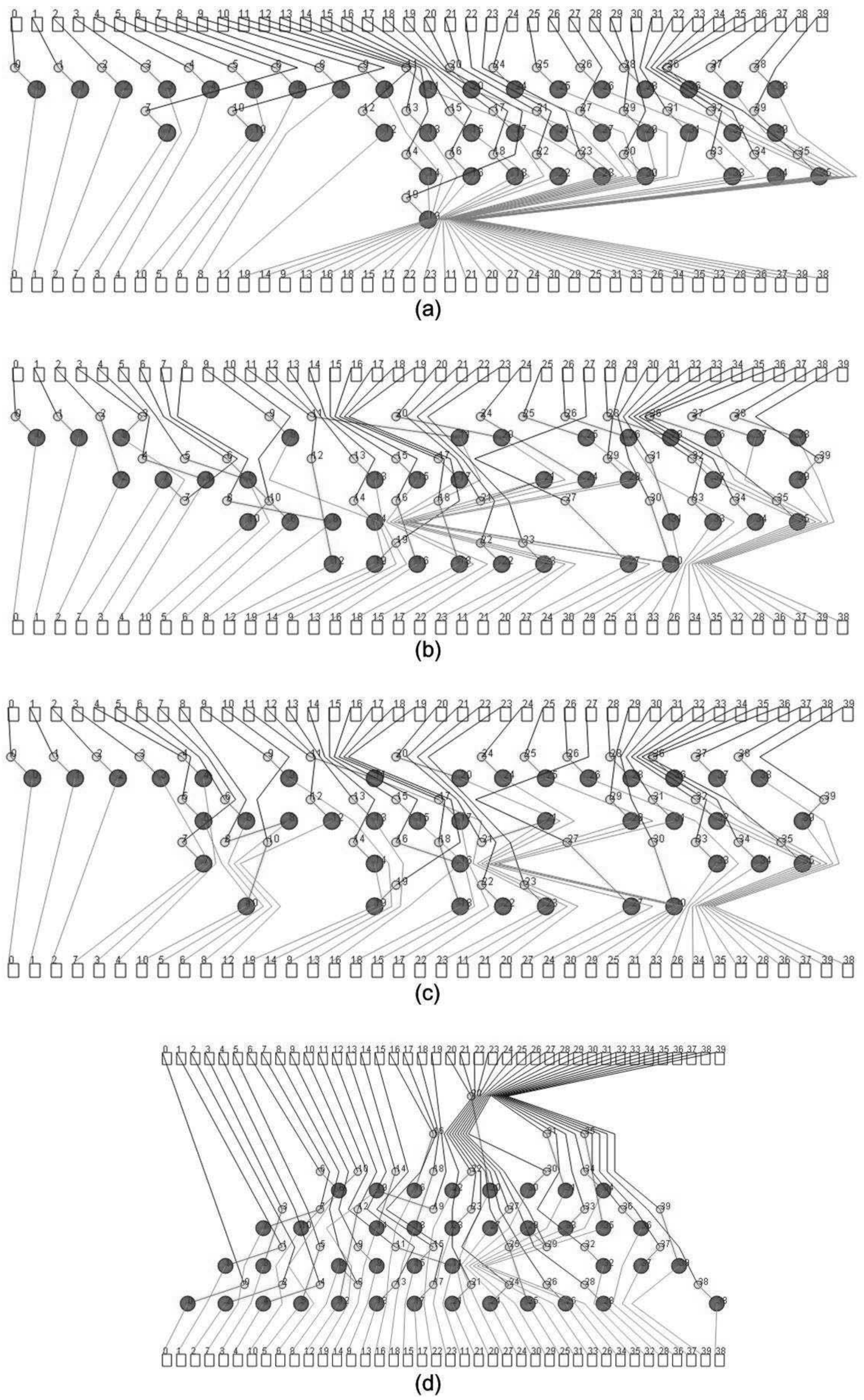


Figure 3.13: Experimental results for *bench*3: (a)the proposed initial solution (b)the result of greedy method (c)the result of LPC method (d)the result of [19].

Table 3.2: Post optimization improves about 20%, compared with the proposed initial solution.

	Greedy Method							
	tie-up mode				full mode			
	cVGA	cBGA	Sum	Imp.%	cVGA	cBGA	Sum	Imp.%
bench1	2.36	2.62	4.99	17%	2.32	2.42	4.74	21%
bench2	2.03	2.15	4.18	30%	1.94	1.95	3.90	35%
bench3	2.38	2.50	4.87	19%	2.11	2.11	4.23	30%
bench4	2.89	2.92	5.81	3%	2.84	2.60	5.45	9%
bench5	2.78	2.68	5.46	9%	2.71	2.51	5.23	13%
avg.	2.49	2.57	5.06	16%	2.39	2.32	4.71	22%
	LPC Method							
	tie-up mode				full mode			
	cVGA	cBGA	Sum	Imp.%	cVGA	cBGA	Sum	Imp.%
bench1	2.36	2.62	4.99	17%	2.32	2.42	4.74	21%
bench2	2.08	2.12	4.20	30%	1.94	2.05	3.99	33%
bench3	2.38	2.50	4.87	19%	2.12	2.42	4.53	24%
bench4	2.89	2.92	5.81	3%	2.85	2.65	5.51	8%
bench5	2.79	2.67	5.46	9%	2.75	2.51	5.27	12%
avg.	2.50	2.56	5.06	16%	2.40	2.41	4.81	20%

Chapter 4

Concluding Remarks and Future Work

This thesis proposes an automated solution for chip design houses to facilitate chip-package-board interfacing. Our methodologies are optimized for planning the critical interfaces between chip, package and board in high-end IC designs. The contributions of this thesis and possible future works are summarized as follows.

4.1 Concluding Remarks

- **RDL Routing on Pseudo Single-Layer**: Algorithms for RDL routing on pseudo single-layer is introduced in Chapter 2. To deal with congested RDL, we propose the concept of pseudo single-layer. It's a compromise solution other than adding an extra metal layer or increasing the die-size. By applying left-edge algorithm, 100% wires are routed and the area of 2-layer routing is minimized. For the real industrial case, commercial tools cannot even initiate routing process. Manual routing can reach 100% routability but the designers in collaboration spent about one month to achieve that. However, our proposed method can finish RDL routing automatically very effectively.

- **Fast Package Pin-out and Wire Planning in Chip-Package-Board**

Codesign: Methodologies for chip-package-board co-design is introduced in Chapter 3. A sub-problem considering the optimization of wire planning, package size, routability, and signal integrity, is proposed. It takes I/O pad ordering and ball pin-out designation from core designers and system designers respectively. Our methodologies greatly fastens turn-around between design, package and system houses. The results show the effectiveness and efficiency.

4.2 Future Work

For RDL routing, we should deal with some additional situations. For example, the bump grid arrays can be staggered for more complex designs. The RDL router has to be more applicable to different design layout pattern, such as multiple rings of IO pads. Also, doglegs are required to support multi-terminal nets to enable PG routing together with signal routing. To make chip-package-board co-design more realizable, some experiments must be conducted from the industry, such as RC extraction and chip-package-board co-simulation. At the same time, we can discover more predicaments underneath today's conventional design flow.

Bibliography

[1] P. Dehkordi and D. Bouldin, "Design for packageability - the impact of bonding technology on the size and layout of VLSI dies," in *Proc. Multi-Chip Module Conference*, pp. 153-159 , 1993.

[2] J.-W. Fang and Y.-W. Chang, "Area-I/O flip-chip routing for chip-package co-design, in *Proc. IEEE/ACM international Conference on Computer-Aided Design*, pp. 518-521, 2008.

[3] H.-C. Lee, Y.-W. Chang, and P.-W. Lee, "Recent research development in flip-chip routing," in *Proc. IEEE/ACM international Conference on Computer-Aided Design*, pp. 404-410, 2010.

[4] J.-W. Fang, I-Jye Lin, Y.-W. Chang, and J.-H. Wang, "A Network-Flow-Based RDL Routing Algorithmz for Flip-Chip Design," in *IEEE Trans. on Computer-Aided Design of integrated Circuits and Systems*, vol. 26, no. 8, pp. 1417-1429, Aug. 2007.

[5] J.-W. Fang, I-Jye Lin, P.-H. Yuh, Y.-W. Chang, and J.-H. Wang, "A routing algorithm for flip-chip design," in *Proc. IEEE/ACM international Conference on Computer-Aided Design*, pp. 753- 758, Nov. 2005.

[6] J.-W. Fang, C.-H. Hsu, and Y.-W. Chang, "An integer Linear Programming Based Routing Algorithm for Flip-Chip Design," in *Proc. ACM/IEEE Design Automation Conference*, pp. 606-611, June 2007.

[7] P.-W. Lee, C.-W. Lin, Y.-W. Chang, C.-F. Shen, and W.-C. Tseng. "An efficient pre-assignment routing algorithm for flip-chip designs," in *IEEE/ACM international Conference on Computer-Aided Design*, pp. 239-244, Nov. 2009.

[8] J.-W. Fang, C.-H. Hsu, and Y.-W. Chang, "An integer-Linear-Programming-Based Routing Algorithm for Flip-Chip Designs," in *IEEE Trans. on Computer-Aided Design of integrated Circuits and Systems*, vol. 28, no. 1, pp. 98-110, 2009.

[9] J.-W. Fang and Y.-W. Chang, "Area-I/O Flip-Chip Routing for Chip-Package Co-Design Considering Signal Skews," in *IEEE Trans. on Computer-Aided Design of integrated Circuits and Systems*, vol. 29, no. 5, pp. 711-721, 2010.

[10] K.-S. Lin, H.-W. Hsu, R.-J. Lee, and H.-M. Chen, "Area-I/O RDL routing for chip-package codesign considering regional assignment," in *Proc. IEEE Electrical Design of Advanced Packaging & Systems Symposium*, pp. 1-4, 2010.

[11] J.-W. Fang, M.D.F. Wong, and Y.-W. Chang, "Flip-chip routing with unified area-I/O pad assignments for package-board co-design," in *Proc. ACM/IEEE Design Automation Conference*, pp. 336339, 2009.

[12] R.-J. Lee and H.-M. Chen, "Fast Flip-Chip Pin-Out Designation Respin for Package-Board Codesign," in *IEEE Trans. on Very Large Scale integration Systems*, vol. 17, no. 8, pp. 1087-1098, Aug. 2009.

[13] J.-W. Fang, I.-J. Lin, Y.-W. Chang ,and J.-H. Wang, "A Network-Flow Based RDL Routing Algorithms for Flip-Chip Design," in *IEEE Trans. on Computer-Aided Design of integrated Circuits and Systems*, vol. 26, no. 8, pp. 1417-1429, Aug. 2007.

[14] J.-W. Fang, C.-H. Hsu ,and Y.-W. Chang, "An integer-Linear-Programming-Based Routing Algorithm for Flip-Chip Designs," in *IEEE Trans. on Computer-*

Aided Design of integrated Circuits and Systems, vol. 28, no. 1, pp. 98-110, Jan. 2009.

[15] M. M. Ozdal and D.-F. Wong, "Algorithms for Simultaneous Escape Routing and Layer Assignment of Dense PCBs," in *IEEE Trans. on Computer-Aided Design of integrated Circuits and Systems*, vol. 25, no. 8, pp. 1510-1522, Aug. 2006.

[16] H. Kong, T. Yan ,and D.-F. Wong, "Automatic Bus Planner for Dense PCBs," in *Proc. ACM/IEEE Design Automation Conference*, pp. 326-331, 2009.

[17] L. Luo and D.-F. Wong, "Ordered Escape Routing Based on Boolean Satisfiability," in *Proc. Asia and South Pacific Design Automation Conference*, pp. 244-249, 2008.

[18] Y. Kubo and A. Takahashi, "A Global Routing Method for 2-Layer Ball Grid Array Packages," in *Proc. international Symposium on Physical Design*, pp. 36-43, 2005.

[19] Y. Tomioka and A. Takahashi, "Routability Driven Modification Method of Monotonic Via Assignment for 2-Layer Ball Grid Array Packages," in *Proc. Asia and South Pacific Design Automation Conference*, pp. 238-243, 2008.

[20] S.-S. Chen, J.-J. Chen, T.-Y. Lee, C.-C. Tsai, and S.-J. Chen, "A New Approach to the Ball Grid Array Package Routing," in *Trans. on IEICE*, vol.E82-A, no.11, pp.2599-2608, 1999.

[21] L.-T. Wang, Y.-W. Chang, and K.-T. Cheng "Electronic Design Automation: Synthesis, Verification, and Test (Systems on Silicon)," *Morgan Kaufmann, 2009, ISBN: 978-0-1237-4364-0.*