國立交通大學

電子工程學系 電子研究所

博 士 論 文

晶片—封裝—印刷電路板共同設計之演算法

Algorithms for Chip-Package-Board Codesign

研 究 生：李仁傑

指導教授：陳宏明 教授

中 華 民 國 九 十 九 年 二 月

晶片—封裝—印刷電路板共同設計之演算法

Algorithms for Chip-Package-Board Codesign

研 究 生：李仁傑　　Student：Ren-Jie Lee

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

博 士 論 文

A Dissertation

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

Doctor of Philosophy

in

Electronics Engineering

February 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年二月

晶片—封裝—印刷電路板共同設計之演算法

學生: 李仁傑　　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　博士班

摘　　要

因應系統晶片(SoC)與系統封裝(SiP)等產品急遽增加的趨勢，晶片、封裝和印刷電路板的設計以及其間的訊號互動，兩者之複雜度均快速成長。典型的周邊打線(wire-bond)封裝方式將不復適用於現今大部份的設計；因此覆晶(flip-chip)封裝成為一個必然的選擇。然而，在傳統的覆晶封裝設計中，工程師通常利用手動的方式來安排關鍵的介面，包括：輸入/輸出(I/O)、連接凸塊(bump)及封裝接腳(pin-out)，這種方式在晶片—封裝—印刷電路板(chip-package-board)的共同設計過程中相當耗時且費工，所以總是造成產品上市時程(time-to-market)的延宕。針對上述的問題，本篇論文提出一系列自動化安排、規劃這些重要介面的方法，配合論文中所發展之同時且共同設計流程(concurrent codesign flow)，可大幅加速產品設計的時程。

本篇論文主要包含三個部份。首先，針對封裝—印刷電路板 (package-board)共同設計的工作，論文中提出了一個全新且非常有效率的方法，以自動安排覆晶封裝的接腳位置，來取代以往耗時、費工的手動設計方式。在此方法中，由於建構與擺置封裝接腳方塊(pin-block)的過程，可以同時考慮訊號完整性(signal integrity)、電源供應(power delivery)和可繞線度(routability)，因此我們的方法不但讓工程師在產品效能及成本之間做彈性的選擇與取捨，並且同時實現封裝尺寸的最小化。其次，為了最佳化接腳方塊的位置，本論文提出另一個規劃封裝接腳的演算法。此方法應用新的接腳方塊擺置表示法(placement representation)，在有

區域限制式(range constraint)的情況下，利用隨機的方法(stochastic framework)達到最佳化的目的。實驗結果顯示，在安排封裝接腳時改善接腳方塊的位置，確實可使系統連線(system interconnect)得以最佳化。除了處理封裝—印刷電路板共同設計的問題外，論文在第三部份亦發展了晶片—封裝(chip-package)同時且共同設計流程。相較於其他文獻中的演算法，我們的晶片—封裝共同設計方法針對繞線交錯(net crossing)與線長差異(length deviation)等重要設計考量，執行初步的探索及研究，試圖使兩者在爾後的設計過程中均可達成最佳化。藉由設計特定的輸出入連接凸塊板(I/O-bump tile)，以及提出新穎的輸出入列(I/O-row)結構，論文中呈現了兩個啟發式方法(heuristic method)和一個指派演算法(assignment algorithm)，根據已知的封裝接腳位置安排輸入/輸出和連接凸塊。綜合上述的研究成果，我們完成了同時兼顧產品效能與成本的自動化晶片—封裝—印刷電路板共同規劃任務。

Algorithms for Chip-Package-Board Codesign

Student: Ren-Jie Lee Advisor: Prof. Hung-Ming Chen

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

ABSTRACT

Due to the trend of more and more SoC and SiP projects, the complication in chip, package and board designs, and signal interactions thereof is increasing very rapidly. Typical peripheral wire-bond design will be inappropriate for most modern designs; therefore flip-chip package becomes an inevitable choice. However, engineers usually designate the key interfaces including I/Os, bumps and package pin-out (ballplan) by hands in conventional flip-chip designs. The chip-package-board co-planning process is indeed time-consuming and always postpones the time-to-market (TTM) of products. In response to the aforementioned issues, this dissertation proposes methodologies in planning those interfaces with concurrent codesign paradigm, thus speeding up the developing time dramatically.

The dissertation contains three parts. First, we propose a novel and very efficient approach to automating pin-out designation in flip-chip BGA packaging for package-board codesign. The manual time-consuming codesign works can be replaced by proposed methodologies. Through considering signal integrity, power delivery, and routability in pin-block design, our frameworks provide trade-offs in signal performance and package cost while achieving the minimum package size. Second, we present a planning algorithm to optimize pin-block locations by using a new representation for pin-block placement, and defining range constraints in stochastic

framework. The experimental results show that our algorithm optimizes the system interconnects during package pin-out planning. In addition to the package-board codesign, we develop a concurrent design flow for chip-package codesign in the third part. Comparing with the previous works, the methods in this part preliminarily provide the optimization study of net crossing and length deviation which are very critical requirements in chip-package codesign. By designing specific I/O-bump tiles and proposing an innovative I/O-row based scheme, two heuristic methods and one assignment algorithm are provided for package-aware I/O-bump planning. As a result, a chip-package-board co-planning automation attempt is accomplished for optimizing performance and design cost simultaneously.

誌　　謝

本論文的完成，首先我要特別感謝指導教授陳宏明教授。由於當年教授的鼓勵與推薦，使我如願進入交大電子所博士班就讀，一圓我在國內一流學府受教的夢想。教授，您豐富的專業涵養、開放的指導方式以及亦師亦友的教學風格，不僅幫助我在攻讀博士學位期間順利度過由於轉換研究領域所遭逢的困境，更在我撰寫博士論文時產生極大的助益。您的指引與建議，使我的研究方向能夠契合目前工業界的迫切需求，並大幅的提升了研究成果的實用性，因而讓我對於電子設計自動化(EDA)產業，有機會貢獻棉薄之力。

其次，畢業論文得以順利定稿，必須要感謝所有的口試委員: 吳誠文教授、張耀文教授、王廷基教授、麥偉基教授、李毅郎教授、李育民教授與陳墅矜博士。您們的不吝指教，讓我深刻體認何謂紮實的研究；您們的寶貴建議，讓我的畢業論文更趨完備與充實。

再者，我最想要感謝的是我的太太沛霖。當我在準備博士資格考驚慌失措時、在經歷研究論文不被接受的挫折時以及在遭遇研究過程中的低潮時，沛霖，妳的陪伴與安慰是驅使我繼續向前邁進的主要動力，妳是我的畢業論文幕後最大的功臣。謝謝妳，沛霖！若沒有妳不斷的支持與鼓勵，我將無法實現多年來的志願。

另外，還要感謝我的父親、母親、岳父和岳母，以及所有的親朋好友們。你們的關心與問候是我最大的依靠，支撐我持續前進而不被現實所擊倒。

最後，謹以此文獻給摯愛沛霖以及所有支持我的家人與好友們。

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Contributions

As silicon technology scales, more and more circuits could be integrated into a single chip. The amounts of input/output (I/O) signals have been dramatically increased per unit area. This trend elevates the degree of the complication in chip, package and board designs [1, 2]. Figure 1.1 shows the relationship between chip, package and board. Signal interactions among them now heavily influence the design performance due to the largely increased number of I/O signals. The impact of bad package and/or board design cannot be ignored in high-end designs any longer [3]. Accordingly, the modern chip design problem must be treated as a system-level problem. Under this circumstance, the chip-package-board codesign is therefore becoming increasingly critical in designing very large scale integration (VLSI) designs. Specifically, those designs are I/O-pad limited chips possessing more than one thousand I/Os.

Regarding the typical flip-chip design flow, an IC-driven flow [4], it has some inevitable shortcomings. As shown in Figure 1.2, after completing the chip physical design tasks including the core-I/O co-placement and detailed routing for core cells, this bottom-up design flow assigns the bump locations then connects bumps and corresponding I/Os with redistribution layer (RDL) routing. Next, beyond the die,

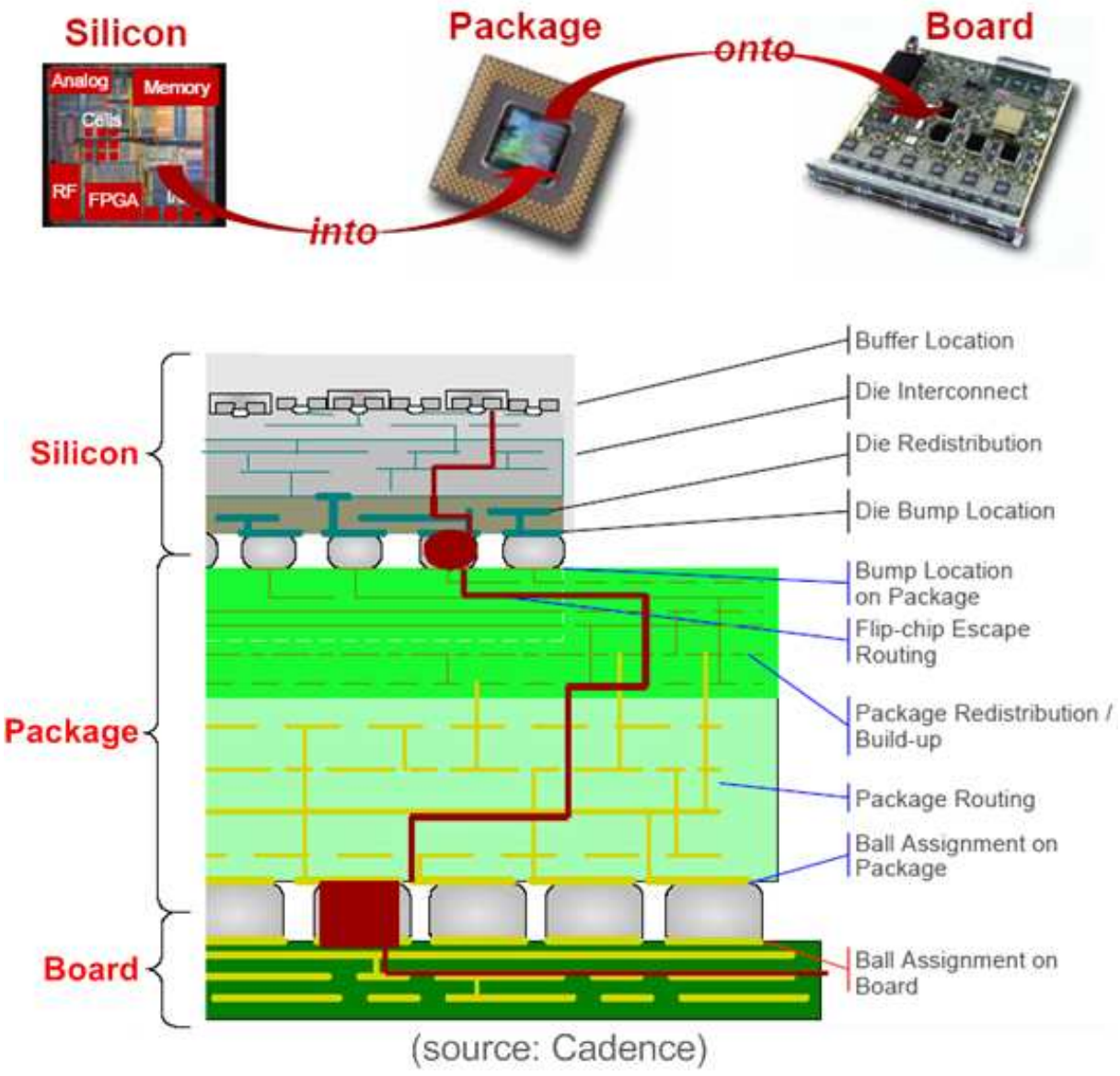


Figure 1.1: The relationship between chip, package and board. Die bumps and package balls are crucial interfaces influencing whole system's performance.

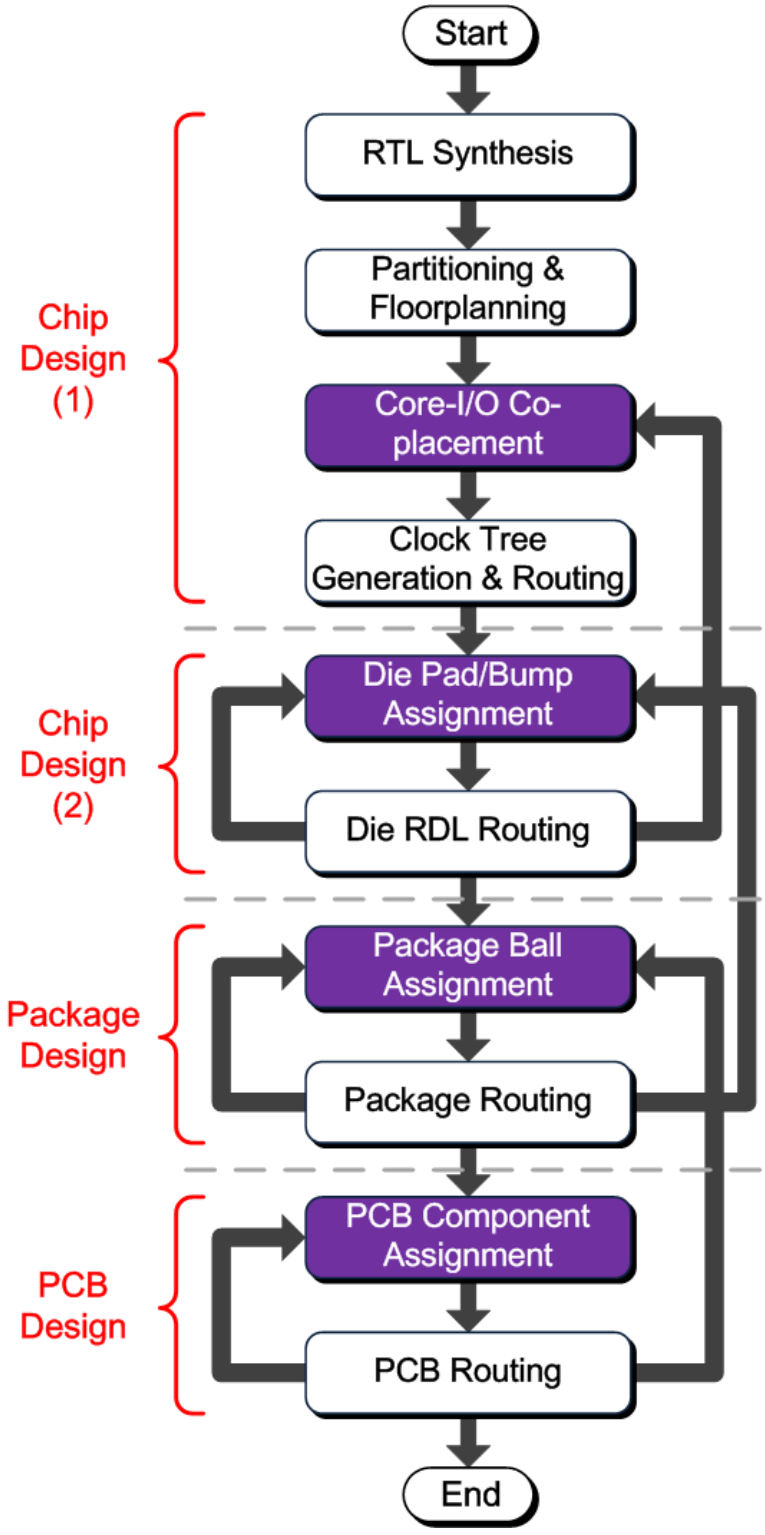


Figure 1.2: The typical sequential chip-package-board design flow. This flow usually takes long and costly turnaround times to meet entire system's design requirements.

the bump placement and ball locations assigned by package designer will be used for package routing. The last step is printed circuit board (PCB) routing in which PCB designer connects package balls to the components on PCB. Since each design stage has its own considerations and requirements, the major disadvantages of this sequential design flow are evident. This flow not only results in long and costly re-spin cycles on satisfying entire system's design constraints, but also inevitably causes the failed package and PCB design.

In order to overcome the drawbacks mentioned above, in this dissertation we have proposed a novel concurrent codesign flow as shown in Figure 1.3. Firstly, according to the given PCB component placement, our flow automatically assigns the package ball locations by planning dedicated pin-blocks. After that, a novel planner will fine-tune those pin-blocks to optimize the system interconnects. At last, through designing the specific I/O-bump tiles it produces the package-aware I/O-bump assignment for chip-level core cell placement. Comparing with the sequential design flow (see Figure 1.4), the concurrent one simultaneously completes the within-die and beyond-die design tasks composed of the chip physical design and routing works on die, package and PCB.

In addition to proposing the concurrent design flow for chip-package-board codesign (Figure 1.3), this dissertation also develops some automated methodologies to plan or to assign the interfaces among chip, package and board. Such interfaces include I/Os, bumps and balls which are very critical and affect whole system's performance. Table 1.1 shows the average runtime of a SoC system design[1] which approximately has $5.0M$ gate count, 500 I/Os and six major components on PCB. While implementing this system with typical design flow, the total runtime obtained

[1]This SoC system provided as an example design is a chipset IC packaged with a flip-chip ball grid array (BGA) and mounted on a motherboard. The related design information about the average runtime and design solutions in typical design flow is derived from an experienced engineer who is working for a design service compony in Hsinchu Science Park, Taiwan.

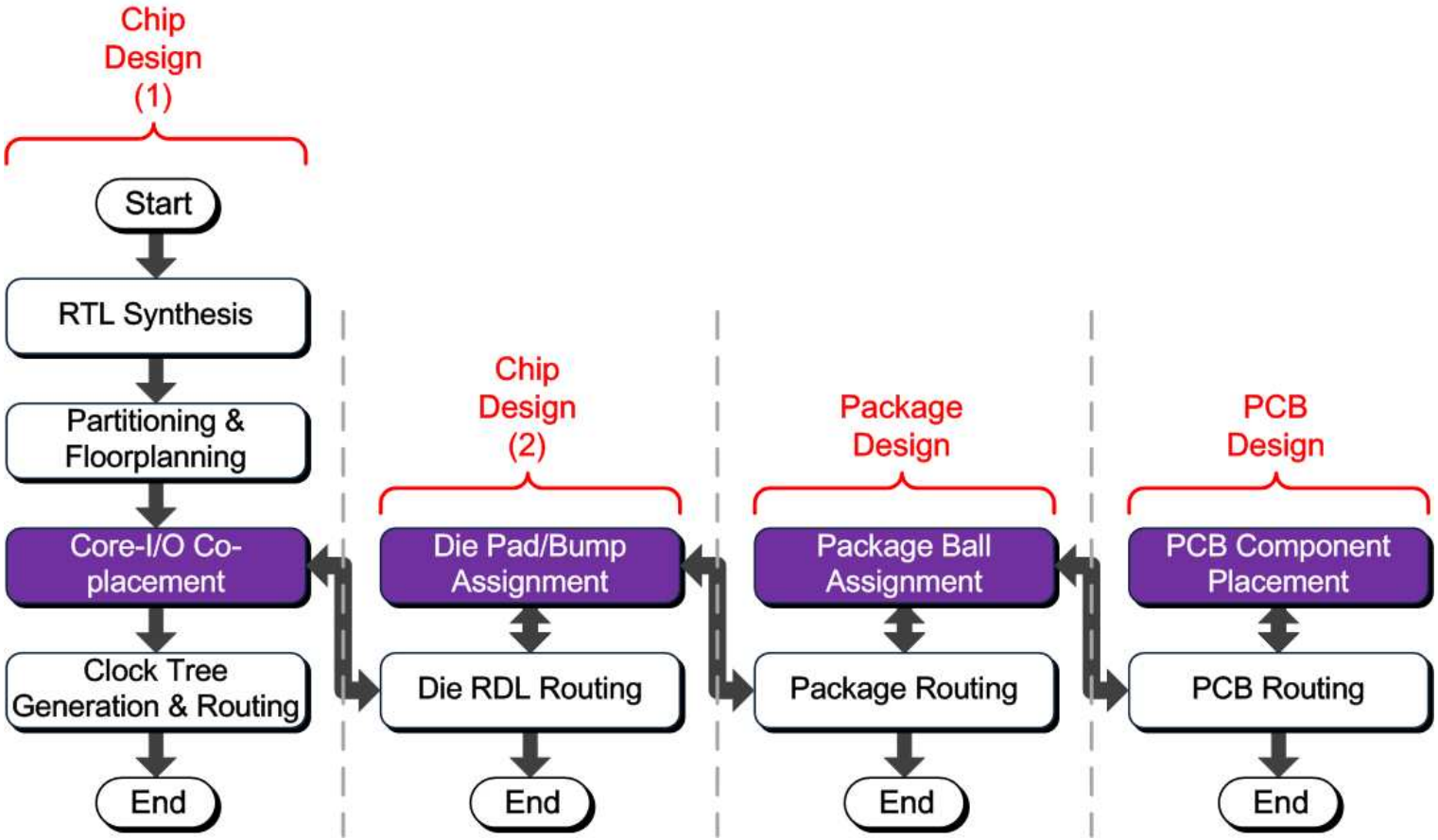


Figure 1.3: The proposed concurrent chip-package-board codesign flow. It converges design solutions within fewer iterations, thus reducing the time-to-market for designs.

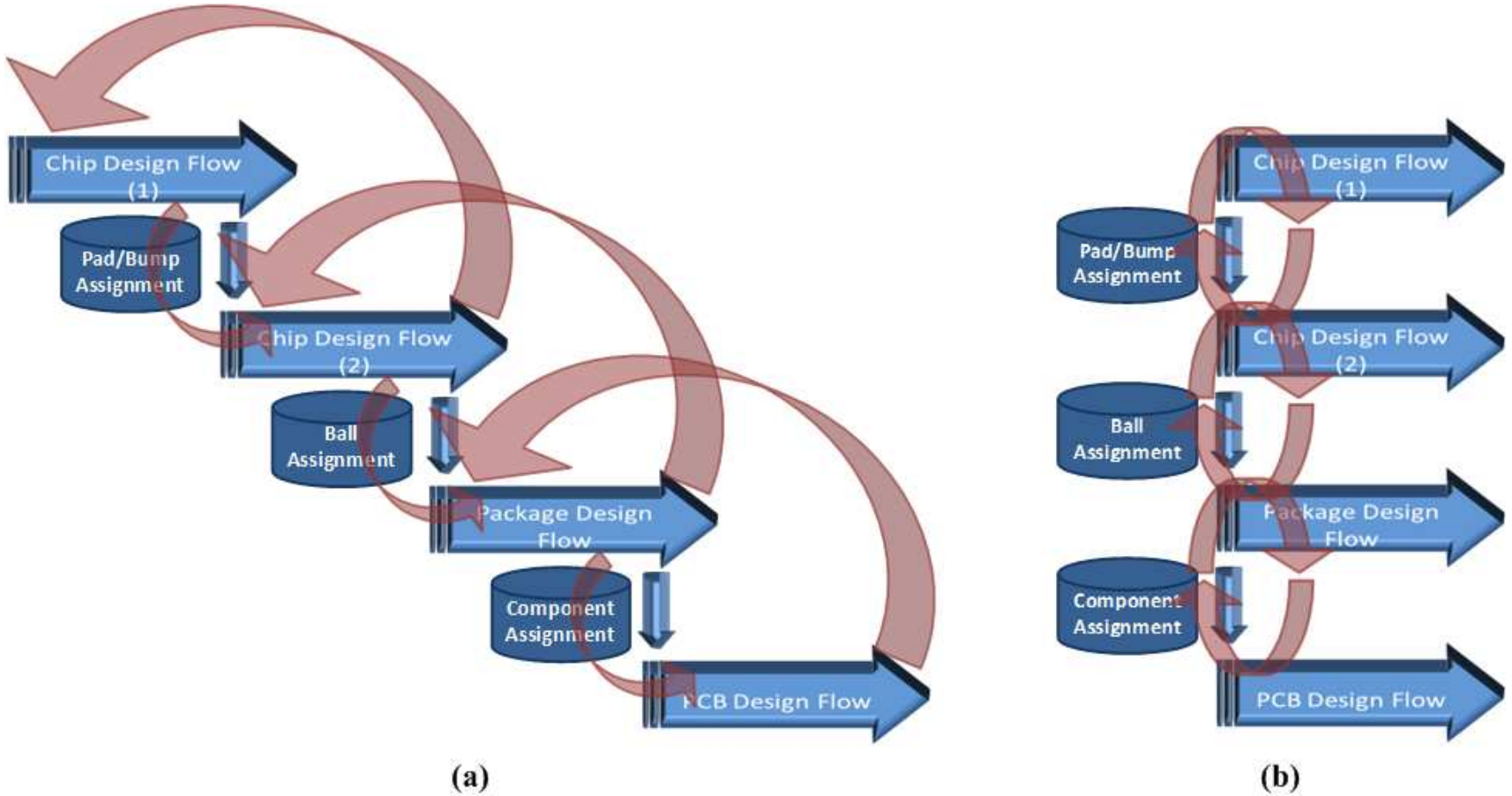


Figure 1.4: The comparisons of chip-package-board design flow. (a) A sequential design flow, and (b) a concurrent design flow which simultaneously completes the within-die and beyond-die design tasks.

from industry is around 6.0 (*week*). However, through using automated methodologies and going with the concurrent design flow, the total runtime can be significantly reduced to 1.5 (*week*). As a result, our approaches converge the solutions of chip, package and PCB design within fewer iterations and tremendously reduce the time-to-market (TTM) for designs.

1.2 Organization of This Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 proposes six signal-pin patterns for pin-block construction and floorplanning in package ball assignment. Signal integrity (SI), power delivery integrity (PI), and routability (RA) have been accounted for in those patterns, which helps to speed up the process of pin-out designation. Furthermore, we have proposed a near optimal approach to minimizing package size by mathematical (linear) programming formulation. In Chapter 3, we present the ideas to optimize the system interconnects during package pin-out design. Those ideas keep the same minimized package size as aforementioned pin-out assignment work and ensure that SI, PI, and RA can still be accounted for significant reduction in design cost. With considering the package ball location, our proposed methodologies provide a package-aware I/O-bump planning for chip-level core cell placement and package-level routing task. Next, in Chapter 4 we skip the redistribution layer (RDL) routing and design the specific I/O-bump tiles based on an innovative I/O-row scheme. Finally, we draw the conclusions and list some future works in Chapter 5.

Table 1.1: The runtime comparison of a SoC system design implemented with typical design flow and our automated methodologies. Through using automated methodologies and going with the concurrent design flow, the total runtime of this system can be significantly improved.

		t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
	Design task	Core cell placement	I/O placement	Core cell routing	Bump placement	RDL routing	Ball assignment	Package routing	PCB component assignment	PCB routing
Typical design flow (sequential)	Solution	Cadence SoC Encounter	Manual design	Cadence SoC Encounter	Manual design	Cadence Allegro PKG Designer	Manual design	Cadence Allegro PKG Designer	Manual design	Cadence Allegro PCB Design
	Average runtime	≈ 2.0 $(week)$			≈ 1.0 $(week)$		≈ 1.5 $(week)$		≈ 1.5 $(week)$	
	$Total\ runtime = \sum_{i=1}^{9} t_i = 6.0$ $(week)$									
Our design flow (concurrent)	Solution	Cadence SoC Encounter	Automated methodology	Cadence SoC Encounter	Automated methodology	Cadence Allegro PKG Designer	Automated methodology	Cadence Allegro PKG Designer	Manual design	Cadence Allegro PCB Design
	Average runtime	< 1.5 $(week)$			< 0.5 $(week)$		< 1.0 $(week)$		≈ 1.5 $(week)$	
	$Total\ runtime = \max(t_i) \approx 1.5$ $(week)$									

Chapter 2

Automated Package Pin-Out Designation for Package-Board Codesign

Deep submicron (DSM) effects drive the complication in designing chips, as well as in package designs and communications between package and board. As a result, the iterative interface design has been a time-consuming process. This chapter proposes a novel and efficient approach to designating pin-out, which is a package ball chart describing pin locations for flip-chip ball grid array (BGA) package when designing chipsets. The proposed approach can not only automate the assignment of I/O pins on package, but also precisely evaluate package size which accommodates all pins with almost no void pin positions, as good as the one from manual design. Furthermore, the practical experience and techniques in designing such interface has been accounted for, including signal integrity, power delivery and routability. This efficient pin-out designation and package size estimation by pin-block design and floorplanning provides much shorter turn around time, thus enormous improvement in meeting design schedule. The results on two real cases show that our methodology is effective in achieving almost the same dimensions in package size, compared with manual design in weeks, while simultaneously considering critical issues and package size migration in package-board codesign.

2.1 Overview

Because of DSM technology, chips now contain more functionality and are being driven to higher performance levels than ever before. Consequently, with more functionality on the chip, designers have to deal with higher I/O densities, more signals coming out of a chip and tighter geometries [5]. This leads to the complication in designing package which accommodates chips, as well as the board which accommodates the packages. As a result, the ability to design the chip, the package and surrounding system concurrently becomes a primary advantage, but also a challenge.

Recently chip-package codesign has drawn attention under the circumstances mentioned above [6, 7]. However package-board codesign, which is definitely not a trivial work, still needs more works. Several researches were related to package and printed circuit board (PCB) physical designs [8, 9, 10, 11]. [8] presented a style for BGA ball-out, but shielding pins used for preventing pin-to-pin crosstalk were not considered. Moreover, when they try to keep the package cost small, this style will put a restriction on the maximum package size. Thus, there is a limit to the number of BGA balls that can be used for power delivery, and area for power delivery from motherboard to package. [9] proposed an algorithm which assigned and routed the solder bumps of a BGA package to a set of fanout points in a single layer. This work only created a topological routing, not precise geometry layout, and only the routability issue on PCB is considered.

For pin assignment problem, [10] presented a simulated annealing algorithm to find a pin assignment solution which considered the routability issue on BGA package and PCB, but no other DSM effects were considered. [11] suggested a direction of research for topological pin assignment. The two-stages heuristic algorithms, initial pin assignment and assignment improvement, can be closely attuned to a specific

router, then enhance the routability of PCB by reducing wiring congestion and path crossovers. Since this methodology disregarded the package pin number, it can not be applied to assign the pin-out of flip-chip BGA package which has large number of pins and significant cost issue. All these researches are not suitable for modern package-board codesign, which is requested to have minimal turn around time and optimized signal performance as well as package cost.

Figure 2.1(a) shows the typical interface design flow for IC-package-PCB codesign. In general, IC designers finish the pin designation based on experience (rule-of-thumb). In order to tradeoff signal performance and package cost, they always take few weeks to modify package size, rework package substrate and PCB layout, then rearrange pin-out. This conventional process can not efficiently estimate an accurate package size during designating pins for flip-chip BGA and possibly degrade signal performance due to the weakness on product experience and basic design concept. Furthermore, the costly refinement mentioned above constantly postpone the schedule of chip implementation, thus lengthen the time to market (TTM). Figure 2.1(b) illustrates the proposed design flow. First of all, we will create the rough pin configuration, which includes only four essential parameters. We automatically determine the pin-block order through an intuitive manner and flexibly design an appropriate pin pattern by solving integer linear programming (ILP) problems. After finishing the pin configuration, the designers will obtain the optimized package size and die size by automatically designating pin-out and locating I/O buffers.

This chapter aims at presenting a novel approach to designating pin-out and replacing heavy-loaded human design by automation process, which accounts for practical experience and techniques. Therefore, the detailed problem definitions are as follows:

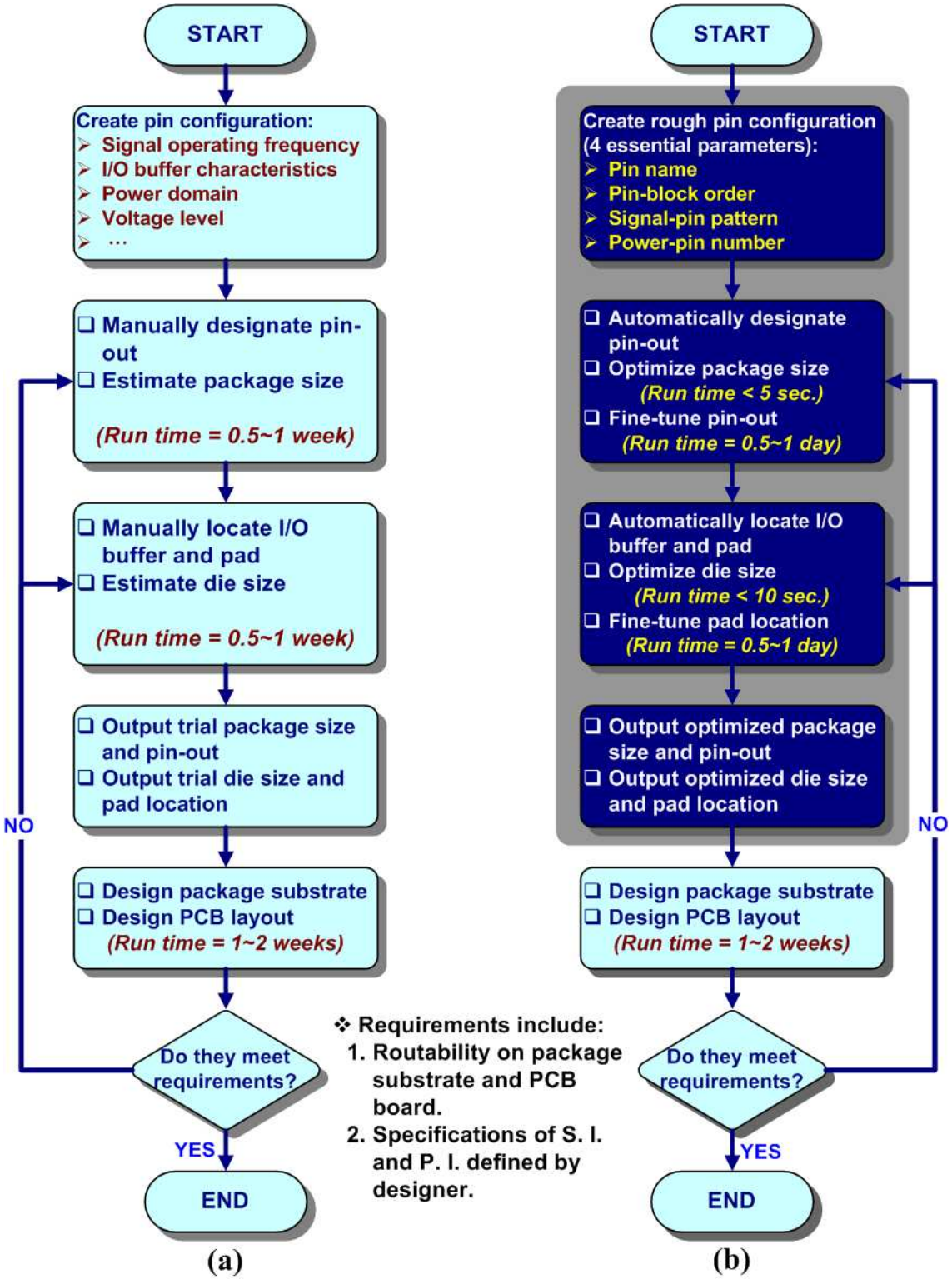


Figure 2.1: (a) is the typical flow and (b) is the proposed approach in interface design planning for IC-package-board codesign. Our approach not only automates pin-out designation efficiently, but also optimizes package size during design stage, thus reducing the time of iteration.

Input:

- The given pin list described pin connections between target package and corresponding PCB components.
- The rough corresponding PCB components' layout.
- The attributes of each I/O pin.

Output:

- The evaluated package size and designated pin-out for target package.

Pin-out criteria:

- The minimized package size.
- The optimized signal integrity, power delivery and PCB routability.

In this chapter, we have formulated feasible constraints for automatically designing pin patterns, which are used to assign the signal pins along the particular constraints and work as templates. And then we proposed six signal-pin patterns for pin-block construction in package design. Signal integrity, power delivery, and routability have been accounted for in those patterns. This helps to speed up the process of pin-out designation. Furthermore, we have proposed a near optimal approach to minimizing package size by mathematical (linear) programming formulation. The package size migration issues are also considered through a simple estimation. The experimental results show that our solution can achieve almost the same results as manually designed by experienced designers, with much less time.

2.2 Pin-Out Designation by Considering Signal Integrity and Power Delivery in Package-Board Codesign

Several critical constraints and considerations need to be taken care of whole designating package pin-out. In this section, we will discuss them comprehensively then introduce the design of pin pattern which can take all the constraints and considerations into account.

2.2.1 Constraints and Considerations

Locations of PCB Components

Figure 2.2 depicts a sketch of PCB layout. Usually PCB board contains several kinds of components and connectors which are applied to specific interfaces. The length of signal net from package pin to component or connector on PCB is the primary contributor to parasitic inductance. Therefore, package pins will exacerbate simultaneous switching noise (SSN) by increasing the parasitic inductance in the signal nets [12]. The familiar equation shown below describes the basic mechanism of SSN (V_{SSN}):

$$V_{SSN} = NL_{tot}(dI/dt) \tag{2.1}$$

where N is the number of switching drivers, L_{tot} is the equivalent inductance in which current must pass through, and I is the current per driver. In order to minimize the physical length of the signal net and thus reduce the total parasitic inductance, package pins should be accommodated in particular regions. As shown in Figure 2.2, the minimum net-length can be obtained by assigning the order of pin-blocks according to the certain location of corresponding components or connectors then fine-tuning the direction of package properly.

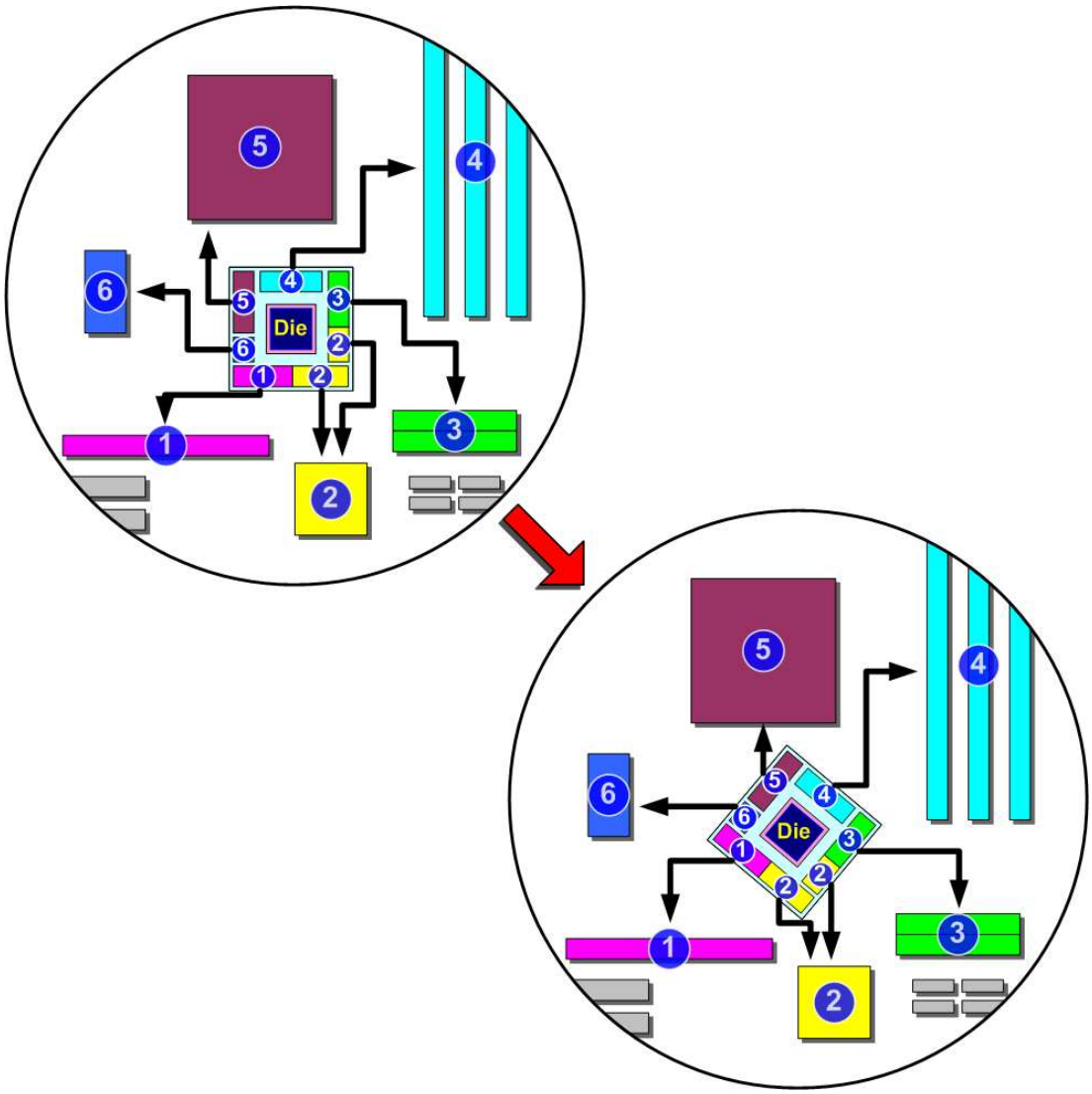


Figure 2.2: A general layout of PCB board. The order of pin-blocks on IC package should be assigned according to the corresponding components then fine-tuned the direction of package to meet minimum net-length.

Routability

Another crucial factor of successful pin designation is routability. For routing issue, the inflexible package-board routing rules force the row number of signal pins, signal net width and spacing on PCB to be critical constraints. Figure 2.3 shows the simplified cross-section of a flip-chip package which is mounted on PCB board. For a general 4-layer PCB board, only the top and bottom layers are allowed to be routed nets; the second and third layers are used for planning power/ground plane. Based on the rules of thumb, package outer pins (solder balls located close to the package edge) connect solder bumps through vias and package top layer routing. These outer pins are then inevitably routed on PCB top layer.

On the other hand, package inner pins located around the core of package must connect solder bumps by package bottom layer routing and then are routed on PCB bottom layer. Figure 2.4 demonstrates the routing pattern on PCB top layer and package bottom layer respectively. For instance, when the diameter of PCB pad is 14 mil (1 $mil = 25.4\ um$), pad pitch is 39.37 mil, signal net width and spacing are both 5 mil on a 4-layer PCB board, the space between two pads can only be penetrated by two nets. It means only three rows of signal pins can be fanned out nets on PCB top layer. Because of these routing rules, the excess row number of signal pins will undoubtedly cause routing congestion due to restricted area between pins. Figure 2.5 lists the confined row number of signal pins is constant and independent of the package sizes. In our example, the maximum row number of outer pins is nine and that of signal pins is seven (this happens when the nets on PCB bottom layer can be connected to four signal pins).

Signal Integrity

According to the routing pattern, shown in Figure 2.4, we can generalize the rule of thumb in assigning pins. That is, if signal pins are allocated on the same row,

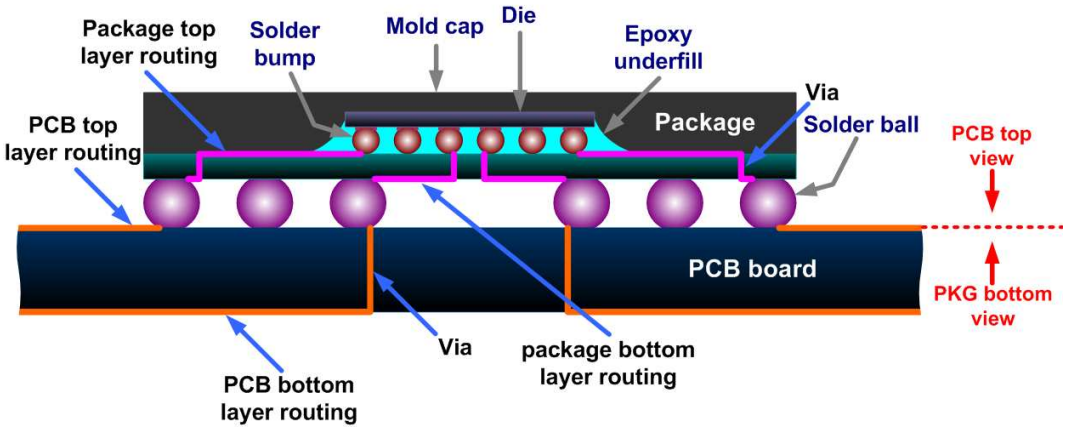


Figure 2.3: Simplified cross-section of a flip-chip package mounted on PCB board.

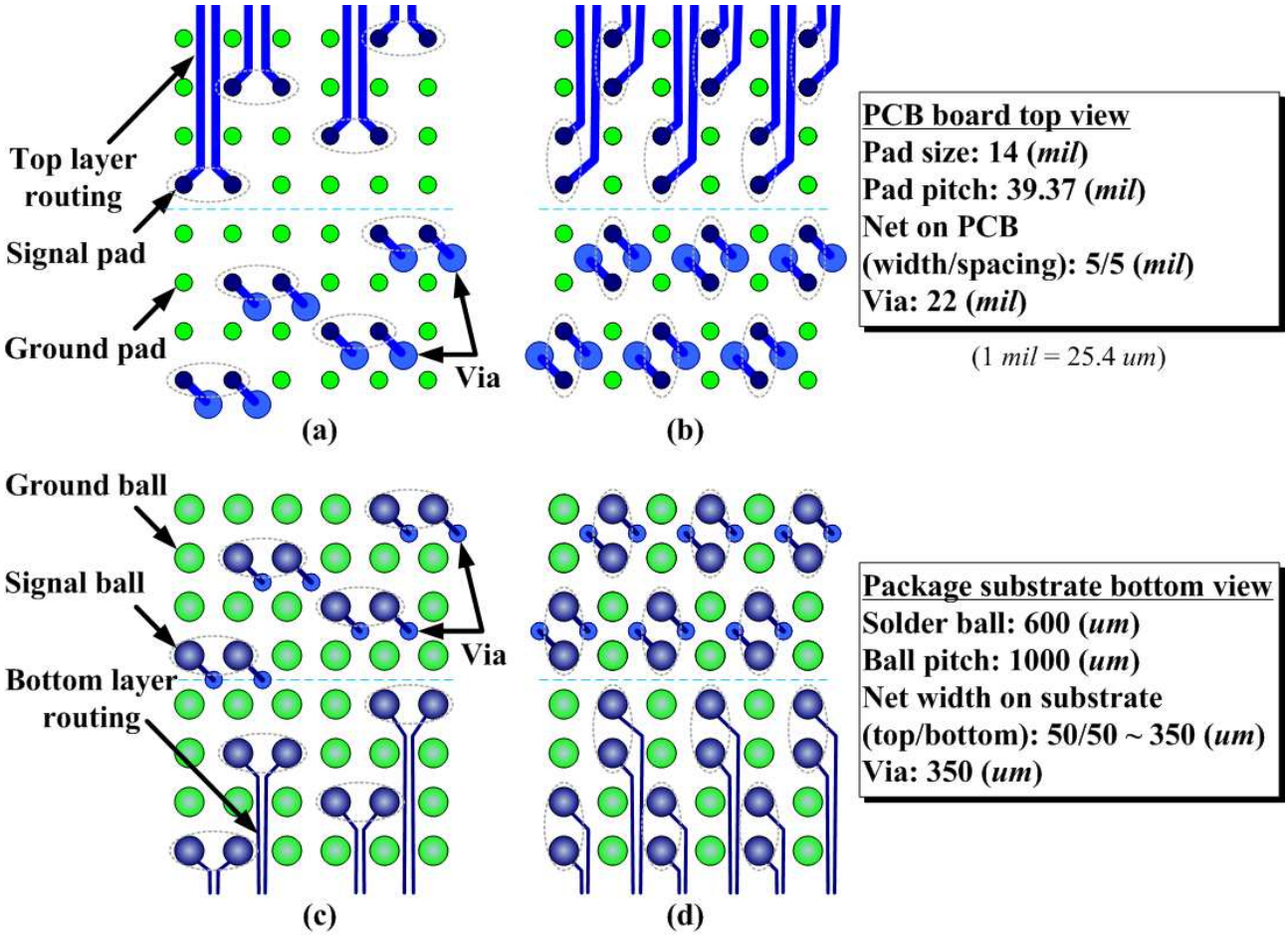


Figure 2.4: The routing pattern on PCB top layer (a)(b) and package bottom layer (c)(d). Because of the routing rules and restricted area between pins, the confined row number of signal pins is six. The excess row number of signal pins will cause routing congestion during the package substrate and PCB routing phase.

Package size (mm) (Width x Height)	Pin number (Row x Column)	Row number of outer-pin (power-pin, ground-pin and signal-pin)		Row number of outer-pin (signal-pin only)	
		Max.	Avg.	Max.	Avg.
37.5 x 37.5	36 x 36	9	8	7	6
35 x 35	34 x 34	9	8	7	6
31 x 31	30 x 30	9	8	7	6
27 x 27	26 x 26	9	8	7	6
...	...	9	8	7	6

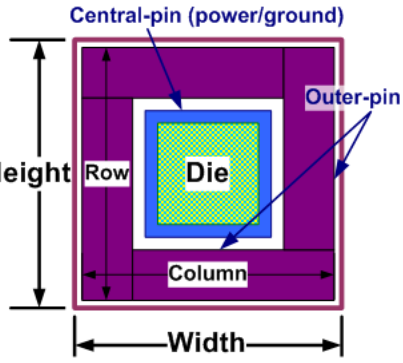


Figure 2.5: The restricted row number of signal-pin is constant and independent of package size due to inflexible package-board routing rules (PCB pad=14 mil, pad pitch=39.37 mil, net width=5 mil, net spacing = 5 mil, for four layer PCB board).

their nets can have balanced routing, which means these nets will have matched impedance on PCB and package layout. On the other hand, if signal pins are allocated on the same column, only some nets can have balanced routing. The matched impedance is an essential requirement for high-speed differential systems, because it can eliminate the common mode noise thus improve the signal performance. For signal integrity reason, return path inductance is another main course. The unsuitable placement and number of return path pins, which are power or ground pins, will maximize current return loops and increase return path inductance. This will dramatically degrade signal integrity and exacerbate radiated emissions. Its mechanism is similar to that of SSN and has been shown in equation (2.1).

With regard to crosstalk noise, one of the major root causes is mutual capacitance [12], mainly because it will inject a current onto the neighbor victim pins. The induced noise (I_{noise,C_m}) is proportional to the mutual capacitance (C_m) and the rate in change of voltage on driven pins (dV_{driver}/dt):

$$I_{noise,C_m} = C_m(dV_{driver}/dt) \tag{2.2}$$

Therefore, the optimal pin designation is to place signal pin and power/ground pin proximally close to each other, so that each signal pin can be tightly coupled to

a return path pin. This will minimize the effect of the return path inductance. Furthermore, if signal pins surrounded with ground pins, the mutual capacitance will be decreased and the noise is shielded extremely. In [13, 14, 15], the effects of shielding, return path and reference plane are considered in package and PCB designs. However, those optimized designs, in terms of signal integrity concern, will create signal-pin blocks which have more power/ground pins but fewer signal pins within a large block area. The feasible designs of pin pattern are proposed in the next subsection.

2.2.2 Pin Pattern Design

In order to automatically and flexibly design an appropriate pin pattern, we formulate the design constraints discussed in the previous subsection as feasible ILP problems. The notation used in the ILP formulation are as follows.

1. PA_i: i_{th} signal-pin pattern.
2. $p_{j,k}$: pin on j_{th} row and k_{th} column of PA_i.
3. C_k: the signal pin capacity on k_{th} column of PA_i.
4. D_j: the differential signaling constraints on j_{th} row of PA_i.
5. SN_i: the signal-pin number of PA_i.
6. SRR_i: the ratio of signal-to-return path pin in PA_i.
7. SSR_i: the ratio of signal-to-shielding pin in PA_i.
8. RPT_i: the type of return path pin in PA_i.

Therefore, we can obtain the proper pin patterns after solving the following ILP problems:

$$p_{j,k} = \begin{cases} 1 & \text{for signal pins} \\ 0 & \text{for power/ground pins} \end{cases}, \quad \forall PA_i \tag{2.3}$$

$$\sum_{j=1}^{row} p_{j,k} \leq C_k, \quad \forall PA_i, C_k \in \mathbf{N} \tag{2.4}$$

$$\sum_{k=1}^{col} p_{j,k} + \sum_{k=1}^{col} p_{j,k} \cdot p_{j,k+1} = D_j, \quad \exists PA_i, D_j \in \mathbf{N} \tag{2.5}$$

$$\frac{SN_i}{col \cdot row - SN_i} \leq SRR_i, \quad \forall PA_i \tag{2.6}$$

$$\frac{p_{j,k}}{4 - (p_{j+1,k} + p_{j-1,k} + p_{j,k+1} + p_{j,k-1})} \leq SSR_i, \quad \exists PA_i \tag{2.7}$$

$$RPT_i = \begin{cases} 1 & \text{for using power pins} \\ 0 & \text{for using ground pins} \end{cases}, \quad \forall PA_i \tag{2.8}$$

where $SN_i (= \sum_{k=1}^{col} \sum_{j=1}^{row} p_{j,k})$, row and col are the signal-pin number per pattern, row number and column number of a pattern respectively.

Equation (2.4) is the signal pin capacity (C_k). It confines the signal pin number within a column for all patterns (PA_i). As our previous discussion, the average number of this value is six. Equation (2.5) is the differential signaling constraints (D_j). The differential signal pins which exist in specific patterns must be strictly assigned at adjacent location in the same row (e.g., $p_{j,k+1}$=1, iff $p_{j,k}$=1). Equation (2.6) is the ratio of signal-to-return path pin (SRR_i). The return path pins play an import role in signal integrity considerations, designers must define the essential ratio for each pattern according to its applications. Equation (2.7) is the ratio of signal-to-shielding pin (SSR_i). For the purpose of isolating cross-talk noise, designers can set higher ratio of signal-to-shielding pin to assign ground pin in the neighboring location of signal pin. Otherwise, the ratio can be disregarded for low cost consideration. Obviously, these two ratios SRR_i and SSR_i will significantly trade off the performance and cost when we are designing pin patterns. Equation (2.8) is the type of return path pin (RPT_i). Once the type of return path pin match that of PCB reference plane, the return path will induce the lower parasitic inductance [12]. Hence, this constraint should be defined along the type of reference plane (power/ground) on PCB.

Table 2.1: The constraints of proposed pin patterns.

		C_k	D_j	SRR_i	SSR_i	RPT_i
Pattern 1 (PA_1)	PA_{10}	6	3	1/2	1/3	0
	PA_{11}	6	3	1/2	1/3	0
Pattern 2 (PA_2)	PA_{20}	6	3	1/2	1/3	0
	PA_{21}	6	N/A	1	1/3	0
Pattern 3 (PA_3)	PA_{30}	6	N/A	1	1/3	0
	PA_{31}	6	3	1/2	1/3	0
Pattern 4 (PA_4)	PA_{40}	6	N/A	1	1/3	0
	PA_{41}	6	N/A	1	1/3	0
Pattern 5 (PA_5)	PA_{50}	6	N/A	1	1/3	1
	PA_{51}	6	N/A	1	1/3	0
Pattern 6 (PA_6)	PA_{60}	6	N/A	3	N/A	1
	PA_{61}	6	N/A	3	N/A	0

For two layers PCB routing, Table 2.1 proposes six sets of constraints for generating six options of signal-pin patterns (PA_{i0} and PA_{i1} represent the fore-half and back-half of patterns). There exists tradeoff between signal performance and package cost. Figure 2.6 illustrates these proposed pin patterns and their simplified impedance models. The impedance of each net is composed of three components: serial resistor, serial inductor and shunt capacitor ($Z_L = R+j\omega L+1/j\omega C$). The first signal-pin pattern depicts that each pair of differential signal has been surrounded by ground pins. These ground pins can be performed as adjacent return path pins to minimize total inductance and as shielding pins to isolate pin-to-pin crosstalk noise. Moreover, the primary concern of differential system is on impendence-matching of nets. The first pattern has an exclusive advantage of nets balancing on PCB as well as package substrate layout, shown in Figure 2.4(a) and (c). Thus it is optimal for differential signals from the performance perspective, and can be modeled by two nets with matched impedance Z_L, as shown in Figure 2.6. The only disadvantage of this pattern is poor pin designation efficiency.

In most cases, if the return current of a signal pin flows on ground planes, it should be coupled to ground pins to result in minimum return path, or vice versa.

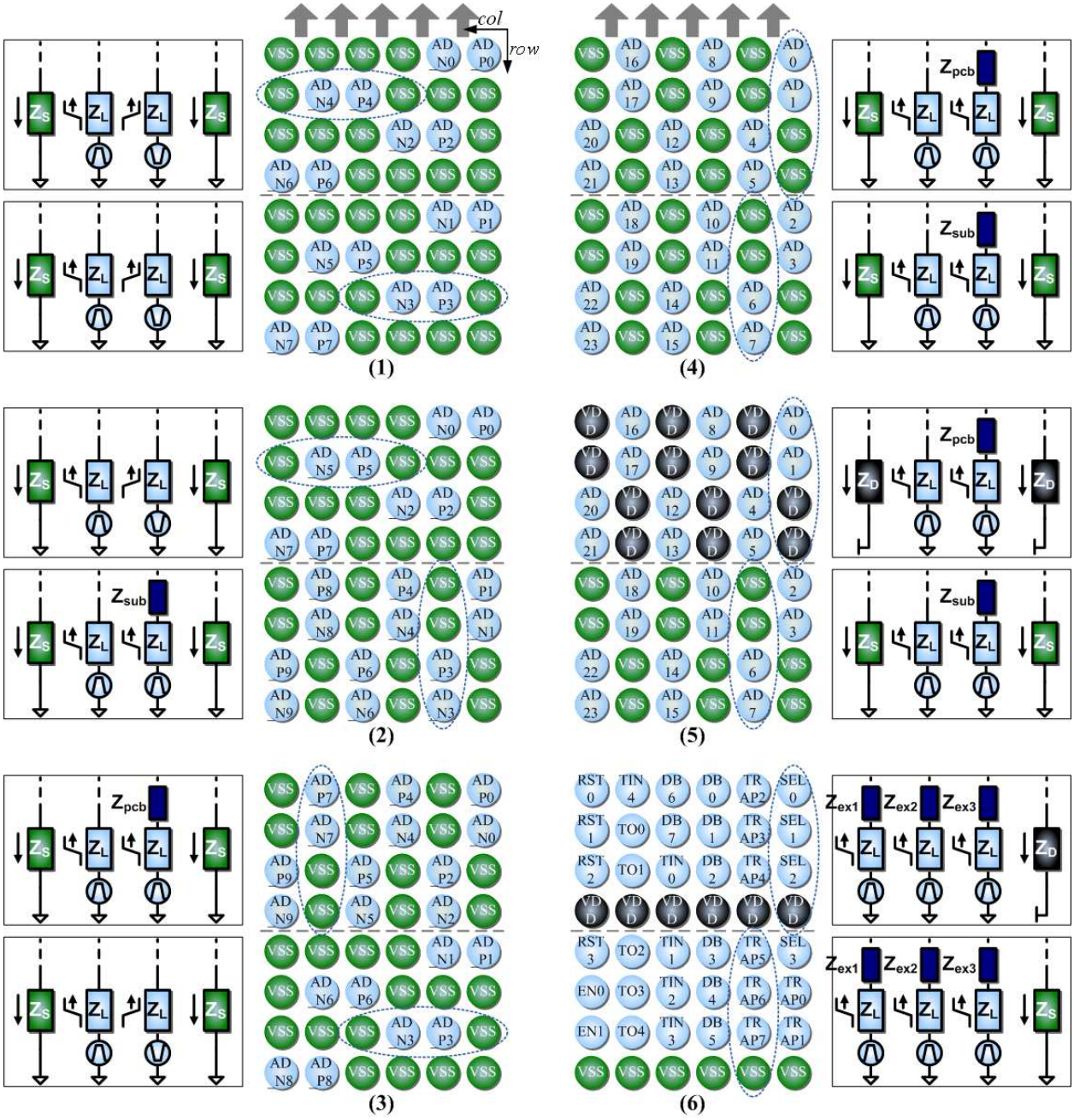


Figure 2.6: Six pin patterns proposed in our methodology. There exists tradeoff between routability and signal integrity concerns. The first pin pattern has better signal integrity, while the sixth one has the most efficient pin designation. In those patterns, AD_P0/AD_N0 is for differential signal (high speed), AD is for single-ended signal (high speed), and SEL or TRAP in sixth pattern are for single-ended signal (low speed or long-pause signal).

Whether a signal is coupled to just one power pin or just one ground pin, this case will emerge from the particular signal type and its configuration. Therefore, the fourth and fifth signal-pin patterns are proposed to provide two options for specific bus. The fifth pattern has better power delivery characteristic than the fourth one because of locating power pins. These two patterns arrange pins more efficiently than first pattern, but they both have worse signal integrity on PCB top-layer-routing and package bottom-layer-routing due to poor impedance-matching, shown in Figure 2.4(b) and (d). Hence, the net of each signal pair in its model has additional impedances except Z_L on PCB board (Z_{pcb}) or on package substrate (Z_{sub}). Both of them include extra equivalent resistance, inductance and capacitance. As compared with above-mentioned patterns, the second and third patterns are the compromises between signal performance and package cost. As for the sixth signal-pin pattern, it is the most efficient pin designation among all patterns because it contains the most signal pins than other patterns. The major disadvantage of this pattern is that it ignores all signal integrity concerns and can only be applied to test-in, test-out or long-pulse control signal, which has less sensitivity in crosstalk. Therefore, its impedance model depicts these characteristics by using an undesirable and unpredictable impedance Z_{ext}, which is induced from PCB board and package substrate.

According to the experiences and basic concept of signal integrity, these six patterns have been characterized and shown in Figure 2.7. Designers can take these patterns as templates and easily choose a specific pattern along the specification of individual bus, or they can design pin patterns which has sensible efficiency, routability and signal integrity for their specific purposes by defining their dedicated constraints.

	Application	Signal-pin NO.	Pin-to-pin crosstalk immunity	Net balance				Signal shielding on package substrate (VDD/VSS)		Power delivary aware	Pin-designation efficiency
				PCB board		Package substrate					
				Top layer	Bottom layer	Top layer	Bottom layer	Top layer	Bottom layer		
Pattern 1	Differential signal	16	Excellent	Good	Good	Good	Good	VSS	VSS	Without	Not good
Pattern 2	Differencial signal / Single-ended signal	20	Good	Good	Good	Good	Not good	VSS	VSS	Without	Average
Pattern 3	Differencial signal / Single-ended signal	20	Good	Not good	Good	Good	Good	VSS	VSS	Without	Average
Pattern 4	Differencial signal / Single-ended signal	24	Excellent	Not good	Good	Good	Not good	VSS	VSS	Without	Good
Pattern 5	Differencial signal / Single-ended signal	24	Excellent	Not good	Good	Good	Not good	VDD	VSS	With	Good
Pattern 6	Single-ended signal	36	Not good	Not good	Not good	Not good	Not good	None	None	With	Excellent

Figure 2.7: The characteristics of signal-pin patterns. According to the properties and requirements of specific signal, we can select a proper pattern to designate pins.

2.3 Fast Pin-Out Designation Automation by Pin-Block Construction and Floorplanning

By using those pin patterns, pin-blocks can be constructed and grouped for pin-out designation. In addition, package size will be minimized by pin-block floorplanning. This section presents the detailed strategies and methodologies. The package size migration issues will be considered in this section as well.

2.3.1 Pin-Block Construction and Grouping

In general, designers always take half or one day to specify the pin configuration for high pin-count chip. It is because the most precise pin configuration will contribute the optimal pin-out and package size in manual design, but it is an exhaustive and time-consuming work. In proposed approach, the runtime of this manual job can be reduced by a rough pin configuration which simply contains four essential parameters: signal-pin name, pin-block placement sequence (order), selected signal-pin pattern and the number of power-pin, as shown in proposed design flow. First of all, we automatically create the pin-block placement sequence via a simple way. As long as we obtain the rough coordinate of each corresponding component, the pin-block placement sequence will be determined by an intuitive manner of enumer-

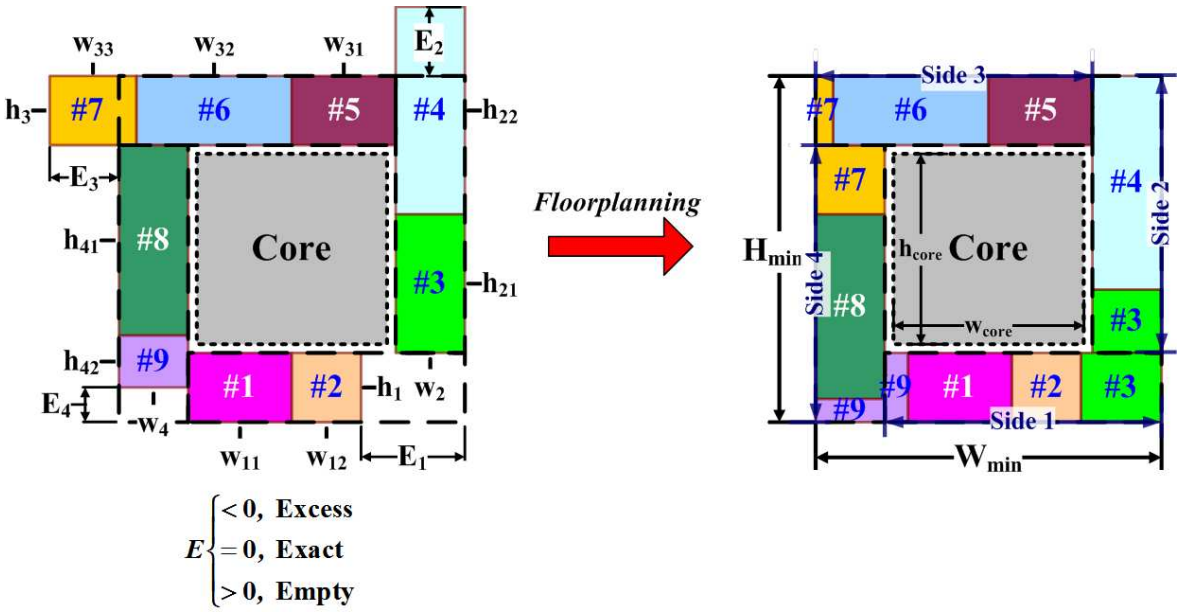


Figure 2.8: A minimum package size can be obtained after we designate and floorplan all pin-blocks.

ating components clockwise (or counterclockwise). And then, we flexibly design and select an appropriate pin pattern as describes in Section 2.2.2. According to the signal-pin name and selected signal-pin pattern, we can automatically construct all signal-pin blocks by locating signal pins within a block along the specific patterns.

The number of power-pin can be used to deal with the power delivery issue. Our strategy is to establish a power-pin block which can provide a power channel on PCB for various power domains. Designers can freely define the demand of power pins for individual signal configuration relying on the power analysis result. While the signal-pin block is constructed, the proposed automation approach will create power-pin block and place it adjacent to the related signal-pin block, then integrate them into single block for a signal bus. Figure 2.8 shows an example, nine pin-blocks (♯1 to ♯9) are constructed for nine different interfaces in a package. Finally, the pin-block placement sequence is applied in pin-block grouping strategies which divide all pin-blocks into four group and place on each package side in the next stage, shown in Figure 2.8.

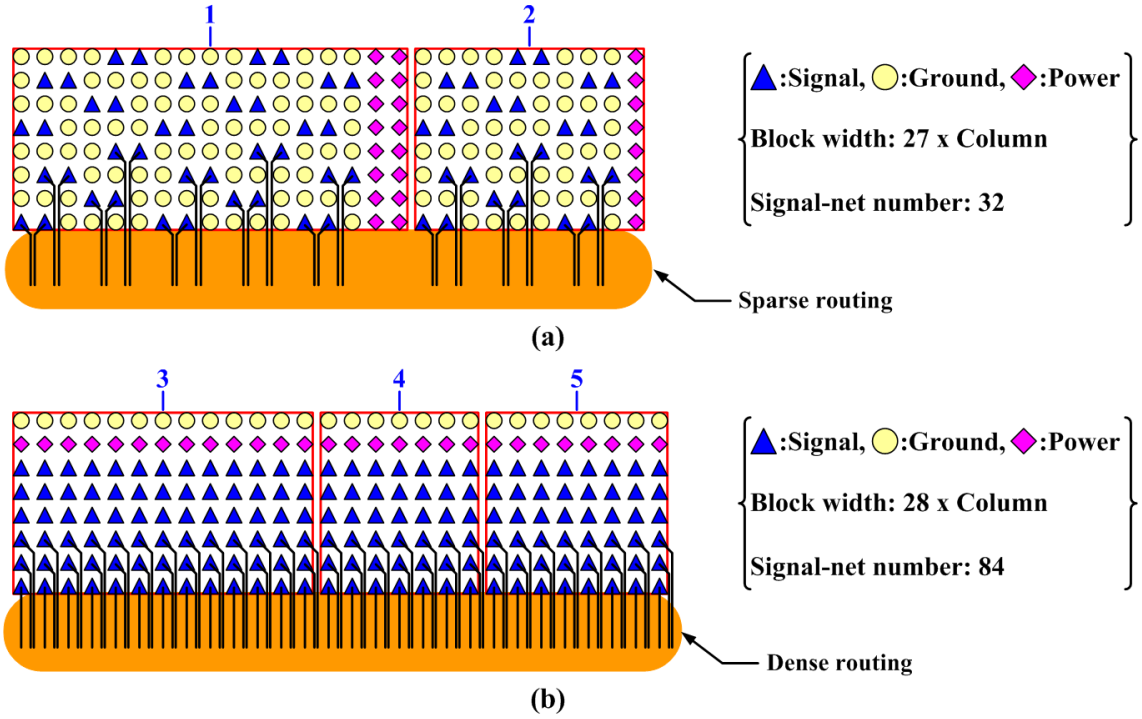


Figure 2.9: The boundary-constrained pin-block grouping strategy (BCPG): all pin-blocks will be grouped into single block in proper order until the integrated block size locate within the safe range. The size of grouped pin-block (a) and (b) are closed to each other. The disadvantage of this method is that it possibly causes the dense routing likes (b).

We have further developed two strategies for grouping pin-blocks into package boundaries, the boundary-constrained pin-block grouping strategy (BCPG) and the congestion-aware pin-block grouping strategy (CAPG). When we design the pin-out for chipset, which acts as a bridge of all components on motherboard, the location of component is one of major constraints presented in Section 2.2.1. Since the locations of components on PCB are boundary-constrained, the grouping strategy BCPG (shown in Figure 2.9) will be applied. We defined the safe range for this method:

$$\phi_1 \cdot AVG_s \leq S_m \leq \phi_2 \cdot AVG_s \tag{2.9}$$

where S_m is the size of grouped block, ϕ_1 and ϕ_2 are user-defined parameters, $AVG_s = (\sum_n w_n)/4$ is the average block size and w_n is the width of each block.

Equation (2.9) shows that the main concern of this methodology is pin-block size. According to the pin-block placement sequence determined in pin configuration, the pin-blocks will be grouped into single block in proper order until the integrated block size locate within the safe range. For this strategy, the size of each grouped block is closed to the average block size then result in minimized E_i value (shown in Figure 2.8) on each side. Therefore, this method will speed up the runtime of minimizing package size. However, the BCPG will introduce the possibility of generating a dense net-routing due to the disregard of signal-pin number. As shown in Figure 2.9, the two grouped pin-blocks (one is grouped with block 1 and 2, the other is grouped with block 3, 4 and 5) have closed block width, but they have very different signal-net number. The worse case shown in Figure 2.9(b) will decrease the routing efficiency on PCB layout and increase the implementation cost for PCB.

The another strategy is CAPG whose primary consideration is to equally distribute signal-pins on each package side. Consequently, the PCB layout will effortlessly lead to a loose density and have more flexibility to match the impedance of critical nets or adjust the location of components. Figure 2.10 shows an example, the signal-pin number of integrated blocks will be close to each other when we adopt CAPG strategy. Therefore, this method is suitable for the package design of field programmable gate array (FPGA) which has the prior concern of routability. For CAPG, it will consider the signal-pin number instead of the placement order or side of each pin-block, hence the locations of PCB components will be determined after it accomplishes the final pin-out. The safe range used for this strategy is also defined below:

$$\psi_1 \cdot AVG_p \leq TP_i \leq \psi_2 \cdot AVG_p \tag{2.10}$$

where TP_i is the total signal-pin number of grouped block, ψ_1 and ψ_2 are user-defined parameters, $AVG_p = (\sum_j p_j)/4$ is the average signal-pin number and p_j is the signal-pin number of each block. Since the equalized signal-pin number is

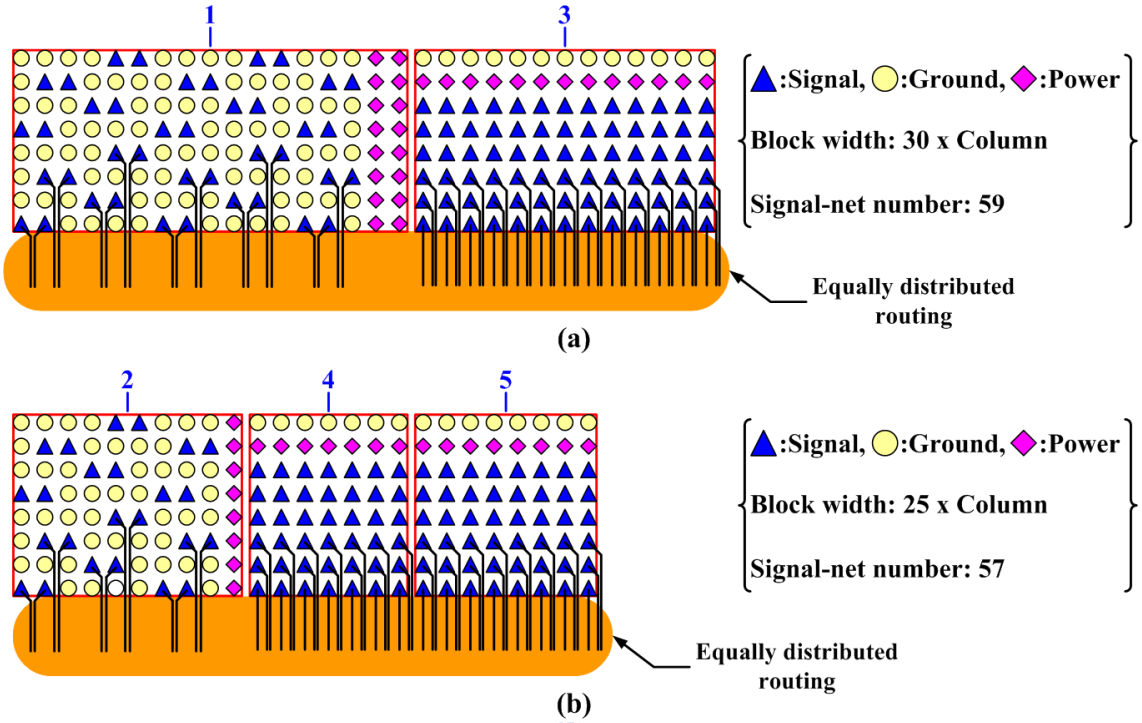


Figure 2.10: The congestion-aware pin-block grouping strategy (CAPG): the first prior consideration is to equalize the signal-pin number. The signal-pin number of grouped pin-block (a) and (b) are very closed. The disadvantage of this method is that the arrangement of PCB components must be restricted and referred to the final pin-out.

usually larger than the size of grouped block, the second method must has tighter safe range (e.g., $|\psi_1 - \psi_2| < |\phi_1 - \phi_2|$) to achieve the same boundary range as that of the first one.

To implement the strategies of BCPG and CAPG, because the pin-block placement sequence will be considered primarily in BCPG, we will use the first-fit heuristic algorithm which is an approximation algorithm for solving bin-packing problem to group pin-blocks. This algorithm sequentially assigns objects into the first bin, and then creates a new bin when the current bin is full. For CAPG, the first prior consideration is to equalize the signal-pin number. Therefore, we can apply another bin-packing approximation, the best-fit heuristic algorithm, to group pin-blocks. This heuristic ignores the order of objects and fills all objects into the feasible bins,

which have the smallest residual capacity.

From the observations in Section 2.2, signal integrity, power delivery and routability issues should be accounted for in general cases when signal pins are placed. After finishing the implementation and placement of all blocks, a rough pin designation can be obtained, shown in Figure 2.8. At the same time, E_1 to E_4 can be evaluated from this rough pin designation (E_1 to E_4 represent the width/height of empty or excess area in each side of minimum package). These E values will be used for package size minimizing and pin-block floorplanning in the next section.

2.3.2 Package Size Minimization and Pin-Block Floorplanning

The next step is to optimize package size and acquire a feasible pin designation by solving mathematical (linear) programming formulation. The notation used in the formulation are shown below.

1. w_{ji}: the width of i_{th} pin-block located at j_{th} side.
2. h_{ji}: the height of i_{th} pin-block located at j_{th} side.
3. E_j: the width/height of empty or excess area in j_{th} side.
4. W_{min}: the minimum package width.
5. H_{min}: the minimum package height.
6. w_{core}: the minimum core width.
7. h_{core}: the minimum core height.

Therefore, the objective function and constraints can be formulated as follows:

Minimize

$$f = \sum_{j=1,3}(\sum_{i} w_{ji} + E_j)h_j + \sum_{j=2,4}(\sum_{i} h_{ji} + E_j)w_j$$

subject to

$$W_{min} = w_4 + \sum_i w_{1i} + E_1 = w_2 + \sum_i w_{3i} + E_3 \tag{2.11}$$

$$H_{min} = h_1 + \sum_i h_{2i} + E_2 = h_3 + \sum_i h_{4i} + E_4 \tag{2.12}$$

$$W_{min} \geq w_2 + w_4 + w_{Core} \tag{2.13}$$

$$H_{min} \geq h_1 + h_3 + h_{Core} \tag{2.14}$$

$$W_{min} = H_{min}; w_{Core} = h_{Core} \tag{2.15}$$

$$E_1 + E_2 + E_3 + E_4 \geq 0 \tag{2.16}$$

where w_{1i},h_1,h_{2i},w_2,w_{3i},h_3,h_{4i},w_4 can be evaluated in the previous step, all shown in Figure 2.8. The *Core* is the center area of BGA package. In principal, the power and ground pins are located at the center of package and the die is located upon these power and ground pins. As a result, the heat generated from the die can be transferred out through these pins [16]. Thus increasing more power and ground pins located at the center area will improve heat dissipation but enlarge the area of *Core*, thereby enlarge the package size. We use (2.13) to (2.14) to define the area of *Core* in accordance with physical die size, where w_{Core} and h_{Core} are user specified parameters. If these two values are not given by designer, the minimum *Core* size can also be obtained when the minimum package size is evaluated. Considering the general industrial chipset designs, constraint (2.11), (2.12) and (2.15) will restrict the package shape to be square. The purpose of (2.16) is to insure that the minimum package size can accommodate all pin-blocks with almost no void pin positions.

After E_1 to E_4 are obtained, we can easily recognize the position of the empty and excess area in the evaluated minimum package. The final step of proposed methodology is to floorplan pin-blocks. Our method splits the pin-blocks in the excess area and fills them into the adjacent empty area. It completely eliminates exceed area and keep those pins being located around the particular region restricted in previous step. The algorithm of pin-block floorplanning is shown below:

1. $i \leftarrow 1$, $i \in 1, 2, 3, 4$ //*start from side 1*
2. $i - 1 \leftarrow 4$, iff $i = 1$; $i + 1 \leftarrow 1$, iff $i = 4$
3. **Repeat:**
4. while $(E_i \neq 0 \bigcap E_i < 0)$ do
5. if $E_{i-1} > E_{i+1}$
6. shift pins clockwise //*fill the pin-block into empty area in last side until the* E_i *is zero* ($E_i \leftarrow 0$, $E_{i-1} \leftarrow E_{i-1} + E_i$)
7. else
8. shift pins counterclockwise //*split the pin-block in excess area then group it into next side* ($E_i \leftarrow 0$, $E_{i+1} \leftarrow E_{i+1} + E_i$)
9. $i \leftarrow i + 1$ //*check next side*
10. **Until** all E values are large than or equal to zero

Figure 2.11 shows an example, where there are two excess areas occurred in second and third side (upper right and upper left corners) and two empty areas occurred in first and fourth side (bottom right and bottom left corners). According to the proposed algorithm, the pin-blocks located in side 1 will be skipped due to $E_1 > 0$ (line 4 in the above algorithm). While it considers the pin-blocks of side 2 (Figure 2.11(a)), some of the pins in group ♯3 will be clockwise filled into the empty area in side 1 ($E_1 > E_3$, line 5 and 6), and then it will consider the pin-blocks in next side. In side 3 (Figure 2.11(b)), because of $E_4 > E_2$ the pins of group ♯7 which excess the range of the side will be split and grouped into side 4 (line 8 and 9). Finally, in the last side the pins of group ♯9 are the same case as that of group ♯7

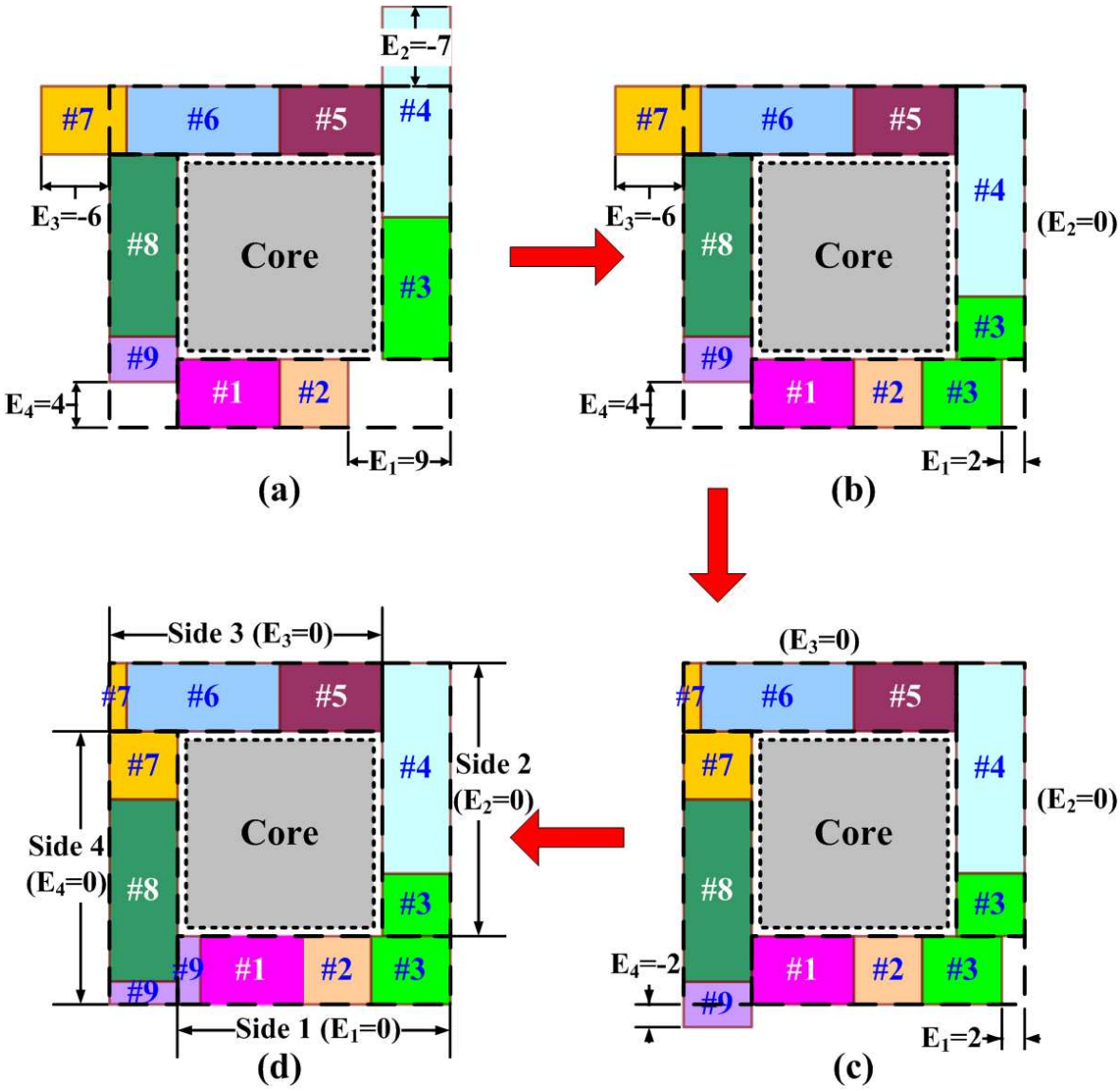


Figure 2.11: The example of pin-block floorplanning. The pins in the excess areas will be shifted into the empty areas through our floorplanning algorithm. A final pin-out can be acquired after finishing the package size minimization and pin-block floorplanning.

(Figure 2.11(c)) and will be floorplanned into the proper locations then acquire an optimized pin-block floorplanning (Figure 2.11(d)) through this simple procedure.

2.3.3 Dealing With Package Size Migration Issues

For practical application, designer usually need to migrate package size from larger to smaller or vice-versa. During chip prototyping, the extra I/O pins are required for monitoring test signals, and then the package size will be dynamically migrated to a larger one. In addition, when the products have cost margin for improving performance or adding the new features, the chip size and package size must be enlarged simultaneously. These requirements can be satisfied easily by changing types of pin pattern from higher SN_i (signal-pin number per pattern) to lower one, which increasing the width of pin-blocks. On the contrary, when the cost issue has higher priority then signal integrity or the die size has to be shrunk due to the removal of some features, the package size must be shrunk at the same time. Consequently, the types of pin pattern should be modified from lower SN_i to higher one, which increased the efficiency of pin designation but relaxed the performance constraints to acquire smaller pin-blocks.

To tackle these package size migration issues, we have defined a migration factor ($M.F.$) to evaluate the enlarged or shrunk column number (width) of pin-blocks during changing types of pin pattern. The migration factor can be simply calculated through the following equation:

$$M.F. = (-1) \cdot col \cdot (\frac{1}{SN_p} - \frac{1}{SN_m})$$

$$\Rightarrow \begin{cases} >0 & \text{for enlarging package} \\ <0 & \text{for shrinking package} \end{cases} \tag{2.17}$$

where col is the given column number of pin pattern, SN_p and SN_m are the signal-pin

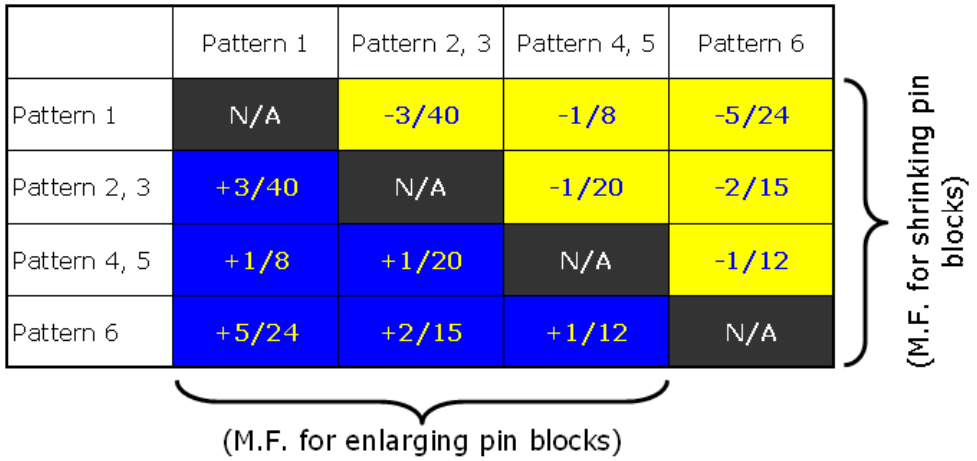


Figure 2.12: The enlarged and shrunk migration factor ($M.F.$) of six proposed pin patterns, designer can decide the modified pattern along these factors.

number per pattern in previous pattern type and modified pattern type. Figure 2.12 shows the migration factors of six patterns proposed in our methodology, where "+" means enlarged factor and "−" means shrunk factor. Therefore, the total pin number of a group multiplied by the migration factor will estimate the modified width of a pin-block. And then designer can decide which pattern should be modified along these estimations.

2.4 Experimental Results

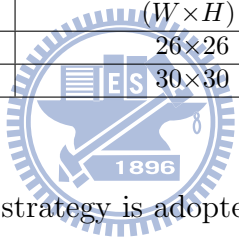
We have implemented the proposed methodology in C++ and the platform is on AMD Sempron 1.75GHz with 1GB memory. To solve the integer linear programming problem, the optimization package LOQO [17] is applied. We use two industrial mass production chipset cases as our benchmarks, the rough pin configuration charts are shown in Table 2.2. Table 2.3, Figure 2.14 and Figure 2.16 show the results of pin-out designation for these two benchmarks. Based on Table 2.3 which is obtained from linear programming formulation shown in Section 2.3.2, we can get corresponding parameters to floorplan all pin-blocks. The runtime of designating pin-out is less than 5 second for both cases.

Table 2.2: Two industrial benchmarks used in proposed methodology.

	Signal bus	Pin number	Order	Selected signal-pin pattern	Power-pin number
Case 1	Bus #1	66	1	4	32
	Bus #2	27	2	5	8
	Bus #3	37	3	6	8
	Bus #4	32	4	2	N/A
	Bus #5	40	5	2	16
	Bus #6	53	6	6	16
Case 2	Bus #1	66	1	4	24
	Bus #2	27	2	5	8
	Bus #3	95	3	4	N/A
	Bus #4	100	4	4	8
	Bus #5	42	5	6	8
	Bus #6	16	6	6	N/A

Table 2.3: The experimental results of Case 1 and Case 2.

	$\sum_i E_i$	Central P/G pins ($w_{Core} \times h_{Core}$)	Manually designed package ($W \times H$)	Evaluated min. package ($W \times H$)
Case 1	0	10×10	26×26	26×26
Case 2	3	14×14	30×30	31×31



For Case 1, when the BCPG strategy is adopted we can obtain very closed pin-block sizes in 1st and 3rd side (Figure 2.13(a)) and a rough pin-out in minimum package size (Figure 2.14(a)). Figure 2.14(b) shows our final pin-out designation is perfectly matched with manual design (Figure 2.14(c)) achieved by an experienced engineer, which spent long turn-around time to respin the design (usually weeks). By using CAPG strategy, we can equally distribute signal-pins on each side of package (Figure 2.13(b)), even though the pin-blocks need to be floorplanned (Figure 2.14(e)) the variation of signal-pin number is restricted (Figure 2.14(f)).

For Case 2, the same flow as that in Case 1 we choose BCPG strategy to group pin-block first (Figure 2.15(a), Figure 2.16(a)). Due to more pin numbers in some buses and signal-pin block pattern usage (while pin number is not divisible by eight for pattern (1), ten for pattern (2) and (3), twelve for pattern (4) and (5), six for pattern (6) will generate void position respectively), a slightly larger package

size (Figure 2.16(b)) is achieved, but still very close to the manual design (Figure 2.16(c)). Next, we use CAPG method to group pin-block (Figure 2.15(b)), the final pin-out (Figure 2.16(e)) shows the signal-pin number in each side of package is very closed to each other (Figure 2.16(f)). In addition to these two cases, we have tested a case which has 25 groups and 720 signal-pins, the runtime is still below 5 second and obtain the minimum package size as well.

In package size migration cases, we keep using those two industrial cases as the examples. By equation (2.17), we can firstly calculate the $M.F.$ as shown in Figure 2.12. And then, we multiply pin number in Case 1 and Case 2 by the $M.F.$ to obtain the results of Table 2.4. This table shows the enlarged or shrunk column number of pin-blocks when we are changing types of pin pattern for all groups in Case 1 and Case 2. For example, if we want to enlarge package size of Case 1 from 26 $mm \times 26$ mm to 30 $mm \times 30$ mm, we will need $(30 - 26) \times 4 = 16$ enlarged column number in pin-blocks. We can change the types of pin pattern in bus ♯1 to ♯3 and ♯6 into first, second or third pattern, which have good signal integrity to gain better performance. On the other hand, when we are trying to shrink package size of Case 2 from 31 $mm \times 31$ mm to 25 $mm \times 25$ mm, the shrunk column number will be $(30 - 26) \times 4 = 16$. The only one choice is to change types of pin pattern in bus ♯1 to ♯4 into sixth pattern. Since the most selected pin patterns have higher pin designation efficiency in this case. The margin of enlarging package size will be larger than that of shrinking size.

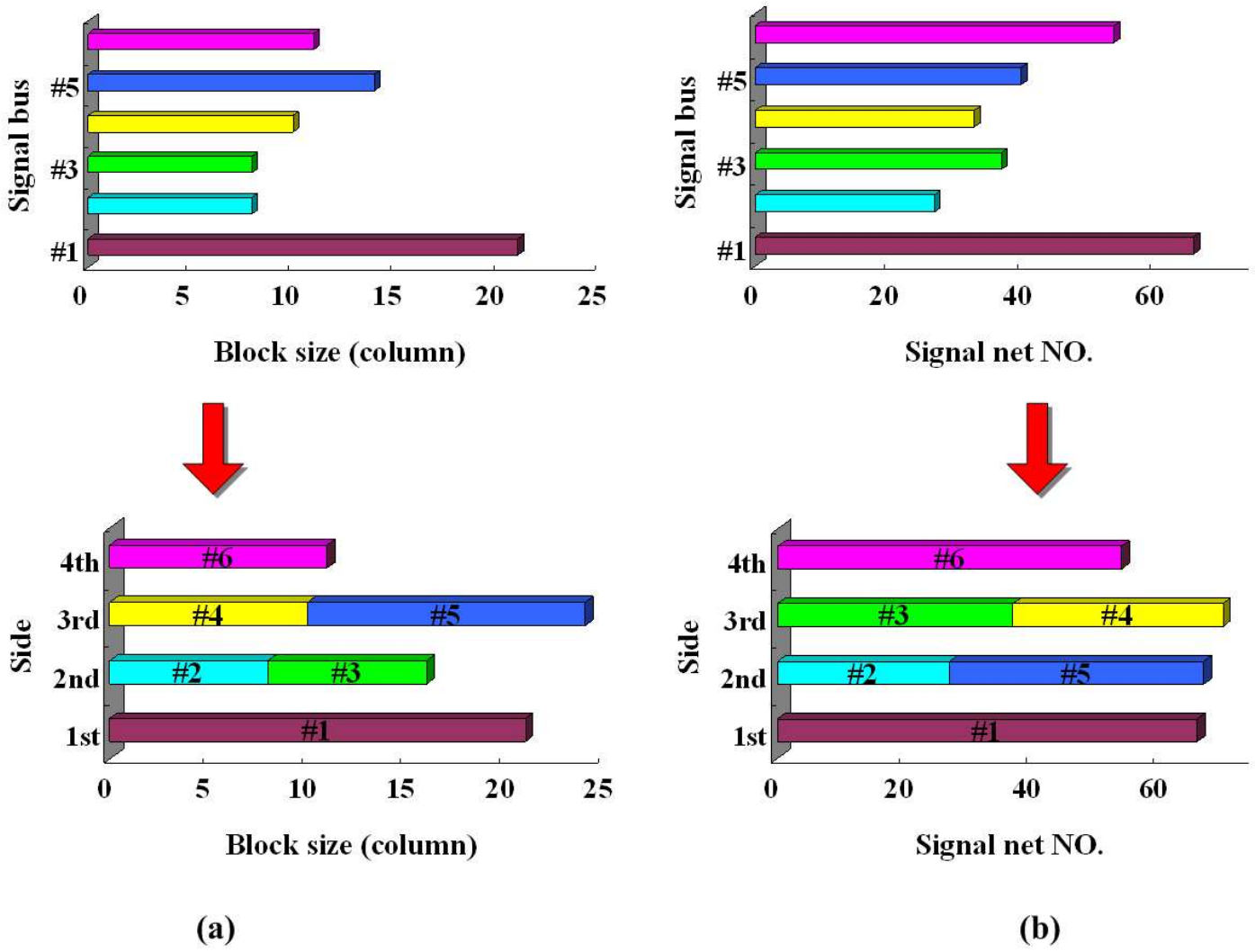
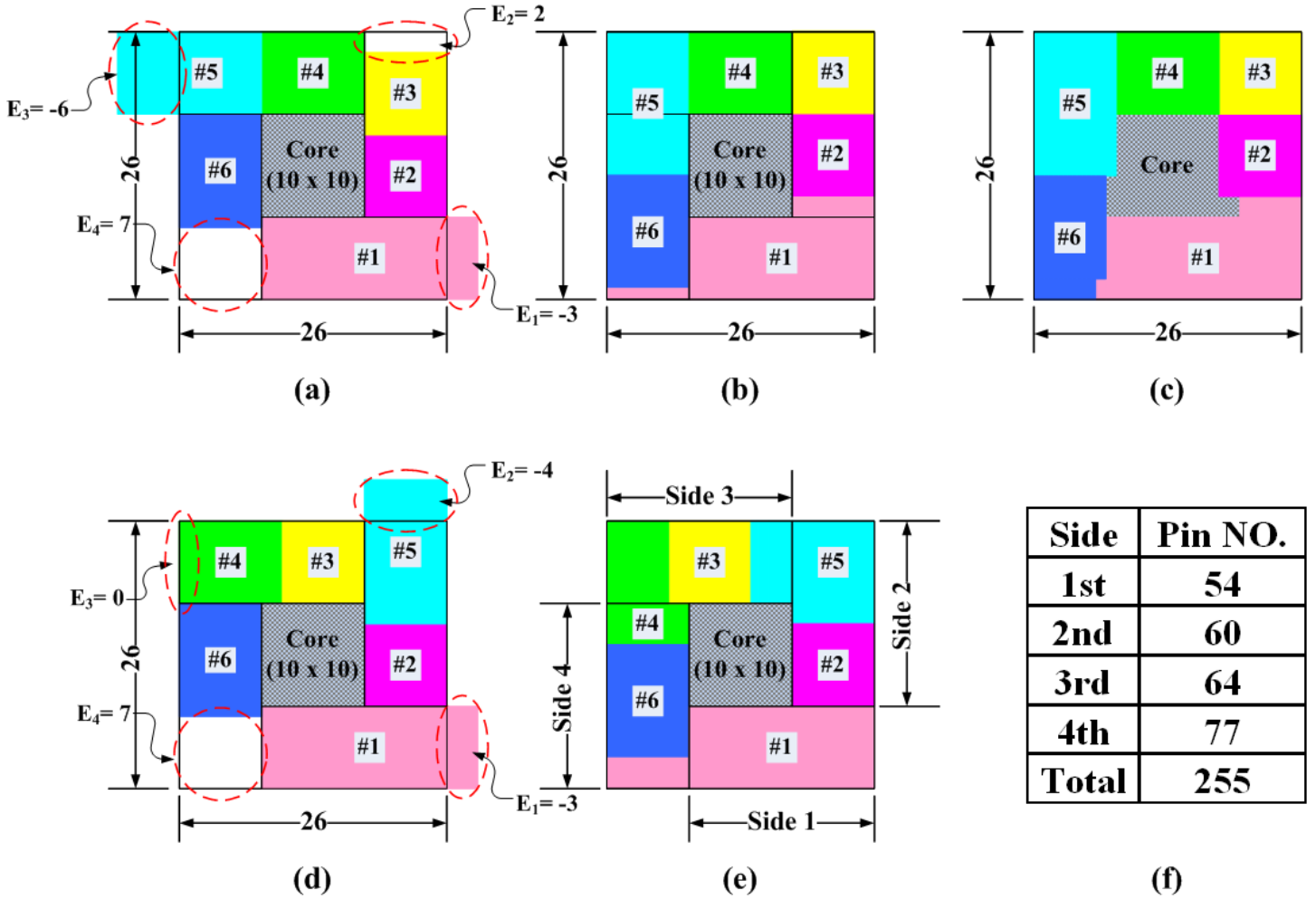


Figure 2.13: Pin-blocks grouping results of Case 1. (a) is grouping with BCPG (ϕ_1=0.5, ϕ_2=1.5), and (b) is grouping with CAPG (ψ_1=0.8, ψ_2=1.2).



Side	Pin NO.
1st	54
2nd	60
3rd	64
4th	77
Total	255

Figure 2.14: Experimental results of Case 1. (a) is placement of pin-blocks grouped with BCPG, (b) is final pin-out after (a) being floorplanned, (c) is manually designated pin-out, (d) is placement of pin-blocks grouped with CAPG, (e)is final pin-out after (d) being floorplanned and (f) is the equalized signal-pin number chart.

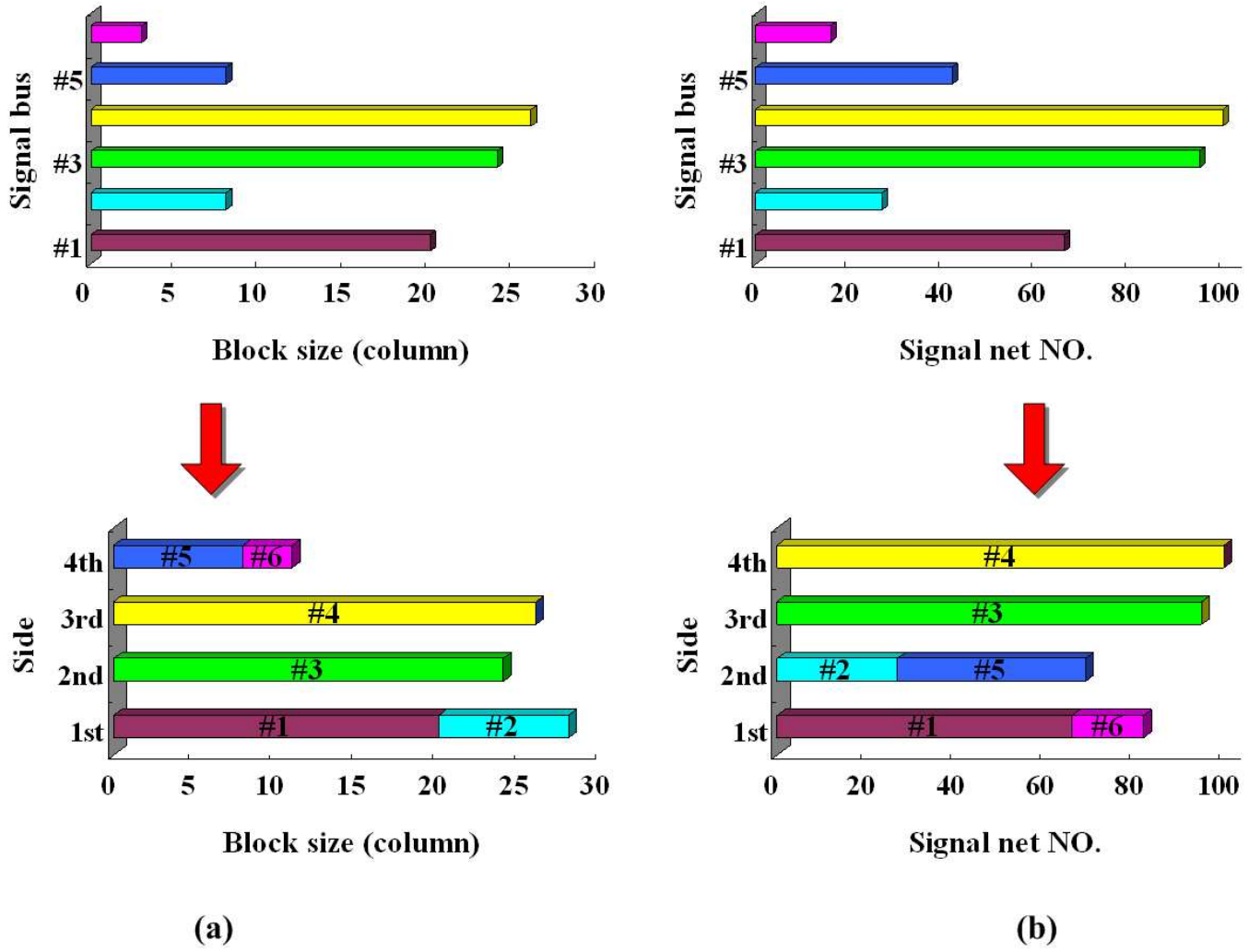
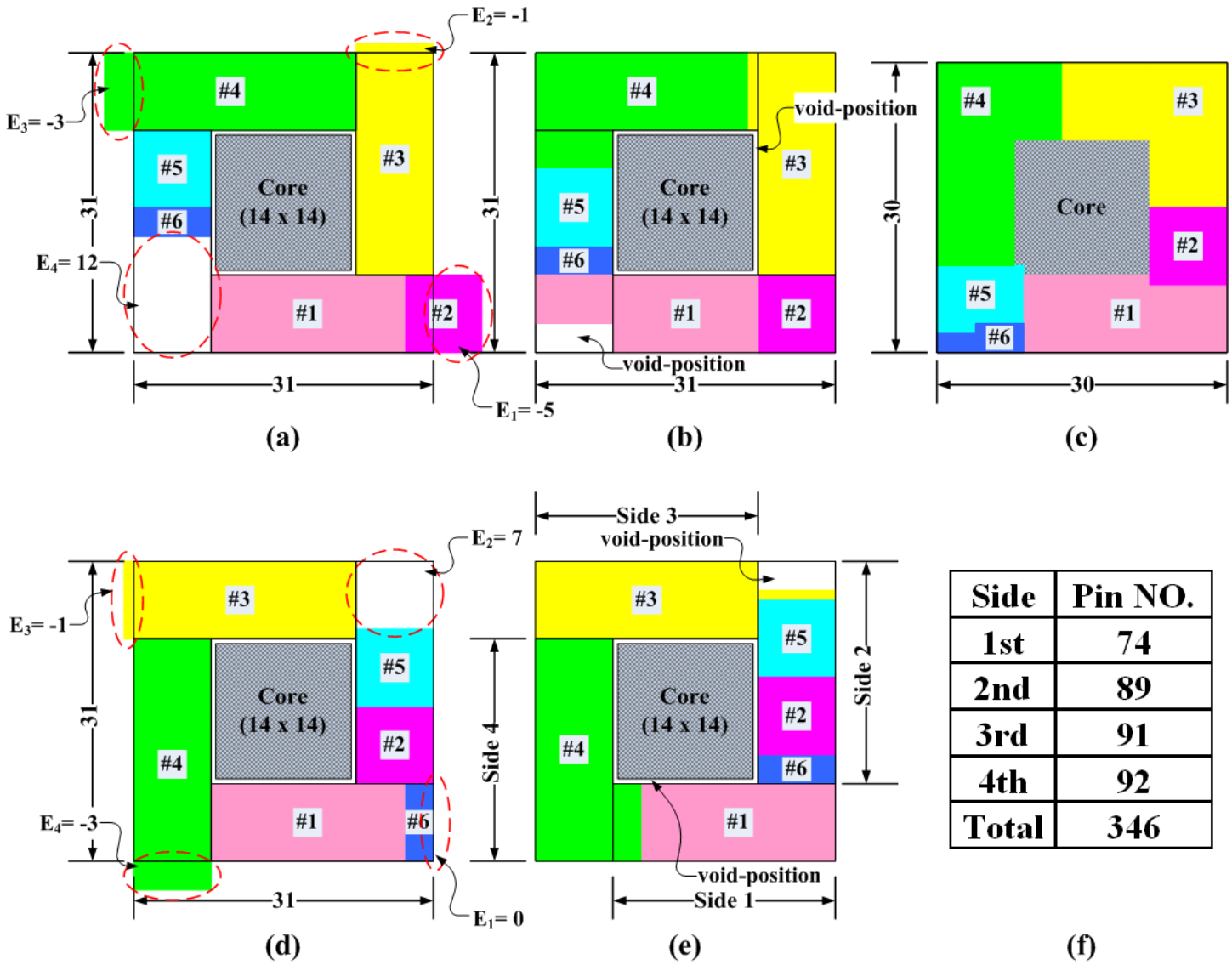


Figure 2.15: Pin-blocks grouping results of Case 2. (a) is grouping with BCPG (ϕ_1=0.5, ϕ_2=1.5), and (b) is grouping with CAPG (ψ_1=0.8, ψ_2=1.2).



Side	Pin NO.
1st	74
2nd	89
3rd	91
4th	92
Total	346

Figure 2.16: Experimental results of Case 2. (a) is placement of pin-blocks grouped with BCPG, (b) is final pin-out after (a) being floorplanned, (c) is manually designated pin-out, (d) is placement of pin-blocks grouped with CAPG, (e)is final pin-out after (d) being floorplanned and (f) is the equalized signal-pin number chart.

Table 2.4: The enlarged or shrunk column number of pin-blocks with modified types of pin pattern in Case 1 and Case 2.

	Signal bus	Previous signal-pin pattern	Enlarged/Shrunk column number of pin-blocks					
			patt. 1	patt. 2	patt. 3	patt. 4	patt. 5	patt. 6
Case 1	Bus #1	patt. 4	+9	+4	+4	N/A	N/A	-6
	Bus #2	patt. 5	+4	+2	+2	N/A	N/A	-3
	Bus #3	patt. 6	+8	+5	+5	+4	+4	N/A
	Bus #4	patt. 2	+3	N/A	N/A	-2	-2	-5
	Bus #5	patt. 2	+3	N/A	N/A	-2	-2	-6
	Bus #6	patt. 6	+12	+8	+8	+5	+5	N/A
Case 2	Bus #1	patt. 4	+9	+4	+4	N/A	N/A	-6
	Bus #2	patt. 5	+4	+2	+2	N/A	N/A	-3
	Bus #3	patt. 4	+12	+5	+5	N/A	N/A	-8
	Bus #4	patt. 4	+13	+5	+5	N/A	N/A	-9
	Bus #5	patt. 6	+9	+6	+6	+4	+4	N/A
	Bus #6	patt. 6	+4	+3	+3	+2	+2	N/A

2.5 Summary

We have proposed a novel and very efficient approach to automating pin-out designation in flip-chip BGA packaging for package-board codesign. Due to the tradeoff in signal performance and package cost, conventional approach usually takes weeks to modify package size and to rework package substrate and PCB layout, and to rearrange pin-out. These time-consuming works can be replaced by the proposed efficient methodology. By considering signal integrity, power delivery, and routability in pin-out block design, our framework provides good signal quality while achieving the minimum package size (close to manual design), which reduces package cost. Finally, the flexibility of package size migration will be preserved by a quick and simple estimation.

In the next chapter, we will introduce new range constraints and a specific representation for improving pin-out designation. Such pin-block planner will optimize system interconnects for high-speed interfaces.

Chapter 3

Package Pin-Out Planning with System Interconnects Optimization

In the previous chapter, we propose a method of automatically generating the pin-out and taking signal integrity (SI), power delivery integrity (PI) and routability (RA) into account simultaneously by pin-block design and floorplanning, thus dramatically speeding up the developing time. However, this approach ignores the considerations of shorter path length and equi-length/length matching in routing PCB trace and pin-out assignment for high-speed interface IP designs, such as USB and PCI Express. Since these features are the most important performance metrics during chip-package-board codesign, in this chapter we propose the ideas to optimize the system interconnects during package pin-out design. These ideas keep the same minimized package size as our previous and ensure that SI, PI, and RA can still be considered with significant reduction in design cost. It is achieved by relaxing the restriction of pin-block side and order on the package, usually specified by package designers. The experimental results on industrial chipset design cases show that the average improvement of our pin-block planner is over 40% when comparing the design cost with the previous work, among which we have one case accommodated over a thousand pins.

3.1 Overview

As we described in the previous chapter, the complete package-board codesign methodology should preserve the signal integrity (SI), power delivery integrity (PI), and routability (RA) of high-speed signals routing from package to PCB while optimizing the package size. In our codesign approach regarding the automation of pin-out designation (see Figure 3.1), an experienced engineer has to determine the pin configuration chart based on the location of PCB components. Next, the proposed signal-pin patterns are selected for pin-blocks construction in package design where SI, PI, and RA have been accounted for after placing pin-blocks. It also proposes a near optimal approach to minimizing package size by mathematical (linear) programming. Finally, this methodology obtains the final pin assignment by applying a rather intuitive floorplanner which bends the pin-blocks located in the excess areas and fills them into the adjacent empty areas.

However, the cost function in the previous work only considers the package size, this work exposes some weaknesses:

- The method in the previous work ignores the connections between the BGA pins and high-speed interface IP designs, which are hard macros located in chip, such as Universal Serial Bus (USB) and PCI Express interface. For the purpose of enhancing performance, the package routing for aforementioned IPs should consider shorter path length and balanced nets. Since the I/O pads in IPs are all fixed, the pin-block bent into two parts or located at the package corner will not meet these requirements. Figure 3.2(a) shows the scenario caused by a poor pin-out.

- In addition to the considerations of pin-out assignment for IPs, the pin-out planner should also regard the general requirements of equi-length or length matching for routing PCB traces. Figure 3.3(a) shows the pin-block floorplan-

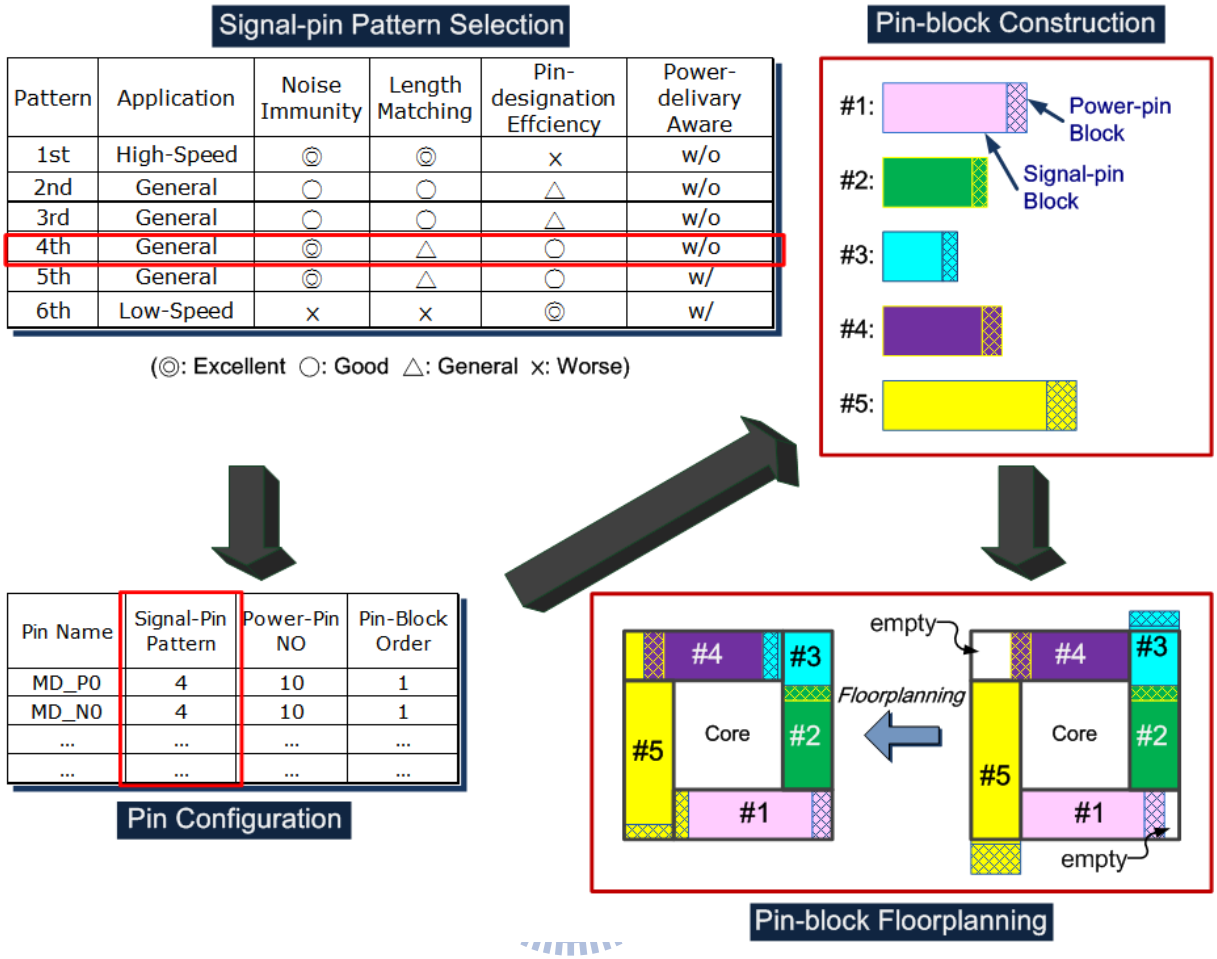


Figure 3.1: The previous work of pin-out designation. In this work, the designer selects the signal-pin patterns and determines the pin configuration chart based on design considerations. Next, pin-blocks are constructed for pin-out assignment where SI, PI, and RA have been accounted for. Finally, this methodology obtains the final pin assignment by applying an intuitive floorplanner.

ning results of previous work. When the floorplanner locates pin-blocks within the unsuitable region, it will cause longer wirelength in PCB escape routing. The longer wirelength illustrated with the darker lines in Figure 3.3 leads to greater efforts in achieving equi-length in PCB routing task. Unfortunately, designers must pre-define the placement side and order for all pin-blocks in previous approach, it then has no opportunity to change this circumstance due to these strictly specified configurations.

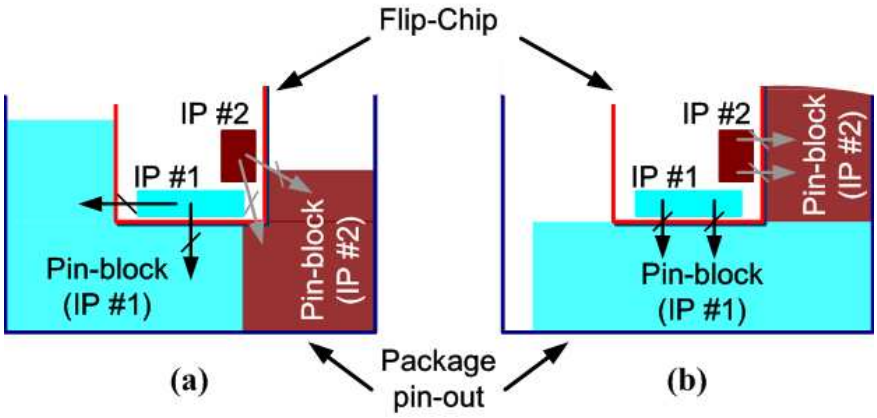


Figure 3.2: The placement of pin-blocks and IPs. (a) shows the worse pin-out assignment where the pin-block located around the package corner cannot meet the objectives of shorter path length and equi-length (length matching consideration) on package routing. (b) shows that our novel planning algorithms can overcome the drawbacks in (a).

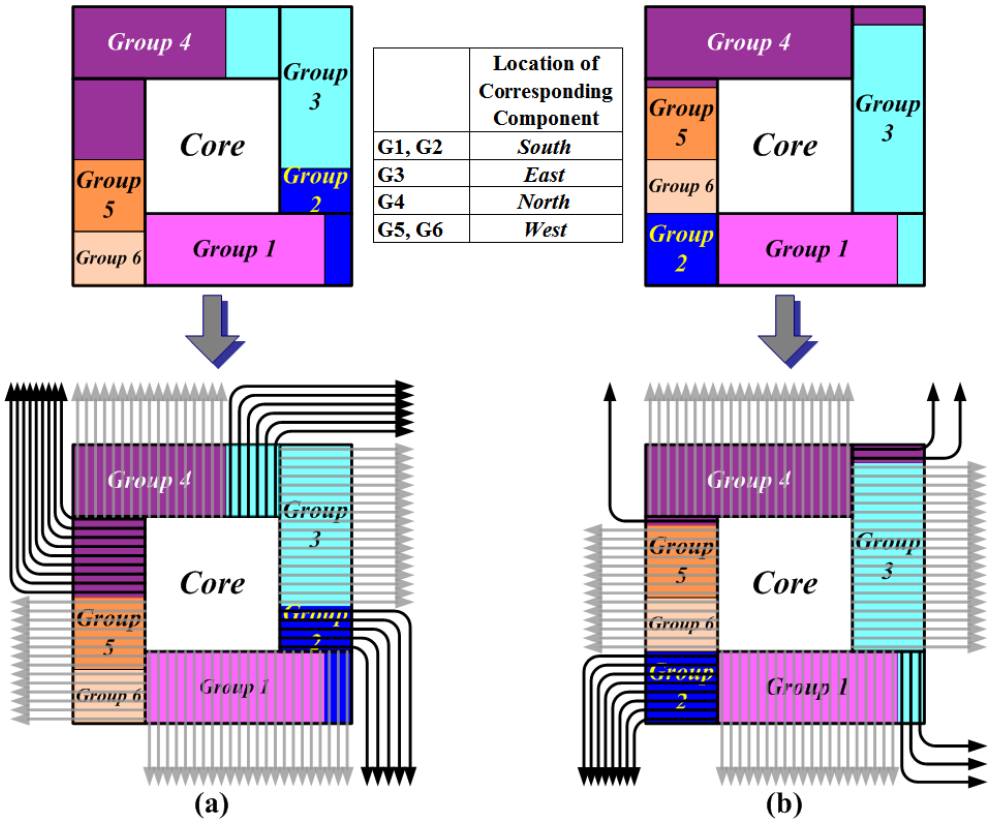


Figure 3.3: Two results of pin-block floorplanning. (a) shows the result of previous work, it causes the longer wirelength (the darker lines) in PCB escape routing due to bad pin-block allocations. (b) is the result from our ideas which provides the shorter wirelength and obtains equi-length routing for most pins.

In order to improve the tasks of package routing for high-speed IPs as well as the PCB routing, the main objectives of this chapter are to place pin-blocks near the preferred region, and to minimize the total wirelength and consider equi-length in PCB escape routing as shown in Figure 3.2(b) and Figure 3.3(b).

In this chapter, we develop an improved pin-block planner to overcome the drawbacks mentioned above. Our methodology applies simulated annealing based heuristic. By using a specially-designed representation for pin-block placement and defining range constraints, the proposed method not only optimizes the location of pin-blocks, but also minimizes the wirelength. Our ideas also work for any kind of pin-block or pin-group configurations.

3.2 Pin-Out Planning in Optimizing Package Performance and Board Wire-Planning

In the typical design flow, designers determine the pin configuration chart based on experience about the locations of PCB components and the characteristics of each signal group. The pin configuration chart defines all critical parameters including the distribution region (side), placement sequence (order), selected signal-pin pattern and the number of power pins. According the definition of this chart, the designer can finish the pin groups (or blocks) construction for all signal groups. Next, all pin-blocks will be placed along the defined side and order in which the first placed pin-block is located at the fixed location. Finally, after obtaining a rough pin-out designation and estimating the minimum package size, the pin-block floorplanning algorithm bends the pin-blocks allocated in the excess regions and shifts them into the adjacent empty regions. As a result, this shifting technique usually produces the bent pin-blocks located in the package corner without considering the package design for high-speed interface IPs. Moreover, the constraints defined in pin configuration chart restrict the margin and flexibility for optimizing the final pin-out.

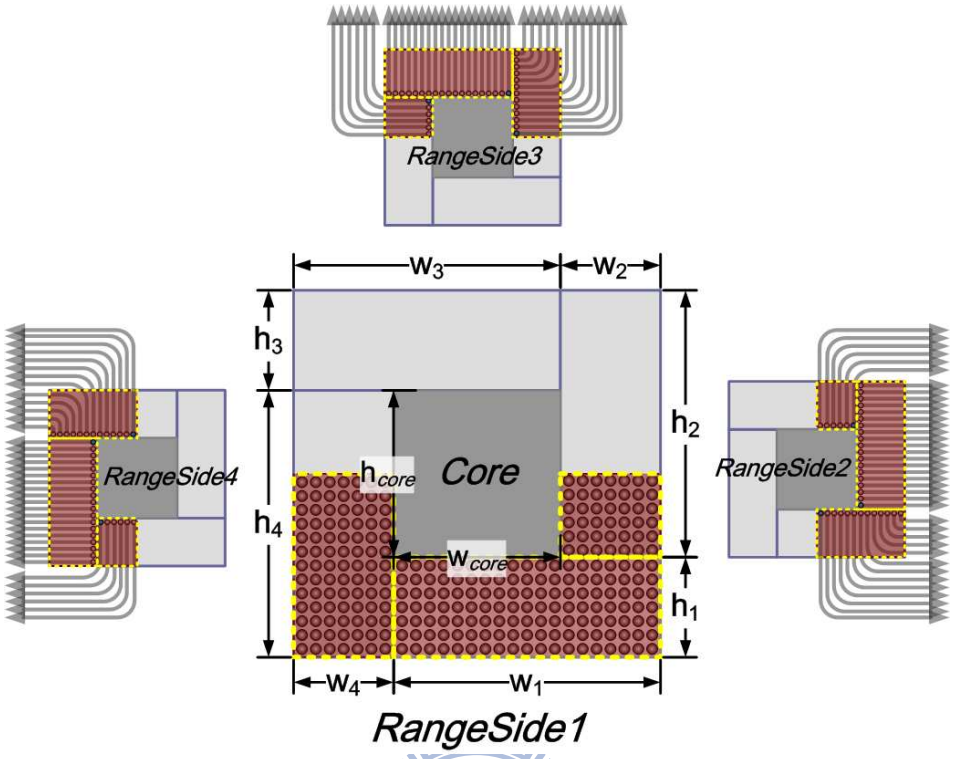


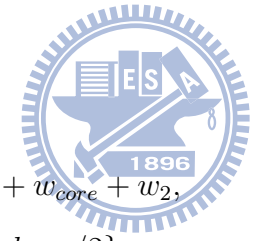
Figure 3.4: Our practical range constraints used for placing pin-blocks. Pin-blocks are restricted in *RangeSide*1, 2, 3 and 4 (each shaded region) when the corresponding components are in the south, east, north and west of PCB board respectively.

In order to loosen the restriction from designers and to obtain a better pin-block placement, we have applied the concepts of defining the pre-placed modules, boundary constraints and range constraints in the tasks of floorplanning [18, 19, 20] and placement [21, 22, 23] to redefine a new set of constraints as follows. In general, the power/ground pins used for supplying power to core logic are arranged within the core block (*Core*). While power/ground pins are at the center of package and located beneath die, the current return path will be shorter and the heat generated from die can be transferred out through these pins [16]. For these reasons, the core block will be restricted by pre-placed constraint and placed at the center of pin-out designation. This constraint is shown as follows:

- $R_{core} = \{(x_p, y_p) | w_4 + 1 \leq x_p \leq w_4 + w_{core},$
  $h_1 + 1 \leq y_p \leq h_1 + h_{core}\}$

where (x_p, y_p) is the coordinate of pin p; w_4, w_{core}, h_1, and h_{core} are the width/height shown in Figure 3.4.

According to the location of components connecting with the pin-blocks, we define a new term *RangeSide* for each pin-block instead of placement side defined by designers. Figure 3.4 shows an example where the pin-blocks are defined in *RangeSide*1 when the corresponding components are located in the south of PCB board. Therefore, all pins constrained in *RangeSide*1 must be located within the shaded region and routed toward the south to connect with components. Along the same rule, the *RangeSide*2, *RangeSide*3 and *RangeSide*4 are defined for the pin-blocks if the corresponding components are located in the east, north and west of PCB board respectively. The detailed range constraints for each side are listed as follows ($(x_p, y_p) \notin R_{core}$):



- $RangeSide1 =$

$$\{(x_p, y_p) | 1 \leq x_p \leq w_4 + w_{core} + w_2,$$
$$1 \leq y_p \leq h_1 + h_{core}/2\}$$

- $RangeSide2 =$

$$\{(x_p, y_p) | w_4 + w_{core}/2 + 1 \leq x_p \leq w_4 + w_{core} + w_2,$$
$$1 \leq y_p \leq h_1 + h_{core} + h_3\}$$

- $RangeSide3 =$

$$\{(x_p, y_p) | 1 \leq x_p \leq w_4 + w_{core} + w_2,$$
$$h_1 + h_{core}/2 + 1 \leq y_p \leq h_1 + h_{core} + h_3\}$$

- $RangeSide4 =$

$$\{(x_p, y_p) | 1 \leq x_p \leq w_4 + w_{core}/2,$$
$$1 \leq y_p \leq h_1 + h_{core} + h_3\}$$

Comparing with the placement side constraint added in previous work, the range constraints define the larger space for placing pin-blocks, thus offering the opportunities of improving pin-out designation. In addition to the optimization issue, our proposed pin-block planner also retains the feasibility of package design while satisfying all placement constraints including the pre-placed and range constraints.

3.3 Range Constrained Pin-Block Planning with System Interconnects Optimization

As described in the previous section, we will consider the core region ($Core$) as a pre-placed module which must be placed in the center of the final pin-out. Besides, pin-blocks will be treated as range-constrained modules and located within given rectangular regions such that no pin-blocks are overlapping. This section presents a pin-block planning heuristic. It applies the algorithm which is based on simulated annealing (called SA) by using a specific Cyclic Number Set (CNS) representation.

3.3.1 SA Pin-Block Planner

In this method, we use the results of previous work as the initial solution (they can be replaced by other grouping configurations). This pin-block planner eases the restriction of placement side and applies simulated annealing based heuristic with range constraints. First we introduce a special representation for pin-block planning, then we describe the floorplanning approach.

The Cyclic Number Set (CNS) Representation

The fundamental problem to floorplanning or placement lies in the representation of geometric relationship among modules [24]. Based on the consideration of the constraints and flexibility in pin-block planner, we propose a Cyclic Number Set (CNS) representation. This representation is specially designed for pin-block plan-

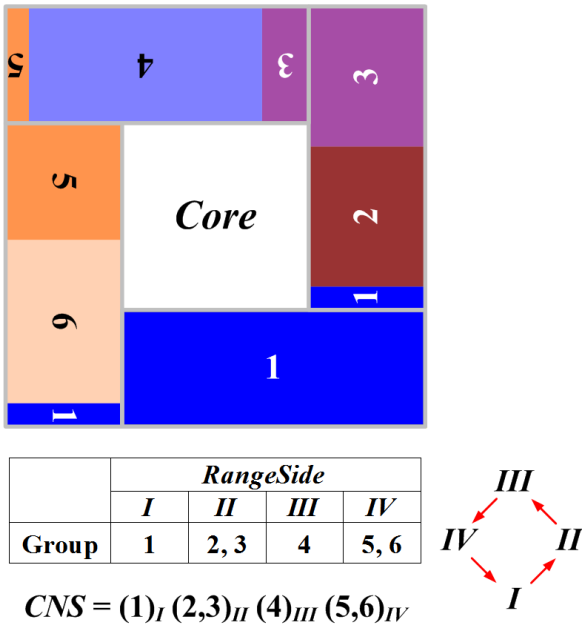


Figure 3.5: The illustration of Cyclic Number Set (CNS) representation. Each parenthesis followed by an index represents one *RangeSide*, and the number set listed within the parenthesis denotes the pin-block groups constrained in that *RangeSide*. By perturbation, the order of each number set will be changed cyclically, then the planner places the pin-blocks along the modified representation.

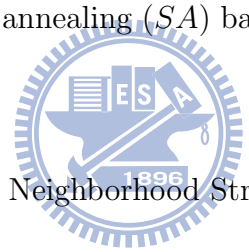
ning since it can represent the adjacent relationship between blocks and the starting point when arranging pin-out. It can also describe all variables in perturbation.

Figure 3.5 illustrates the CNS, the parentheses followed by an index represent the *RangeSide*, and those indices I, II, III and IV represent *RangeSide*1, *RangeSide*2, *RangeSide*3 and *RangeSide*4 respectively. Pin-block groups constrained in each *RangeSide* are denoted as a number set within the parenthesis. Moreover, the placement sequence of pin-blocks is determined by the order of number set. For instance, the location of pin-blocks shown in Figure 3.5 is represented as CNS=$(1)_I(2,3)_{II}(4)_{III}(5,6)_{IV}$. It presents that *RangeSide*1 is the first *RangeSide* randomly selected by the planner, and the first group to be placed in this *RangeSide* is *group*1. *RangeSide*2, the next selected *RangeSide*, contains two groups where the placement order is *group*2, *group*3. *RangeSide*2 follows the *RangeSide*1, *RangeSide*3 follows the *RangeSide*2, and so forth.

Unlike other representations in floorplanning/placement which are complicated and inapplicable for pin-block planning, the CNS representation describes the physical region and the relationship among pin-blocks. Once the CNS has been determined based on designer input, the planner can easily place the pin-blocks. Compared with the pin-block floorplanner in previous work, which used 2-D array to store the locations for all pins, our planner can simply and efficiently transform the representation to real pin-block placement.

Simulated Annealing Based CNS Floorplanning

The features of CNS presented above simplify the transformation between representation and pin-block placement. They also facilitate the optimization of pin-block planning in our simulated annealing (SA) based algorithm. The optimization process is described as follows:



- Solution Perturbation and Neighborhood Structure:

  Step 1: Randomly select one *RangeSide* from the CNS of initial (or previous) solution.

  - **Move**: Randomly choose two groups in this *RangeSide*, then exchange their sequence.

  Step 2: Randomly decide the first pin location of the updated first group then place the pin-block.

  Step 3: The rest of groups defined in the selected *RangeSide* are placed along the updated sequence determined in previous move.

  Step 4: The remainder of groups defined in the other *RangeSide* are placed according to the sequence determined in previous solution.

Step 5: Save the updated CNS representation for the new solution.

To produce a feasible solution, after randomly selecting one *RangeSide* from the CNS of previous solution, it randomly chooses two groups in the selected *RangeSide* and swaps their sequence thus modifying the CNS. The rest of steps are proceeded depending on the perturbed CNS. Figure 3.6 shows the examples of perturbation processes, (a) is the initial/previous solution and the placement of pin-blocks starts from *group*1 in *RangeSide*1. Since the *RangeSide* has been perturbed, the planner revises the CNS and the placement is reinitiated from *RangeSide*2 as shown in Figure 3.6(b). According to the move, the group orders in *RangeSide*2 are exchanged (first step). Next, the first pin location of *group*3 is randomly decided, and the planner places the pins of *group*3 and *group*2 (second and third steps). After these steps, the rest of groups must be located along the range constraints and the sequence described in the perturbed CNS (fourth step). Finally, our method saves the updated CNS of modified pin-block location for next iteration (fifth step). Figure 3.6(c) and (d) show the other two perturbation cases.

- Annealing Schedule: our SA planner uses the following schedule to minimize the cost function, then obtains an optimized pin-out.

  - $T_0 = 100$; $\alpha = 0.9$; $M = 5$; $Maxtime = 500$.

  where T_0 is the initial temperature, α is the cooling rate, M represents the time until the next parameter update, and $Maxtime$ is the total allowed time for the annealing process. After obtaining the initial solution, the perturbation procedure is iteratively invoked to perturb this given solution and get new solution until the total allowed time is exceeded.

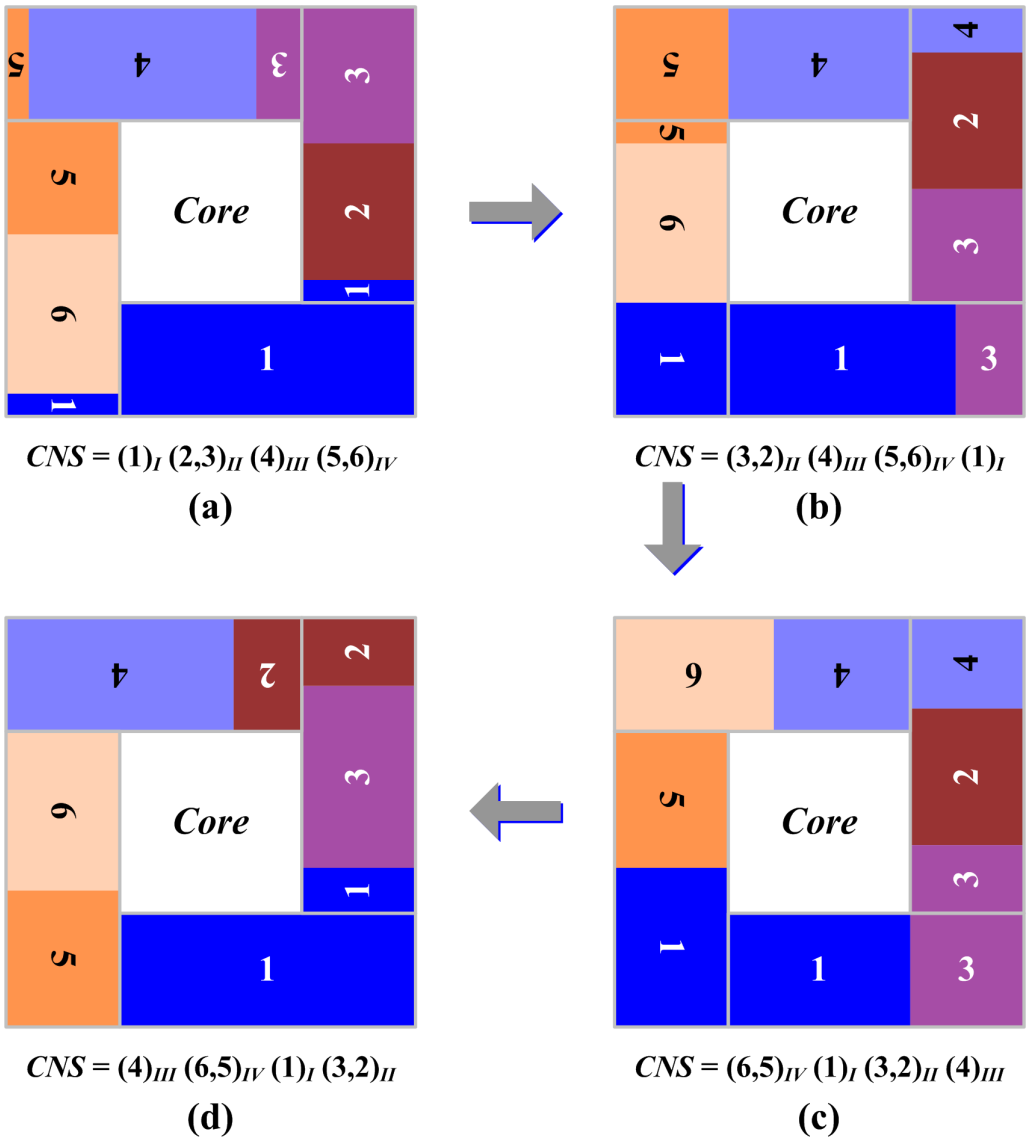


Figure 3.6: Examples of perturbation process. (a) is the initial configuration. (b) is the first perturbation case, the *RangeSide*2 has been selected and its group orders are exchanged (Step 1). The first pin location of *Group*3 is randomly decided, then the planner places all pins in *RangeSide*2 (Step 2 and 3). Following the updated CNS, the groups defined in the remainder of *RangeSide* are placed (Step 4). (c) and (d) show another two perturbation cases.

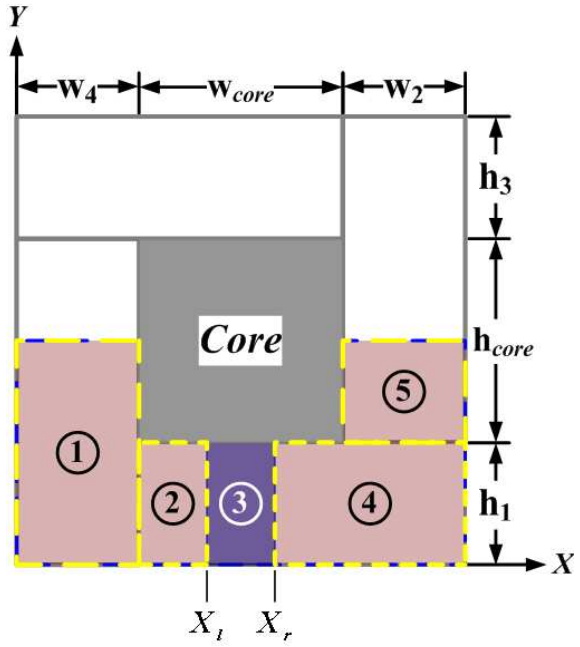


Figure 3.7: Penalty estimation for *RangeSide*1. The penalty is placement deviation induced when pin-blocks are placed away from the defined region ($X_l \leq x_p \leq X_r$).

3.3.2 Optimizing Objective Function

For the purpose of optimizing the pin-out designation, we use the penalty term, which is the deviation of desired pin-block location, as our cost function. To emphasize the location difference, its value is set to be the square of distance estimated between the pin location and the defined placement boundary. An example is shown in Figure 3.7, the designer can define a preferred boundary as a constrained region ($X_l \leq x_p \leq X_r$) for assigning pins according to the size and floorplan of corresponding IPs. Therefore, signal pins will obtain zero penalty when they are placed within the preferred region. The detailed estimation of penalty term in *RangeSide*1 is formulated as follows:

- *Region* 1:

  $Penalty = (|y_p| + |w_4 - X_l|)^2$

  when $1 \leq x_p \leq w_4$, $1 \leq y_p \leq (h_1 + h_{core}/2)$

- *Region* 2:

  $Penalty = |x_p - X_l|^2$

  when $w_4 + 1 \leq x_p \leq X_l$, $1 \leq y_p \leq h_1$

- *Region* 3:

  $Penalty = 0$

  when $X_l \leq x_p \leq X_r$, $1 \leq y_p \leq h_1$

- *Region* 4:

  $Penalty = |X_r - x_p|^2$

  when $X_r \leq x_p \leq (w_4 + w_{core} + w_2)$, $1 \leq y_p \leq h_1$

- *Region* 5:

  $Penalty = [|X_r - (w_4 + w_{core} + w_2)| + |y_p - h_1|]^2$

  when $(w_4 + w_{core} + 1) \leq x_p \leq (w_4 + w_{core} + w_2)$,

  $(h_1 + 1) \leq y_p \leq (h_1 + h_{core}/2)$

Since designers usually connect power/ground pins with power/ground planes by using the nearest vias, penalties which are added by power/ground pins located outside the constrained region will be ignored in our proposed method. By minimizing the total cost, our methodology not only decreases the signal-net length but also locates the pin-blocks near the defined boundary. Therefore, the pin-block planner can match most of the requirements of shorter path length and equi-length on package design and PCB routing.

3.4 Experimental Results

We have implemented our methodology in C++ and the platform is on Intel Pentium M 1.7GHz with 512MB memory. Five chipset cases, which act as bridges

Table 3.1: Summary of five test cases. Test Case I and Test Case II are the same cases as those in Chapter 1. Test Case III to Test Case V are modified with industrial data ("Group NO." is amount of interfaces between chipset and individual components).

	Group NO.	Signal Pin NO.	Power Pin NO.	Total Pin NO.
Test Case I	6	255	80	335
Test Case II	6	346	48	394
Test Case III	20	510	168	678
Test Case IV	25	504	216	720
Test Case V	27	770	232	1002

Table 3.2: Comparisons of penalty term (placement deviation) in previous work and *SA* pin-block planner. The results show that our approach has significant improvement in all test cases ("*Imp.*" is the improvement on the penalty term).

	Previous work	*SA*		
	Penalty	Penalty	*Imp.* (%)	Time
Test Case I	6200	2619	+57.76	< 2.0 *min*
Test Case II	8708	5802	+33.37	< 3.5 *min*
Test Case III	27048	14818	+45.25	< 6.0 *min*
Test Case IV	31590	16961	+46.31	< 6.0 *min*
Test Case V	77614	51079	+34.19	< 9.5 *min*
Avg.	–	–	+43.38	–

of all components on motherboard are used as our benchmarks (shown in Table 3.1). Test Case I and Test Case II are industrial designs which are the same cases as those in Chapter 1. Test Case III to Test Case V are derived from industrial data with some modifications. In our experiments, the penalty term (in Section 3.3.2) which is the placement deviation is considered as our cost function. For the reason of acquiring shorter path length and equi-length (length-matching consideration) on package design and PCB routing, the designer can define a preferred region then force the pin-blocks to be planned in that boundary by minimizing the penalty term. In our experiments, we set the center area of each package side as the preferred region as shown in Figure 3.7.

Experimental results are presented as the comparisons of the *SA* pin-block plan-

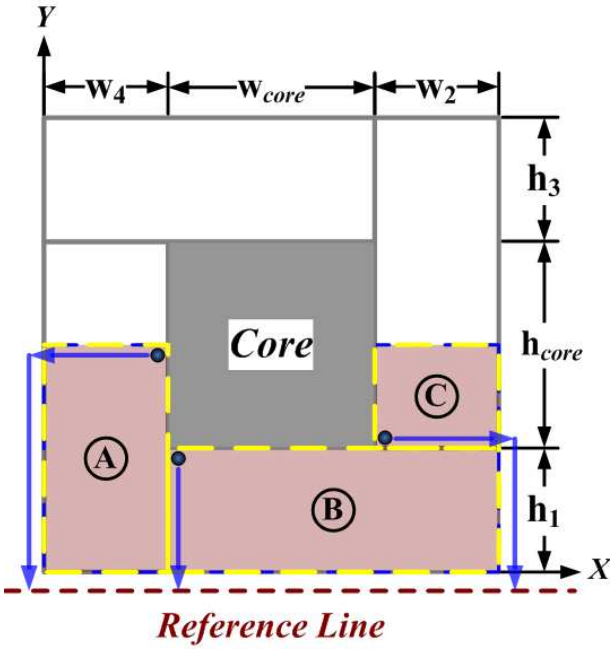


Figure 3.8: Wirelength estimation for *RangeSide*1. The wirelength is calculated in Manhattan distance from signal pin to the reference line (dotted line on the bottom).

ner and previous work. Although the *SA* planner needs more runtime, the results shown in Table 3.2 demonstrate that the *SA* planner is better than the previous work in average. Table 3.2 also shows that *SA* planner has positive improvement in penalty term when compared with that in previous work, and the runtime of designating and optimizing final pin-out for all test cases is less than ten minutes. For the design which has enormous pin-block groups, our approaches can obtain the significant improvement.

As described in the definition of *RangeSide*, signal pins located in *RangeSide*1 will route nets toward the south of PCB board then connect with the components. When our algorithm finds the minimum cost, it is to drive the pin-blocks to move to the center of *RangeSide*1 thus theoretically minimizing the signal-net length. Therefore, the optimized pin-out designation is evaluated by means of calculating the performance metric, the total wirelength. Figure 3.8 shows an example of wirelength estimation for pins located in *RangeSide*1. It is estimated in Manhattan distance

Table 3.3: Comparisons of wirelength with approaches in previous work and SA pin-block planner. The results show that our improved method has positive improvement over previous work except the Test Case I ("$Imp.$" is the improvement on the total wirelength).

	Previous work	SA	
	Wirelength	Wirelength	$Imp.$ (%)
Test Case I	1199	1216	−1.42
Test Case II	1712	1650	+3.62
Test Case III	2406	2226	+7.48
Test Case IV	2442	2173	+11.02
Test Case V	4124	3592	+12.90
Avg.	–	–	+6.72

from signal pin to the reference line (indicated in a dotted line) of each package side. The wirelength estimation for $RangeSide1$ are listed below:

- *Region A*:

  $WireLength = |x_p| + |y_p|$

  when $1 \leq x_p \leq w_4$, $1 \leq y_p \leq (h_1 + h_{core}/2)$

- *Region B*:

  $WireLength = |y_p|$

  when $(w_4 + 1) \leq x_p \leq (w_4 + w_{core} + w_2)$, $1 \leq y_p \leq h_1$

- *Region C*:

  $WireLength = |x_p - (w_4 + w_{core} + w_2 + 1)| + |y_p|$

  when $(w_4 + w_{core} + 1) \leq x_p \leq (w_4 + w_{core} + w_2)$,

  $(h_1 + 1) \leq y_p \leq (h_1 + h_{core}/2)$

According to the definition of $RangeSide$, the reference lines used for calculating the wirelength in $RangeSide2$, $RangeSide3$ and $RangeSide4$ are individually established in the east, north and west of package. The results of wirelength estimation

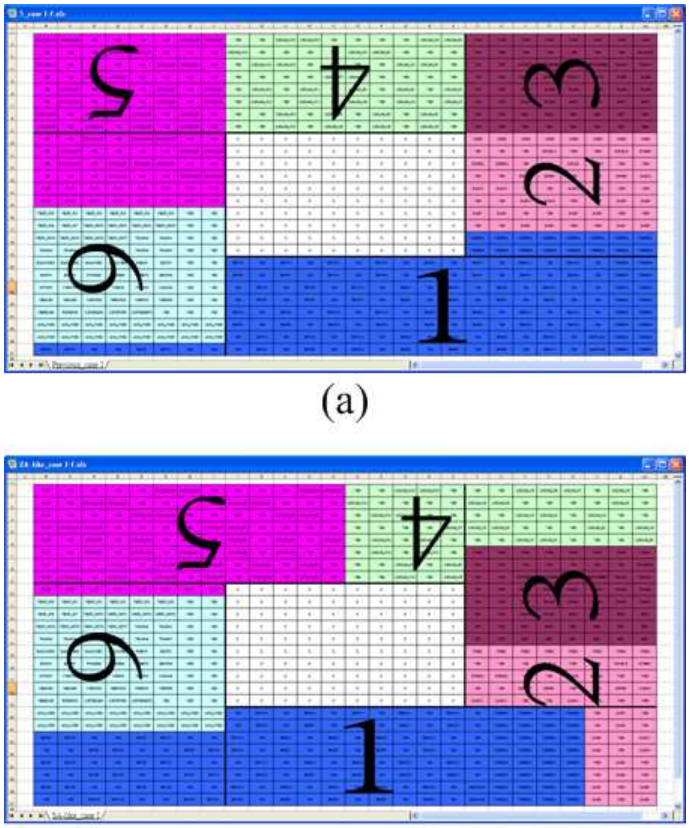
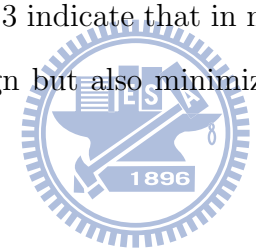


Figure 3.9: The pin-out designation results of Test Case I. (a) is generated in previous work where the *group*3, *group*5 and *group*6 are located at the corner of package. (b) is optimized by *SA* planner, each group will be moved to the center of package which is the preferred location for high performance package design.

are shown in Table 3.3. Again, in most cases the *SA* planner has positive improvements over previous work by minimizing total cost. However, there is negative improvement produced by our planner in Test Case I. Because the pin-block size and group number in each *RangeSide* are varied, in our planner all pin-blocks are located near each center of *RangeSide* to optimize the package performance for high speed IPs. In this case the wirelength is increased slightly due to the compromise between penalty of each *RangeSide*.

Figure 3.9 shows the pin-out designation of Test Case I. The pin-out planner of previous work places the most pins of *group*3, *group*5 and *group*6 at the package cor-

ner, thus causing the difficulty in achieving equi-length escape routing. By applying proposed method and defining "*Center*" as the preferred region for each *RangeSide*, Figure 3.9(b) shows the optimized results, these location-violated groups are moved to the center area of each side. As we mentioned in Section 3.1, our method will try to avoid the bent pin-block to meet the objectives of shorter path length and equi-length on package routing. However, in our experimental results some pin-blocks are still bent into two parts after minimizing total cost. That is because some interfaces possess enormous I/O pins and are grouped into large pin-blocks in our test cases. Besides, power/ground pins will not be added penalties in proposed method when they are located outside the constrained region. As a result, the bent pin-blocks are inevitable, but the proposed method will mitigate their impacts. Finally, the results shown in Table 3.2 and Table 3.3 indicate that in most cases our methodologies not only consider the package design but also minimize the wirelength in PCB escape routing.



3.5 Summary

We have proposed an improved pin-block planner with range constraints and a representation for automating pin-out designation. Based on the method of pin-block design in previous work, our approach minimizes the package size and considers SI, PI and RA as that in previous work. The experimental results show that the proposed methodologies provide significant improvement especially for large number of pin-block groups. Furthermore, we can use the range concept to restrict the pin-block location within the preferred region thus optimizing the package performance and board wire-planning.

In addition to the package-board codesign methodologies proposed in Chapter 2 and Chapter 3, we also develop a concurrent chip-package codesign flow to deal with the critical high I/O-count issues in the next chapter.

Chapter 4

Preliminary Study on Row-Based Area-I/O Planning for Chip-Package Codesign

During the early design stages, I/O and bump layout should be evaluated to optimize design cost and to avoid product failures. In this chapter, the concurrent chip-package codesign flow has been proposed for solving pressing I/O and bump planning problem. On the basis of planning area-array ICs, we firstly design the I/O-bump tile which integrates I/O and bump into a hard macro with the considerations of I/O power connection and electrostatic discharge (ESD) protection. Then we propose an I/O-row based scheme to place I/O-bump tiles with existed metal layers. By such a scheme, it not only reduces efforts at redistribution layer (RDL) routing and package design rule check (DRC), but also insures that the proposed codesign flow can be applied thus theoretically speeding up the design convergence from weeks to days. In our methodology, three intuitive attempts are proposed for package-aware I/O-bump planning. Such planning methods provide the preliminary study of performance metrics in designing the interface between chip and package, including net crossing, total wirelength and length difference/deviation. The experimental results show that our methodologies reduce the die size of I/O-pad limited ICs without sacrificing the utilization rate required for core cell placement.

4.1 Overview

Modern I/O planning is divided into two categories: peripheral I/O and area-array I/O. The peripheral I/O planning is to place I/Os along the sides of core boundary and the I/Os are connected with package balls by using wire-bonding process. Whereas, this planning method always requires larger die size to accommodate I/Os and pads, and degrades the signal integrity and power integrity for off-chip signaling due to parasitics and coupling effects [25, 26]. On the other hand, the area-array I/O planning using the flip-chip technique overcomes those drawbacks in the peripheral style. The flip-chip area-array I/O technology offers the considerable flexibility in optimizing core-I/O placement and package routing. It also has the features of smaller die size, higher I/O density, lower parasitic effects and better heat dissipation, and therefore meets the requirements of designing advanced ICs in deep-submicrometer (DSM) environment [27, 28].

4.1.1 Previous Works

Regarding the current flip-chip area-array ICs, two different area-array I/O regimes are widely utilized in industry: the extrinsic area-array I/O and intrinsic area-array I/O [29]. In [30], Maheshwari *et al*. distinguished area-array I/O as redistribution and true area-I/O. For the extrinsic area-array IC, I/Os are placed along the peripheral boundary of the core. It is similar to the peripheral I/O planning but uses a dedicated redistribution layer (RDL) to redistribute nets from peripheral I/Os to area-array bumps located in the center of core area, as shown in Figure 4.1(a). It migrates package design from wire-bonding to flip-chip technology by a re-design process, namely RDL routing task, thus gaining the advantages of smaller parasitic effects and less thermal issues. However, the die size of this distributed design will still be enlarged while the number of I/Os being increased due to the same I/O

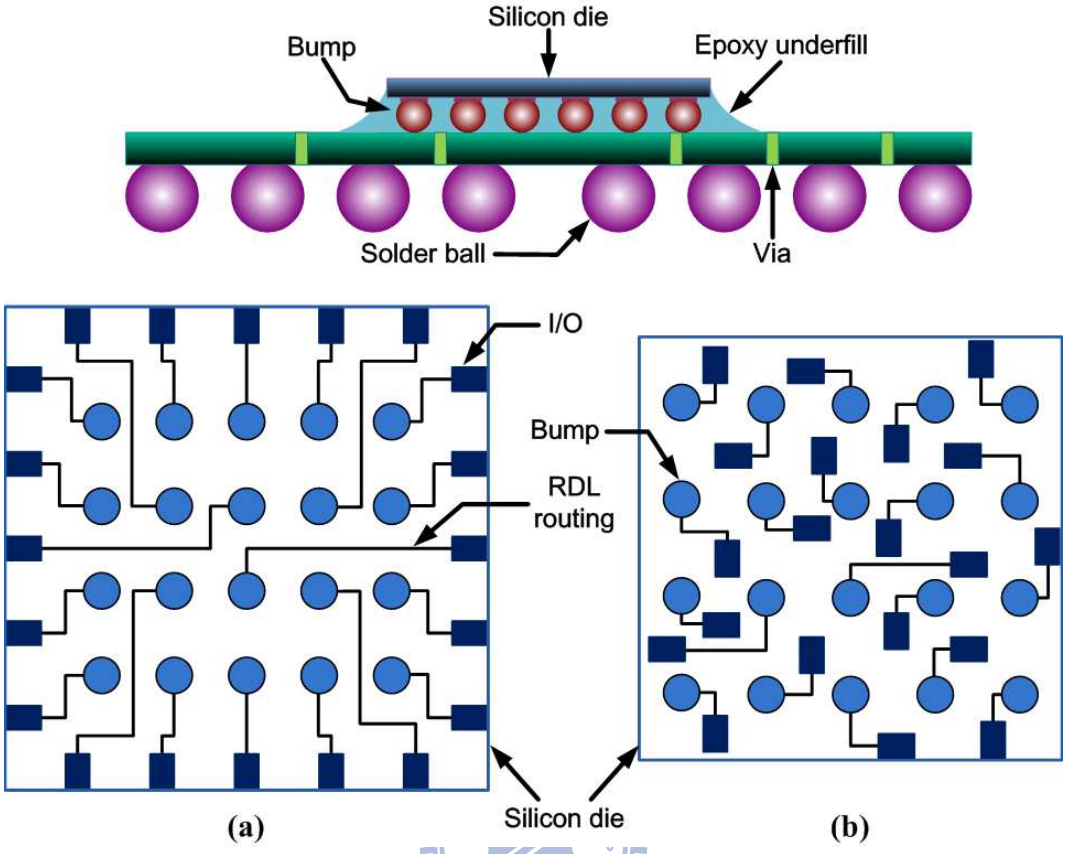


Figure 4.1: The flip-chip area-array ICs. (a) is extrinsic area-array IC which places I/Os on the core boundary and uses RDL to connect with area-array bumps. (b) is intrinsic area-array IC which freely locates I/Os on the core center thus shrinking the die size and giving the flexibility in core-I/O placement.

planning with that of peripheral I/Os. Figure 4.1(b) shows the intrinsic area-array IC, on the contrary, I/Os and bumps are freely located in the center of core region thus shrinking the die size and giving the flexibility in core-I/O placement.

Several works [7, 31, 32, 33, 34] proposed methodologies to deal with flip-chip area-array ICs. Fang *et al.* applied the network-flow-based and the integer-linear-programming-based RDL routing algorithms for designing extrinsic area-array ICs. The two-stage technique not only completes 100% routability, but also reduces the total RDL wirelength and the signal skews compared with an industrial heuristic algorithm [31, 32]. On the other hand, two recent works focused on planning and placing intrinsic area-array I/Os. [33] applied I/O clustering concept to place I/Os,

and formulated the problem as a min-cost maximum flow problem. The encouraging results indicate that the method not only achieves better timing performance but also reduces the design cost when compared with the conventional method commonly used by designer. In [7], based on integer linear programming (ILP), Xiong *et al.* formulated a constraint-driven I/O planning and placement problem, and solved it by a multi-step algorithm. The experimental results show that their algorithm can effectively deal with large scale I/O placement problem and satisfy all design constraints in real design. Another previous work proposed a network-flow based multi-RDL router for the intrinsic area-array flip-chip ICs [34]. For chip-package codesign, their router completes both chip-level routing from block ports to I/O pads and package-level RDL routing from I/O pads to bump pads, thus improving the design convergence.

All the aforementioned researches achieve some notable results. However, more considerations need to be taken to apply their methods in the hostile chip-package codesign environment:

- These approaches assume that bumps are arranged in fixed array locations with unique spacing, they then design the area-array ICs by connecting bumps and I/Os with RDL routing or planning the I/O placement with cost functions and constraints. Unfortunately, area-array I/O planning and RDL routing need a lot of efforts to coordinate with the core cell placement and routing. Moreover, the presumed uniform and fixed bump location restricts the flexibility in optimizing both chip and package designs.

- Most of previous works focus on designing area-array ICs and do not emphasize on the package ballplan given and optimized for PCB design. Without considering package ballplan, the I/Os and bumps will possibly lead to complicated package substrate routing, even failed package design [3, 35].

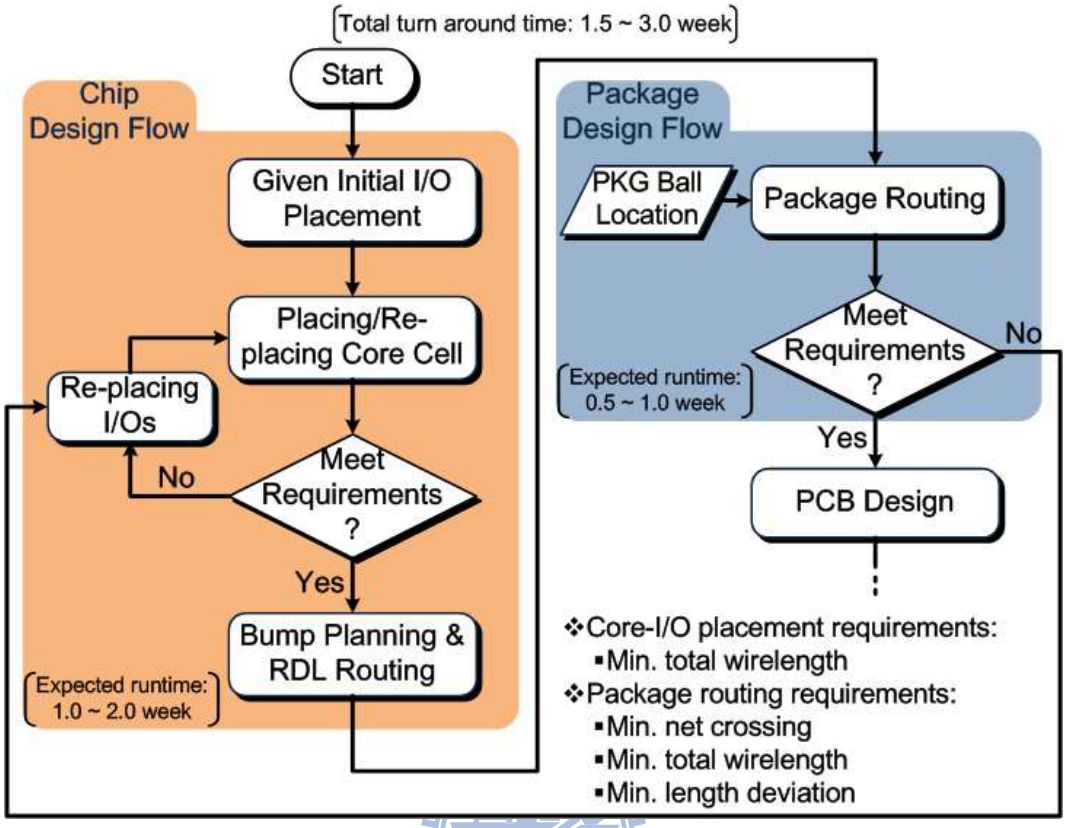


Figure 4.2: The conventional design flow. It iteratively optimizes the locations of core cells and I/Os, then performs the bump planning, RDL routing and finishes the bump placement for package routing. This sequential design flow takes long turn around time to meet all design requirements.

- The conventional design flow, which is IC-driven [4], has an inevitable shortcoming. As shown in Figure 4.2, the initial I/O placement is usually assigned by chip designer according to the core cell floorplanning results. After that, it iteratively optimizes the locations of core cells and I/Os until the final placement meets design requirement such as the minimum total wirelength. Finally, this bottom-up design flow starts to process the bump planning and RDL routing, then finishes the bump placement and provides for package routing. The major disadvantage of this sequential design flow is evident: it will probably result in long and costly re-spin cycles on satisfying entire system's design constraints.

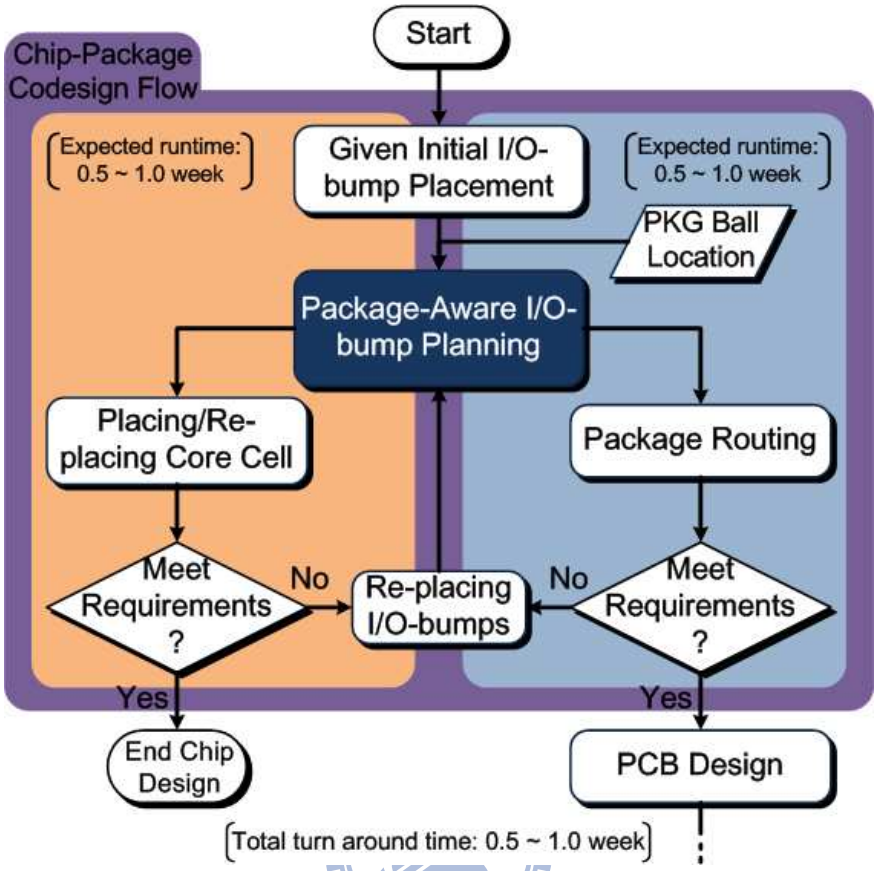


Figure 4.3: The proposed concurrent chip-package codesign flow. Through package-aware I/O-bump planning, it completes the core cell and I/O placement and package routing simultaneously, thus reducing the design cycles between chip and package.

4.1.2 Our Contributions

In order to overcome these drawbacks, here we propose a feasible area-I/O design methodology. The contributions presented in this chapter are as follows:

- We propose a concurrent codesign flow as shown in Figure 4.3. Comparing with the sequential design flow (Figure 4.2), the concurrent one completes the core cell and I/O placement and package routing in parallel, thus reducing the turn around time when designing chip and package.

- Through designing the specific I/O-bump tiles shown in Figure 4.4, complicated RDL routing efforts can be avoided. In addition, with our innovative

I/O-row based scheme shown in Figure 4.5, I/O-bump tiles can be freely placed at core area without keeping the same spacing. As a result, we significantly improve the flexibility in arranging I/Os and bumps for chip-level and package-level design.

- In this study, we propose the package-aware I/O-bump planning in our concurrent codesign flow, which consider the package ball locations and the individual objectives in chip-package codesign such as non-crossing routing, the shortest net length and the minimum length deviation among all nets.

- For I/O-pad limited design, the experimental results show that our I/O-row based scheme can effectively reduce the die size. The proposed package-aware I/O-bump planning also provides initial planning to drive concurrent codesign with design trade-offs.

4.2 Novel I/O-Bump Tile Design and I/O-Row Based Planning

In order to implement our concurrent chip-package codesign flow, one of the major constructs is to integrate the I/O and bump into a specific tile. It consequently provides all information needed in both chip-level core-I/O placement and package-level bump-ball routing, while accomplishing the I/O-bump tiles planning. As shown in Figure 4.4, each I/O-bump tile is designed as a hard macro which contains I/O driver, bump pad and power/ground trunk. All necessary interconnections are made using RDL layer usually dedicated for connecting I/Os and bumps in flip-chip design. In addition to the signal bumps, we also design the power/ground-bump tile for power supply and ground connection based on the same concept. They both include the indispensable electrostatic discharge (ESD) protection circuit commonly used in modern ICs for preventing signal and power/ground bumps from ESD damage.

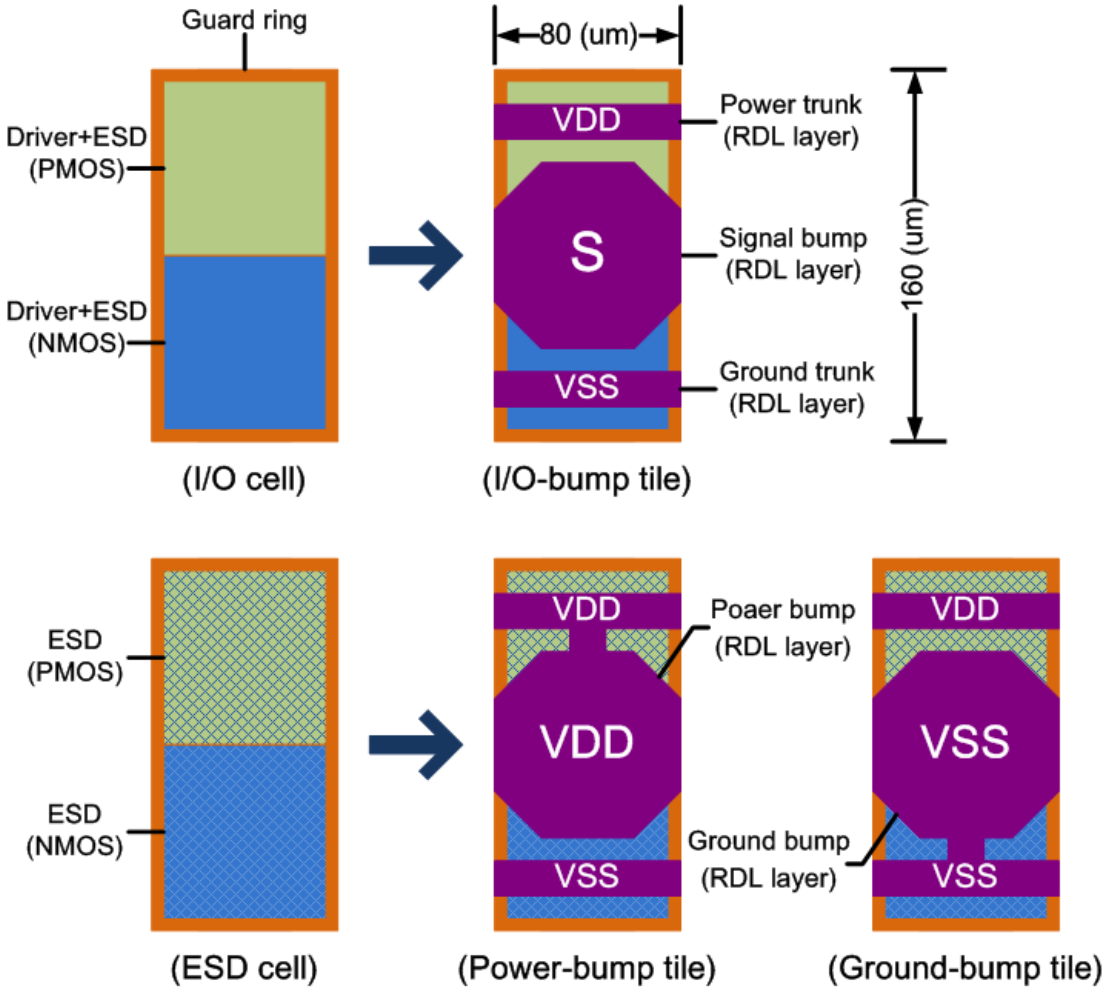


Figure 4.4: The design of proposed I/O-bump tile which integrates the I/O and bump into the single and unique interface between chip and package.

Moreover, to follow the package design rules, the bump size and pitch must be taken into consideration when planning I/O-bump tiles. In legalizing the I/O-bump tile placement, we propose an I/O-row scheme without adding extra routing layers. Figure 4.5 shows that each row is constructed with RDL layer. The width/height of tile and I/O-row are designed to follow the design rules of bump size/pitch (ex. 80 um/160 um). Once the tile is placed on the I/O-row, based on this design, only the rules within a row should be checked. It simplifies and facilitates the task for resolving the placement legalization issue. Although the I/O-row based scheme is not flexible for alternating core and I/O placement methodologies [25], this scheme

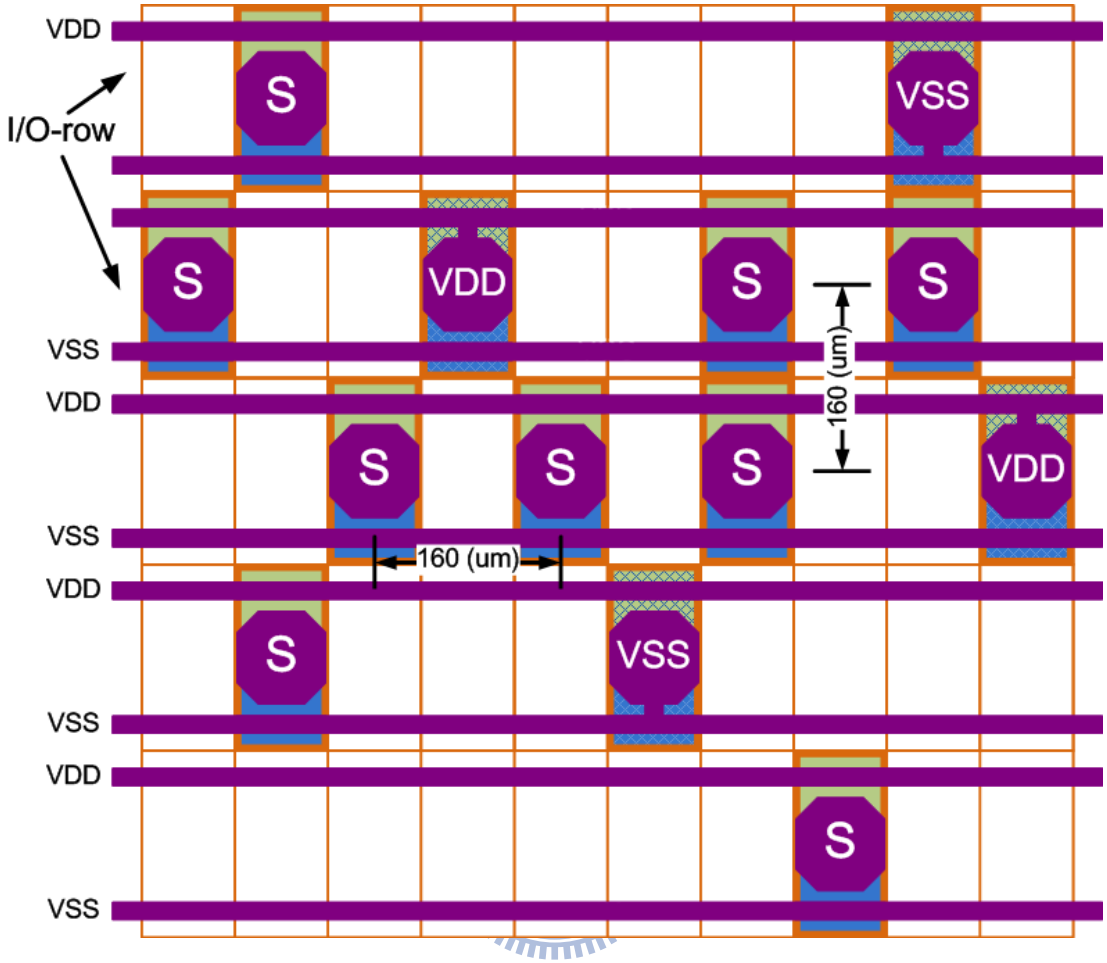


Figure 4.5: The I/O-row based I/O-bump planning scheme. It shows the width/height of tile and I/O-row are designed for satisfying the bump size/pitch. This scheme therefore simplifies the placement legalization of I/O-bump tiles.

makes the RDL routing trivial and creates the single and unique interface between chip and package by combining I/O with bump, thus actually implementing the chip-package codesign.

4.3 I/O-Bump Planning Problem in Concurrent Codesign

With I/O-bump tile design, such I/O-bump planning can provide a fairly good starting point for both chip-level and package-level design in proposed codesign flow. We define the package-aware I/O-bump planning problem as the assignment

problem which assigns the I/Os and bumps according to the distribution of package balls. Here are the detailed problem definitions:

Input:

- The given net names and locations for n package balls.
- The p I/Os and p bumps unassigned net names and locations ($p = n$).
- The design rules for chip and package.

Output:

- The assigned net names and locations for I/Os and bumps.
- The preliminary assignment provided for chip-level core-I/O placement and package-level bump-ball routing.

Assignment criteria (considering the flyline between bumps and balls):

- The minimized net crossing number.
- The minimized total wirelength.
- The minimized sum of length difference/deviation on each net.

In the next section, we show how we arrange I/Os and bumps simultaneously with the specific I/O-bump tiles invention.

4.4 Package-Aware I/O-Bump Planning Methods

To plan I/O-bump tiles along the package ball locations (given pin-out/ballplan), here we have performed two heuristic methods and applied one assignment algorithm. Each of them is distinguished by different design goals. Aiming at minimizing package routing layer to reduce package cost, the first heuristic is used to

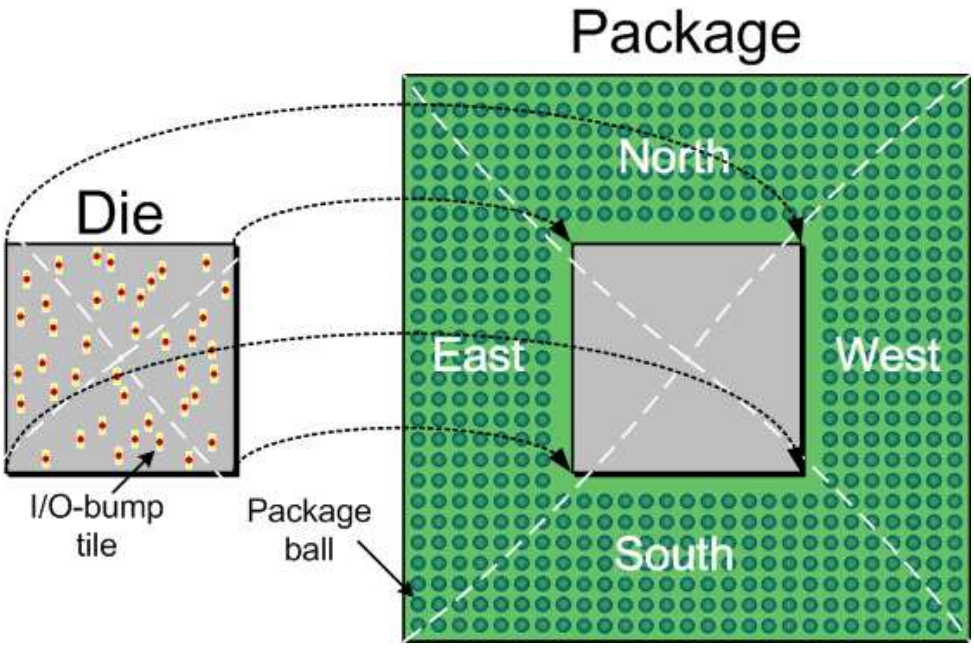


Figure 4.6: The I/O-bump tiles placement regions. The whole package will be partitioned into four sectors and the initial placement of corresponding I/O-bump tiles will be randomly generated within each sector.

obtain a zero net crossing design. To mitigate the parasitic effect on routing nets and facilitate the wirelength matching, the second heuristic focuses on shortening the net length and minimizing the wirelength deviation. As for the assignment algorithm, it balances those design requirements simultaneously. Like the works in [31] and [32], we partition the whole package into four sectors: north, west, south and east. As shown in Figure 4.6, the initial placement of corresponding I/O-bump tiles will be randomly generated in each sector. After that, each I/O-bump tile planning method mentioned above starts at the east sector. While the I/O-bump tiles are planned within this sector, the package will iteratively be rotated counterclockwise and employed methodologies until all sectors' tiles are assigned.

4.4.1 Heuristic *SORT*: Sorting I/O-Bump Tiles

In order to obtain the non-crossing (planar) routing from die bumps to package balls, we propose a heuristic method called *SORT*. Referring to the order of balls, this method sorts the I/O-bump tiles and produces their proper order, thus resulting in zero net crossing by monotonic package routing[1]. The detailed steps are as follows:

Sort package balls:

1. $Order_{ball} \leftarrow 0$
2. **Repeat:**
3. sorting ball rows ($top \Rightarrow bottom$)
4. **Repeat:**
5. sorting balls ($outer \Rightarrow inner$)
6. ordering the ball: $Order_{ball} \leftarrow Order_{ball} + 1$
7. **Until** all balls are sorted within a ball row
8. **Until** all ball rows are sorted within a package sector

Sort I/O-bump tiles:

1. $Order_{bump} \leftarrow 0$
2. **Repeat:**
3. sorting I/O-bump tiles ($top \Rightarrow bottom$, $inner \Rightarrow outer$)
4. ordering the bump: $Order_{bump} \leftarrow Order_{bump} + 1$

[1]The monotonic package routing is a routing method which routes nets from die bumps to package vias on package top layer without U-turn path. It consumes less routing resource and results in higher routing completion compared with nonmonotonic routing method [32].

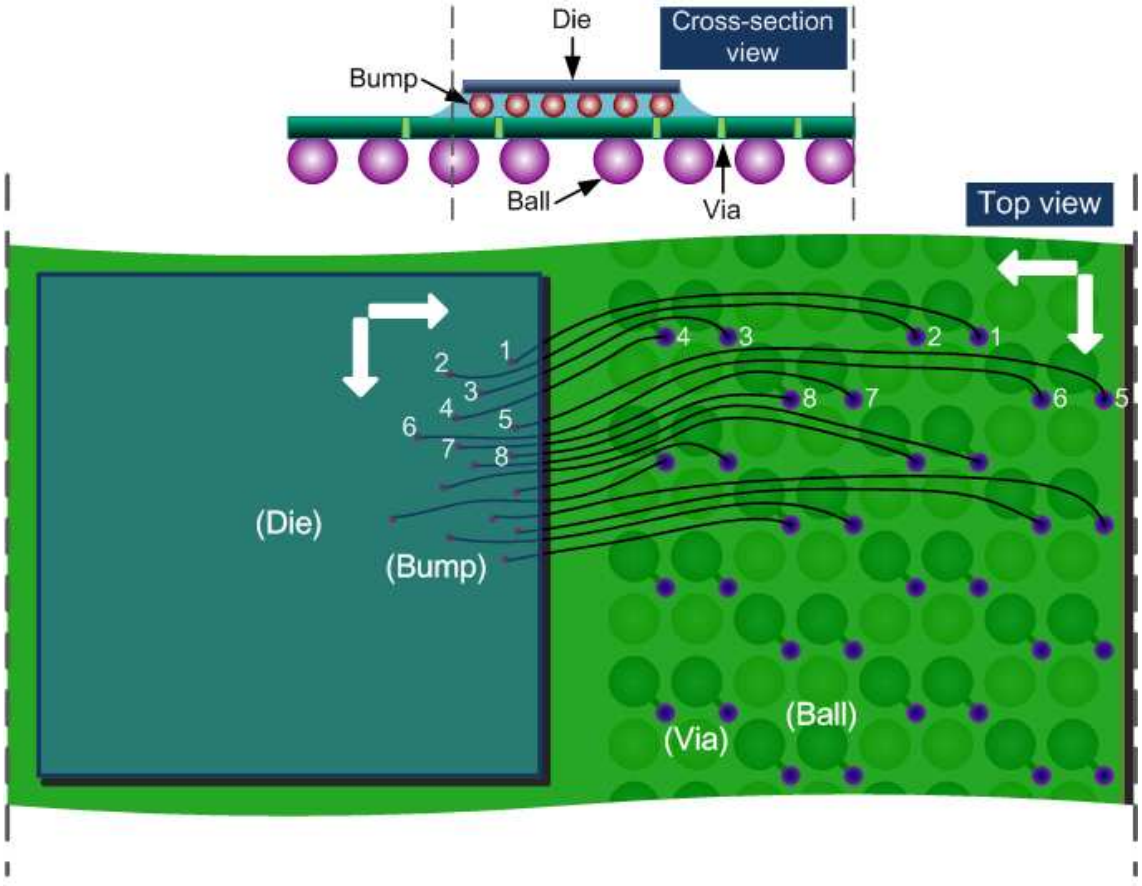


Figure 4.7: The first heuristic method: *SORT*. It sorts the I/O-bump tiles and produces the proper order by referring to the order of balls, thus resulting in zero net crossing when using the monotonic package routing. Such routing method routes nets from die bumps to package vias on package top layer without U-turn path.

5. **Until** all I/O-bump tiles are given an order within a package sector

$Order_{ball}$ and $Order_{bump}$ are used to assigned the serial number for balls and bumps, the same numbers of ball and bump are paired for connection. Figure 4.7 shows an example, the sorted order of package balls and I/O-bump tiles will lead to non-crossing package routing while applying the monotonic routing. For the tiles and balls located on other package sectors, we can also sort them as long as we rotate the package counterclockwise and implement this two-stage *SORT* heuristic.

4.4.2 Heuristic *GREEDY*: Greedily Choosing the Shortest Flyline

The *SORT* method intuitively succeeds in producing a zero-crossing package routing. However, regarding the package routing task, the net length is another critical factor influencing its performance. Since the longer wirelength induces the larger parasitic effects, nets from bumps to balls should be routed as short as possible. Besides, to achieve impedance matching, each net should be kept in the similar wirelength. For these two objectives (the shortest nets and equi-length), we propose another heuristic to simultaneously shorten the total wirelength and the length deviation called *GREEDY*. The main idea of this method is to choose the shortest flyline between bumps and balls greedily. The *GREEDY* method also consists of two stages: sorting balls and greedily find shorter flylines. The first stage of process is same as that in *SORT* method (ball sorting), and the detailed steps in the second stage are listed below:

Choose the shortest flyline greedily:

1. $Order_{bump} \leftarrow 0$
2. given the ball order in *SORT*, starting from the first ball
3. **Repeat:**
4. connecting the ball with all unchosen I/O-bump tiles
5. choosing one I/O-bump tile which can result in the shortest flyline
6. ordering the chosen bump: $Order_{bump} = Order_{ball}$; move to the next ball
7. **Until** all balls are connected within a package sector

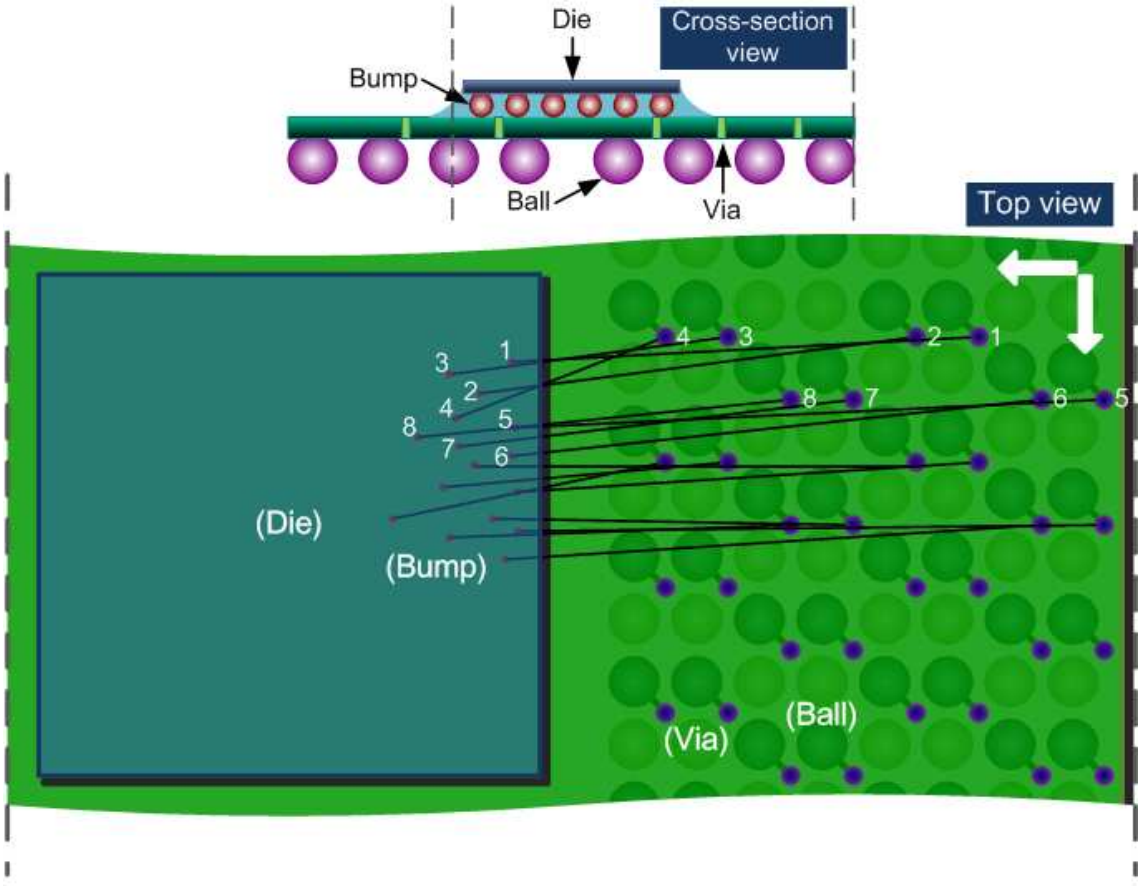


Figure 4.8: The second heuristic method: *GREEDY*. By greedily choosing the shortest flyline between bumps and balls, this method shortens the total wirelength and the length deviation.

The flyline length is calculated with the Euclidean distance between the package ball and assigned bump, and the length deviation is the difference between flyline length and the average length obtained from *SORT* method. Figure 4.8 shows the results achieved by the *GREEDY* method. It shows that each ball is greedily paired with the closed I/O-bump tile at the moment. As a result, the *GREEDY* method reduces both total wirelength and length deviation. Comparing with *SORT* method which has zero-crossing routing, the *GREEDY* method establishes the fairly different bump order. Therefore, it will inevitably cause the net crossing in package routing and increase the package design cost. The number of net crossing in *GREEDY* is calculated on swapping bump order as that in *SORT*.

4.4.3 Algorithm $WBIPT$: Weighted Bipartite Matching

For optimizing the requirements in chip-package codesign, designer must minimize the net crossing, total wirelength and length deviation at the same time. We use the results obtained from the $SORT$ method as the initial solution, and model the package-aware I/O-bump planning into a weighted bipartite matching problem as shown in Figure 4.9. The assignment problem then becomes a matching problem to match the pre-ordered ball set ($Ball_i$) and bump set ($Bump_j$) with the minimum edge weight (w_{ij}). The objective functions are as follows:

Minimize

$$\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij} \cdot x_{ij}$$

subject to

$$\sum_{i=1}^{m} x_{ij} = 1, \forall j = 1, \ldots, n \tag{4.1}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \forall i = 1, \ldots, m \tag{4.2}$$

$$x_{ij} \in \{0, 1\} \tag{4.3}$$

The element x_{ij}, a binary variable, is 1 if $Ball_i$ is assigned to $Bump_j$, otherwise x_{ij} is 0. Variables m and n are the total number of balls and bumps respectively (m=n). The edge weight w_{ij} is formulated below:

$$w_{ij} = \alpha \cdot Diff_{ij} + \beta \cdot |l_{ij} - AvgLength| \tag{4.4}$$

where $Diff_{ij}$ ($= |Order_{ball_i} - Order_{bump_j}|$) is obtained through directly subtracting the order of $Bump_j$ from that of $Ball_i$, and therefore calculating the upper bound of crossing number [36]. $AvgLength$ is the average length obtained from $SORT$ method and l_{ij} ($= \sqrt{dx^2 + dy^2}$, $dx = |x_i - x_j|$, $dy = |y_i - y_j|$) is the flyline length

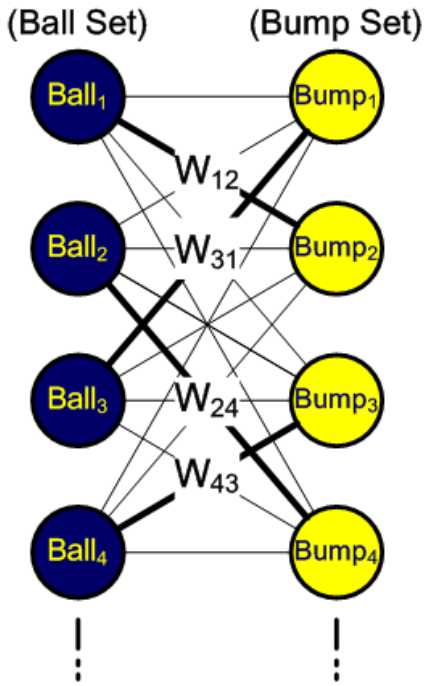


Figure 4.9: The weighted bipartite matching algorithm: $WBIPT$. This method models the package-aware I/O-bump planning into a weighted bipartite matching problem.

(mentioned in $GREEDY$ method). It can be seen that the edge weight consists of the net crossing (first term) and the wirelength deviation (second term). In addition, the user-defined parameters α and β are used to adjust the importance of net crossing and wirelength deviation. Through implementing the weighted bipartite matching ($WBIPT$) algorithm and carefully specifying these user-defined parameters, this method reassigns the order of I/O-bump tiles. Because such reassignment has more balanced net crossing and wirelength deviation, the $WBIPT$ algorithm optimizes the I/O-bump planning comparing with the previous heuristics.

Considering the given package ball locations, the I/O-bump locations planned by applying $WBIPT$ algorithm will be provided in the proposed chip-package codesign flow (Figure 4.3). In this codesign flow, the methodology of alternating the core cells and I/O-bump tiles placement/replacement and package routing will be iterated until the convincing design requirements are fulfilled.

Table 4.1: The industrial chipset designs. Results show that the die size of I/O-pad limited designs ($d2$ and $d3$ are core limited designs) can be reduced by using proposed I/O-bump tiles instead of traditional peripheral I/Os.

	Peripheral I/O				Area-Array I/O		
	Tech. (um)	Die size (um^2)	I/O size (um^2)	I/O number	Die size (um^2)	I/O-bump size (um^2)	Die size alteration
$d1$	0.18	2500^2	115×65	220	2327^2	160×80	-6.92%
$d2$	0.18	3250^2	200×60	188	3475^2	160×80	+6.93%
$d3$	0.18	2510^2	140×65	130	2742^2	160×80	+9.25%
$d4$	0.13	2580^2	120×75	200	2364^2	160×80	-8.39%
$d5$	0.13	4720^2	115×50	628	4600^2	160×80	-2.55%
$d6$	0.09	6800^2	175×65	390	6645^2	160×80	-2.29%
	(The utilization rate of core cells is kept the same)						

4.5 Experimental Results

We have implemented our methodologies in C++, and the platform is on Intel Pentium 4 3.20GHz processor with 1.5GB memory. Firstly, we use six industrial chipset designs as our test cases to demonstrate the effectiveness of I/O-row based scheme on shrinking die size (or increasing I/O number). According to the core cell floorplanning results, the peripheral I/Os originally used in those designs are replaced with our I/O-bump tiles while keeping the utilization rate of core cells the same. Without sacrificing the wiring and placement resource of core cells, Table 4.1 shows that the die size can be reduced for those I/O-pad limited designs ($d1$ and $d4$ to $d6$). On the contrary, the core limited designs ($d2$ and $d3$) will not have this advantage due to the larger I/O-bump tiles.

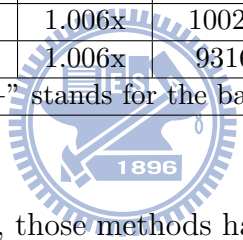
To test our I/O-bump tile planning methods, we use $d5$ as the test case and summarize all algorithms in Table 4.2. The first item $INIT$ is the given initial I/O-bump locations without considering the package balls. The methods $SORT$ and $GREEDY$ are two heuristic methods described in Section 4.4, and the last three algorithms are applying $WBIPT$ with specific user-defined parameters. As

Table 4.2: The summary of six I/O-bump planning methods.

	I/O-Bump Planning Method
♯1	$INIT$
♯2	$SORT$
♯3	$GREEDY$
♯4	$WBIPT$ ($\alpha = 1000$, $\beta = 1.0$)
♯5	$WBIPT$ ($\alpha = 500$, $\beta = 1.0$)
♯6	$WBIPT$ ($\alpha = 100$, $\beta = 1.0$)

Table 4.3: The experimental results of I/O-bump planning on test case $d5$.

	Flyline criteria					Total runtime (sec)
	Net crossing	Wirelength		Length deviation		
		Total (um)	Increase	Total (um)	Increase	
♯1	21596	6487720	1.219x	1790460	2.459x	< 1.0
♯2	0	5369640	1.009x	1297756	1.782x	< 2.0
♯3	964	5324000	–	728060	–	< 2.0
♯4	28	5358600	1.006x	1106244	1.519x	< 5.5
♯5	124	5356760	1.006x	1002125	1.376x	< 5.5
♯6	500	5353480	1.006x	931676	1.280x	< 5.5
	("–" stands for the baseline)					



we have described in Section 4.4, those methods have different characteristics, such as lower package design cost, less parasitic effect and better length matching. All these characteristics are evaluated with three terms: Net crossing, Total wirelength and Length deviation in Table 4.3.

The experimental results shown in Table 4.3 are fairly reassuring and encouraging. When ignoring the package design in planning I/Os and bumps, the $INIT$ (♯1) definitely produces the worst results comparing with all proposed package-aware I/O-bump planning algorithms. The $SORT$ (♯2) heuristic works toward obtaining the zero net-crossing in monotonic package routing through ordering I/O-bump tiles. Comparing with the other methods, the $GREEDY$ (♯3) heuristic succeeds in shortening the total wirelength and length deviation for flylines. Furthermore, the assignment algorithms $WBIPT$ (♯4 to ♯6) balance the net crossing and length deviation. Figure 4.10 is made by normalizing those performance metrics with their

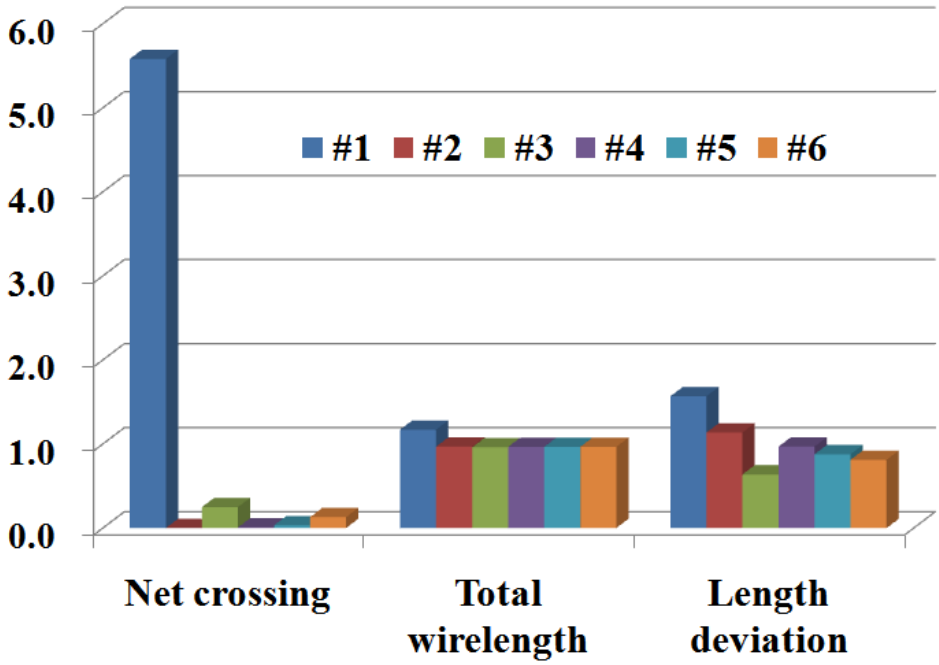


Figure 4.10: The results of normalized performance metrics. It shows the proposed methods (♯2 to ♯6) achieve the individual objectives. The assignment algorithms (♯4 to ♯6) determine the priority of net crossing and length deviation by specifying the suitable user-defined parameters (α and β).

average value. The results show that we can specify the appropriate user-defined parameters (α and β) to determine the priority of net crossing and length deviation according to the design requirements.

4.6 Summary

To develop the concurrent chip-package codesign flow, in this chapter we have proposed a novel I/O-row based scheme to place I/O-bump tiles. Two heuristics and one assignment algorithm are also provided for package-aware I/O-bump planning. The drawbacks of previous works are therefore mitigated or eliminated. Comparing with the I/O placement produced without considering the package balls, our methodologies are realizable and provide a preliminary study of chip-package codesign requirements thus helping to improve design cost and to avoid product failures.

Chapter 5

Concluding Remarks and Future Work

5.1 Concluding Remarks

To develop an automated beyond-die codesign flow, this dissertation proposes optimized methodologies for planning the critical interfaces between chip, package and board in high-end IC designs. The contributions of this dissertation are summarized as follows.

- **Automated Package Pin-out Designation for Package-Board Codesign**: In Chapter 2, we propose a novel and very efficient approach to automating pin-out designation for package-board codesign. The conventional approach which usually takes weeks to rearrange pin-out and to rework package substrate and PCB layout, can be replaced by the proposed methodology. Our 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. Finally, the flexibility of package size migration is preserved by a quick and simple estimation.

- **Package Pin-Out Planning with System Interconnects Optimization**: In Chapter 3, we propose an improved pin-block planner with new constraints and a specially-designed representation for optimizing pin-out designation. Based on the pin-block design method in Chapter 2, the proposed approach minimizes the package size and considers SI, PI and RA as that in previous work, it also provides significant improvement in shortening wirelength and accomplishing equi-length for package routing and PCB escape routing. This method therefore optimizes the package performance and board wire-planning.

- **Preliminary Study on Row-Based Area-I/O Planning for Chip-Package Codesign**: To develop the concurrent chip-package codesign flow, in Chapter 4 we present a novel I/O-row based scheme to place I/O-bump tiles. By such a scheme, it not only reduces efforts at redistribution layer (RDL) routing and package design rule check (DRC), but also achieves our concurrent chip-package co-planning/codesign flow thus shortening design turnaround times. Two heuristic methods and one assignment algorithm are provided for package-aware I/O-bump planning. Comparing with the I/O placement produced without considering the package ball locations, our methodologies preliminarily study the performance metrics in designing the interface between chip and package. That consequently facilitates the optimization works in chip-package codesign.

5.2 Future Work

We list some of possible future works as follows.

- **Pin-out Designation Considering Ordered Escape Routing**: The PCB routing problem has become increasingly difficult and no automated commercial tool can deal with high-end board routing very well [37]. As a result, routing task for high-speed PCB should be seriously confronted with. In general, such kind of routing problem can be divided into two categories, one is escape routing which routes nets from pin terminal (ball) to component boundaries, and another is area routing which routes nets between component boundaries [38]. For area routing, the planar bus routing fashion is always preferred to control and match impedance on each high-speed signal. One approach regarding automatic bus planner for PCB was published very recently in [39]. On testing a state-of-the-art industrial circuit board, their bus planner achieves 98.5% routing completion while simultaneously assigning routing layer and routing nets. However, the basic requirement of this bus planner is ordered escape routing which routing nets from balls to component boundaries with a given order. Without ordered escape routing, no way can ensure that the planar bus routing between components can be done [40]. Therefore, our future work will aim at proposing a pin-out designation methodology considering ordered escape routing. As shown in Figure 5.1, besides the signal integrity, power delivery integrity and routability we have already taken into account, we should strictly follow the given order and reduce escape routing layer at the same time while assigning pin-out.

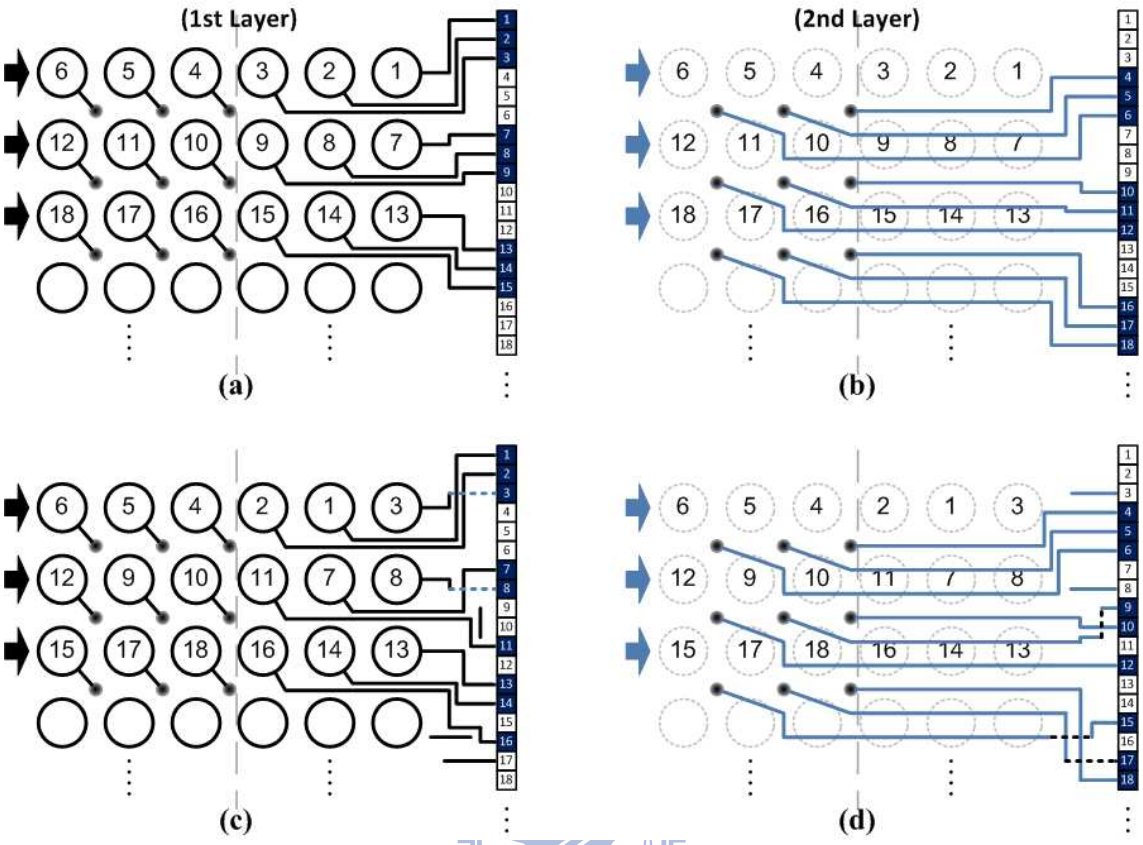


Figure 5.1: The ordered escape routing (a)(b) and disordered escape routing (c)(d). Our future work will follow the given order and reduce escape routing layer simultaneously while designating pin-out.

- **Chip-Package-Board co-simulated and co-optimized methodologies**: In designing high-speed electronic systems, more attention should be paid on interconnection between chip, package and board, it is because parasitics, interconnect attenuation and noise degrade signal quality on and off the chip thus limiting the system performance [41]. To enhance design's robustness, another future work focuses on the co-simulation and co-optimization of system interconnects after I/O-bump planning and pin-out designation. Figure 5.2 shows the expected design flow of our future work. To establish system's wide-band model for co-simulation, we firstly extract the net parasitics from package and PCB routing results, then create wide-band lossy transmission line model for interconnects [42] [43]. Since the impedance-controlled interconnects play

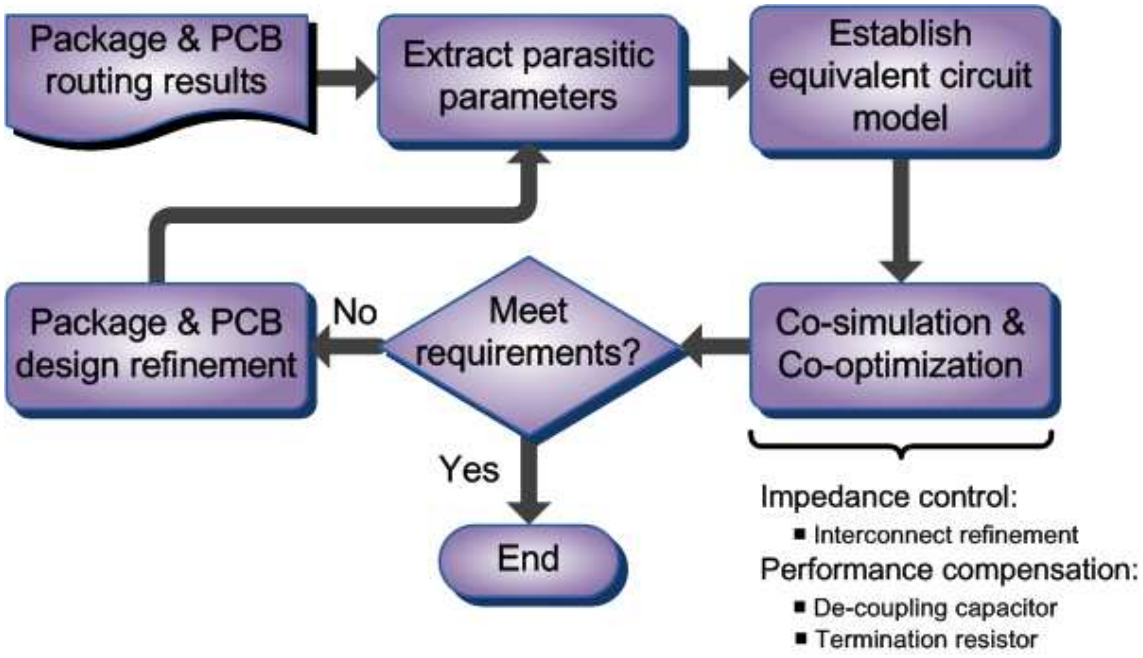


Figure 5.2: The expected design flow of our future work at chip-package-board co-simulation and co-optimization.

a critical role in high-speed off-chip communications, they should be carefully designed in order to avoid signal integrity problems such as reflection, overshoot, undershoot, and ringing [44]. Next, we want to accomplish the co-optimization tasks with impedance control and performance compensation. Such tasks will refine interconnect and add passive devices such as de-coupling capacitor and termination resistor [45].

Bibliography

[1] A. Hasan and D. Sato, "BGA Package Ball Field Interaction with Manufacturing and Design," In *Proceedings IEEE Electronic Componenets and Technology Conference*, pp. 326-333, 2004.

[2] A.H. Titus and B. Jaiswal, "A Visualization-Based Approach for Bump-Pad/IO-Ball Placement and Routing in Flip-Chip/BGA Technology," In *IEEE Transactions on Advanced Packaging*, vol. 29, no. 3, pp. 576-586, Aug. 2006.

[3] K. Sheth, E. Sarto and J. McGrath, "The Importance of Adopting a Package-Aware Chip Design Flow," In *Proceedings ACM/IEEE Design Automation Conference*, pp. 853-856, 2006.

[4] A. Fontanelli, S. Arrigoni, D. Raccagni and M. Rosin, "System-on-Chip (SoC) Requires IC and Package Co-Design and Co-Verification," In *Proceedings IEEE Custom Integrated Circuits Conference*, pp. 319-322, 2002.

[5] J. Mcgrath, "Chip/Package Co-Design: The bridge between chips and systems," In *Advanced Packaging Magazine*, Jun. 2001 [Online]. Available: http://ap.pennnet.com/display_article/103319/36/ARTCL/none/none/1/Chip/package-co-design/

[6] H.-M. Chen, I-M. Liu, D.-F. Wong, M. Shao and L.-D. Huang, "I/O Clustering in Design Cost and Performance Optimization for Flip-Chip Design," In

Proceedings IEEE International Conference on Computer Design, pp. 562-567, 2004.

[7] J. Xiong, Y.-C. Wong, E. Sarto and L. He, "Constraint Driven I/O Planning and Placement for Chip-package Co-design," In *Proceedings Asia and South Pacific Design Automation Conference*, pp. 207-212, 2006.

[8] T.-O. Chong, S.-H. Ong, T.-G. Yew, C.-Y. Chung and R. Sankman, "Low Cost Flip Chip Package Design Concepts for High Density I/O," In *Proceedings IEEE Electronic Componenets and Technology Conference*, pp. 1140-1143, 2001.

[9] M.-F. Yu and W. W.-M. Dai, "Single-Layer Fanout Routing and Routability Analysis for Ball Grid Arrays," In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 581-586, 1995.

[10] S.-S. Chen, W.-D. Tseng, J.-T. Yan and S.-J. Chen, "Printed Circuit Board Routing and Package Layout Codesign," In *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 155-158, 2002.

[11] H.N. Brady, "An Approach to Topological Pin Assignment," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 3, no. 7, pp. 250-255, Jul. 1984.

[12] S.H. Hall, G.W. Hall and J.A. McCall, *High-Speed Digital System Design*. New York: Wiley-Interscience Publication, 2000.

[13] N. Oka, C. Miyazaki, T. Uchida and S. Nitta, "Effect of a Shielding Plane Connected to Ground Plane of a PCB in EMI Reduction," In *Proceedings International Symposium on Electromagnetic Compatibility*, pp. 204-207, 1999.

[14] T. Sudo, Y. Ko, S. Sakaguchi and T. Tokumaru, "Electromagnetic Radiation and Simultaneous Switching Noise in a CMOS Device Packaging," In

Proceedings IEEE Electronic Componenets and Technology Conference, pp. 781-785, 2000.

[15] E. Diaz-Alvarez and J.P. Krusius, "Design, Simulation, Fabrication, and Characterization of Package-Level Micro-Shielding for EMI/EMC Management in BGA Environment," In *Proceedings IEEE Electronic Componenets and Technology Conference*, pp. 793-798, 2000.

[16] Altera Corp., "Designing with High-Density BGA Packages for Altera Devices," Appl. Note AN-114-4.0, Feb. 2006.

[17] R. Vanderbei, "LOQO: An interior point code for quadratic programming," Technical report, Princeton University, Princeton, NJ, 1998.

[18] F.-Y. Young and D.-F. Wong, "Slicing Floorplans with Pre-Placed Modules," In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 252-258, 1998.

[19] F.-Y. Young, D.-F. Wong and H.-H. Yang, "Slicing Floorplans with Boundary Constraints," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 9, pp. 1385-1389, Sep. 1999.

[20] F.-Y. Young, D.-F. Wong and H.-H. Yang, "Slicing Floorplans with Range Constraint," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 272-278, Feb. 2000.

[21] Y.-H. Jiang, J. Lai and T.-C. Wang, "Module Placement with Pre-Placed Modules Using the B*-Tree Representation," In *Proceedings Internationl Symposium on Circuits and Systems*, pp. 347-350, 2001.

[22] J.-M. Lin, H.-E. Yi and Y.-W. Chang, "Module Placement with Boundary Constraints Using B*-trees," In *IEE Proceedings–Circuits, Devices and Systems*, pp. 251-256, 2002.

[23] H. Murata, K. Fujiyoushi, S. Nakatake and Y. Kajitani, "Rectangle-Packing-Based Module Placement," In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 472-479, 1995.

[24] Y.-C. Chang, Y.-W. Chang, G.-M. Wu and S.-W. Wu, "B*-Trees: A New Representation for Non-Slicing Floorplans," In *Proceedings ACM/IEEE Design Automation Conference*, pp. 458-463, 2000.

[25] A. E. Caldwell, A. B. Kahng, S. Mantik and I. L. Markov, "Implications of Area-Array I/O for Row-Based Placement Methodology," In *Proceedings IEEE Symposium on IC/Package Design Integration*, pp. 93-98, 1998.

[26] J. Wang, K. K. Muchherla and J. G. Kumar, "A Clustering Based Area I/O Planning for Flip-Chip Technology," In *Proceedings International Symposium on Quality Electronic Design*, pp. 196-201, 2004.

[27] G. Pascariu, P. Cronin and D. Crowley, "Next Generation Electronics Packaging Utilizing Flip Chip Technology," In *IEEE International Electronics Manufacturing Technology Symposium*, pp. 423-426, 2003.

[28] C.-Y. Chang and H.-M. Chen, "Design Migration From Peripheral ASIC Design to Area-I/O Flip-Chip Design by Chip I/O Planning and Legalization," In *IEEE Transactions on Very Large Scale Integration Systems*, vol. 16, no. 1, pp. 108-112, Jan. 2008.

[29] C. Tan, D. Bouldin and P. Dehkordi, "Design Implementation of Intrinsic Area Array ICs," In *Proceedings Seventeenth Conference on Advanced Research in VLSI*, pp. 82-93, 1997.

[30] V. Maheshwari, J. Darnauer, J. Ramirez and W. W.-M. Dai, "Design of FPGAs with Area I/O for Field Programmable MCM," In *Proceedings ACM International Symposium on Field-programmable gate arrays*, pp. 17-23, 1995.

[31] 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 Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 8, pp. 1417-1429, Aug. 2007.

[32] J.-W. Fang, C.-H. Hsu and Y.-W. Chang, "An Integer-Linear-Programming-Based Routing Algorithm for Flip-Chip Designs," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 1, pp. 98-110, Jan. 2009.

[33] H.-M. Chen, I-M. Liu and D.-F. Wong, "I/O Clustering in Design Cost and Performance Optimization for Flip-Chip Design," In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2552-2556, Nov. 2006.

[34] J.-W. Fang and Y.-W. Chang, "Area-I/O Flip-Chip Routing for Chip-Package Co-Design," In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 518-522, 2008.

[35] J. Desai and Y. Rao, "Avoid Design Issues with Package-Aware I/O Planning," In *EE Times India*, Oct. 2008 [Online]. Available: http://www.eetindia.co.in/ART_8800546925_1800000_NT_2f421812.HTM

[36] T. Meister, J. Lienig and G. Thomke, "Novel Pin Assignment Algorithms for Components with Very High Pin Counts," In *Proceedings Design, Automation and Test in Europe*, pp. 837-842, 2008.

[37] H. Kong, T. Yan, D.-F. Wong and M. M. Ozdal, "Optimal Bus Sequencing for Escape Routing in Dense PCBs," In *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 390-395, 2007.

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

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

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

[41] M. Shen, L.-R. Zheng and H. Tenhunen, "Robustness Enhancement Through Chip-Package Co-Design for High-Speed Electronics," In *Proceedings International Symposium on Quality Electronic Design*, pp. 184-189, 2004.

[42] A. Deutsch, P. W. Coteus, G. V. Kopcsay, H. H. Smith, C. W. Surovic, B. L. Krauter, D. C. Edelstein and P. J. Restle, "On-Chip Wiring Design Challenges for Gigahertz Operation," In *Proceeding of the IEEE*, vol. 89, no. 4, pp. 529-555, Apr. 2001.

[43] C. S. Yen, Z. Fazarinc, and R. L. Wheeler, "Time-Domain Skin-Effect Model for Transient Analysis of Lossy Transmission Line," In *Proceeding of the IEEE*, vol. 70, no.7, pp. 750-757, Jul. 1982.

[44] B. Young, *Digital Signal Integrity: Modeling and Simulation with Interconnects and Packages*, Prentice Hall PTR, 2001.

[45] T. J. Gabara and S. C. Knauer, "Digitally Adjustable Resistors in CMOS for High-Performance Applications," In *IEEE Journal of Solid-State Circuits*, vol. 27, no. 8, pp. 1176-1185, Aug. 1992.

Vita

- ***Personal Information:***
  - Name: Ren-Jie Lee
  - Date of Birth: 1972/09/23
  - Sex: Male
  - Permanent Address: No.120-9, Guandong, Gongguan Township, Miaoli County 363, Taiwan (R.O.C.)
  - Permanent Phone: 886-37-224953
  - Email: rjlee@vda.ee.nctu.edu.tw; d9511836@hotmail.com
- ***Education:*** (School; Degree; Majors [Research Topics]; Period; Status)
  - Feng Chia University; Master; Electronics Engineering [RFIC design]; 1998/09/01∼2000/06/23; Graduate
  - National Chiao Tung University; Ph.D.; Electronics Engineering [EDA]; 2006/09/01∼2010/02/25; Graduate
- ***Employment History:*** (Position; Period; Company; Address)
  - Project Manager; 2000/06/26∼2006/07/31; Silicon Integrated System (SiS) Corp.; NO. 180, Sec. 2, Gongdaowu Rd., Hsinchu City 300, Taiwan (R.O.C.)
- ***Research Collaboration Experience:*** (Project; Period; Collaborator; Affiliation)
  - PCB planar router development; 2007/07/01∼ ; Prof. Yoji Kajitani; Department of Information and Media Engineering, The University of Kitakyushu, Kyushu, Japan

- Touch Pad Sensor Modeling and Analysis; 2008/06/01∼2009/10/31; R&D Director, Vincent Tao; Product Development Division, ELAN Microelectronics Corp., Hsinchu Science Park, Hsinchu City 300, Taiwan (R.O.C.)

- ***Specialties and Skills:***
  - SoC designs implementation flow
  - Package and PCB design and planning
  - C/C++ programming

- ***Research Interests:***
  - Design automation beyond die-integration
  - Chip-package-board codesign
  - System-in-package design

- ***Honors:***
  - *Outstanding Employee Award*, Silicon Integrated System (SiS) Corp., 2005.
  - *Pre-PhD Teaching Assistant Scholarship*, National Chiao Tung Univ., Fall, 2006.
  - *Outstanding Teaching Assistant Award*, National Chiao Tung Univ., Spring, 2007.
  - *Pre-PhD Teaching Assistant Scholarship*, National Chiao Tung Univ., Fall, 2007.
  - *Outstanding Teaching Assistant Award*, National Chiao Tung Univ., Spring, 2008.

Publication List

1. **R.-J. Lee** and H.-M. Chen, "Fast Flip-Chip Pin-Out Designation Respin for Package-Board Codesign," In *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 17, no. 8, pp. 1087-1098, Aug. 2009.

2. **R.-J. Lee** and H.-M. Chen, "Efficient Package Pin-Out Planning with System Interconnects Optimization for Package-Board Codesign," to appear, *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*.

3. **R.-J. Lee**, M.-F. Lai and H.-M. Chen, "Fast Flip-Chip Pin-Out Designation by Pin-Block Design and Floorplanning for Package-Board Codesign," In *Proceedings IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 804-809, 2007.

4. **R.-J. Lee**, C.-L. Weng and H.-M. Chen, "Wirelength-Driven Flip-Chip Pin-Out Designation by Range Constrained Pin-Block Planning in PCB-Package Codesign," In *19th VLSI Design/CAD Symposium*, Kenting, Taiwan, 2008.

5. **R.-J. Lee** and H.-M. Chen, "Design Planning for Chip-Package-Board Co-Design," In *12th SIGDA Ph.D. Forum at ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, 2009.

6. **R.-J. Lee** and H.-M. Chen, "Efficient Package Pin-Out Planning with Chip-Package Interconnects Optimization," In *Proceedings IEEE Electrical Design of Advanced Packaging & Systems Symposium (EDAPS)*, Hong Kong, China, 2009.

7. **R.-J. Lee** and H.-M. Chen, "Novel I/O-Bump Design and Optimization for Chip-Package Codesign," In *Proceedings IEEE Electrical Design of Advanced Packaging & Systems Symposium (EDAPS)*, Hong Kong, China, 2009.