Constraint Driven I/O Planning and Placement for Chip-package Co-design

ABSTRACT

System-on-chip and system-in-package result in the increased number of I/O cells and complicated constraints for both chip and package designs. This renders the traditional manually tuned (or chip-centered) I/O design suboptimal in terms of both turn around time and design quality. In this paper we formally introduce a set of design constraints suitable for chip-package co-design. We then formulate a constraint-driven I/O planning and placement problem, and solve it by a multi-step algorithm based on integer linear programming. Experiment results using real industry designs show that the proposed algorithm can effectively find a large scale I/O placement solution and satisfy all given design constraints in less than 10 minutes. In contrast, the state-of-the-art without considering those design constraints simply *cannot meet all design constraints by relying solely upon the conventional iterative approach.*

1. INTRODUCTION

I/O placement plays a key role as the interface between chip and package designs in a co-design flow. I/O placement not only significantly affects a chip's performance, but also determines the feasibility of a package design. Moreover, because the manufacturing cost for both chip and package is proportional to the number of routing layers used, a good I/O placement can not only help to achieve design closure, but also reduce the number of layers for both chip and package designs. However, because of the ever-increasing requirement for functionality, the number of I/O cells in a single die keeps increasing, rendering traditional manually tuned I/O placement extremely difficult.

Recently, *flip-chip* emerges as an increasingly popular alternative packaging technology for many high performance IC designs [1]. Compared to wire-bonding packaging, flip-chip technology allows shorter connection between chip and package and it permits more I/O cells to be implemented on the die. However, flip-chip packaging also brings many new design challenges for I/O placement. For example, instead of being restricted to the peripherals, I/O cells now can be placed anywhere on the die, and the placement of I/O cells also needs to consider the bump locations on the package in order to minimize the number of extra die layers for connecting I/O cells to the bumps. Therefore, the traditional timing-driven I/O placement formulation [2, 3, 4, 5] without considering package design issues is no longer applicable. A more realistic I/O placement formulation is necessary to support chip-package co-design. Moreover, I/O placement also needs to address many issues on timing closure, signal integrity (SI) and power integrity for chip-package co-design. To tackle these problems, complicated design constraints are generated in practice to guide the I/O placement. However, to the best of our knowledge, there is no study on I/O placement in the literature that has formally considered these real design constraints [6, 7, 2, 3].

The major contributions of this work include: (1) a formal definition of a set of *design constraints* suitable for chip-package co-design; (2) a new formulation of constraint-driven I/O placement problem ($CIOP$); and (3) an effective multi-step algorithm to solve $CIOP$ for chip-package co-design. To the best of our knowledge, it is the first automatic I/O planning and placement algorithm available in industry for chip-package co-design.

2. PRELIMINARY

In traditional wire-bonding designs, I/O cells are placed on chip boundaries and I/O pads on these cells are then bonded to the substrate through wires. Because of the limited boundary area, the number of I/O cells is also limited. Moreover, high inductance and high crosstalk effects due to wire bonding also limit the use of this traditional packaging technique in today's high performance IC designs.

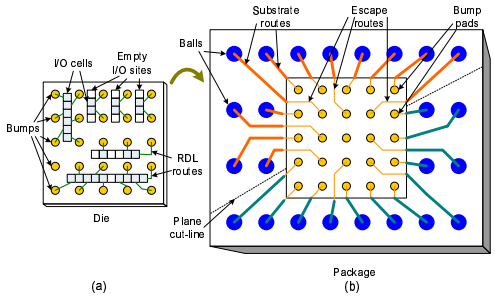


Figure 1: Area I/O Flip-chip design.

On the contrary, flip-chip technology eliminates wires for chip-package bonding. The bonding is achieved through *bumps* via the *surface mount technology* (SMT). As shown in Figure 1, I/O cells are first connected to bumps on the die via *redistribution layer* (RDL) routing, then the die is "flipped" and mounted on the surface of the substrate, where bumps are connected to *bump pads* on the substrate. Finally, package trace routing is performed to furnish the connection between bump pads to *balls* (or *package pins*). Because of the pitch mis-match problem between bumps (on the chip side) and balls (on the package side), package trace rout-

ing can be further divided into two parts. The first part is routing traces under the die, which is called *escape route* as its main goal is to *escape* traces from the die through an appropriate number of substrate layers. The second part is routing traces after escaping and we call it *substrate route*. Package trace routing is preferred to be *planar*, as it not only reduces the number of high cost *buried vias* on the package, but also makes transmission line modeling valid for timing and SI analysis of traces.

3. I/O PLACEMENT CONSTRAINTS

In a nanometer design regime, many new design issues emerge from both chip design and package design. For example, in addition to the traditional concerns on timing closure, SI and power integrity problems including crosstalk, ground bounce noise, simultaneous switching noise (SSN), and noise margin become increasingly important. To tackle these problems, complicated design constraints are generated in practice to guide the I/O placement. However, we have not found any study in literature on I/O placement that has formally considered these real design constraints. We discuss some of the common design constraints that we encountered in a number of real industrial designs in this section, and propose how to consider these design constraints for I/O placement in the next sections.

3.1 Power Integrity Constraints

Signal I/O cells' voltage specification describes the nominal voltage level as well as various voltage levels associated with the signal switching, like the allowable voltage overshoots, undershoots and ring-back values. All signal I/O cells that share the common power and ground nets fall into one *power domain*, and they are expected to be physically placed close to each other.

Moreover, in order to provide good reference planes for signal I/O traces in the package, the power/ground planes in the package have to be cut based upon I/O cells' power domain properties and their physical locations. For example, the power/ground planes in Figure 2 are cut into three parts by the *plane cut-lines*. How to define plane cut-lines heavily depends on the physical locations of I/O cells. If no attention were paid to I/O cells power domain constraints, I/O cells would scatter all over the die, which means that signal traces originating from I/O cells of the same power domain would also be scattered on the package and interleaved with signal traces from other power domains. This would either require many small zig-zag cuts on the package power/ground reference planes, or some I/O signal traces may not have the correct power/ground reference planes. In either cases, SI and power integrity are compromised. Therefore, a good I/O placement should not only consider the power domain constraints, but also should find a solution that would minimize the number of cuts on package's power/ground planes.

Power/ground (P/G) nets that provide power supply to signal I/O cells also require a set of corresponding *P/G driver cells* to be connected with the package power/ground planes. To ensure low voltage drop and minimize Ldi/dt noise, it is required that enough number of corresponding P/G driver cells be interleaved with signal I/O cells in the final placement solution. We call P/G driver cells that provide primary power supply to signal I/O cells as *primary* P/G cells. In addition to primary P/G cells, there could be other set(s) of P/G driver cells that may supply power at levels different from those of the primary P/G cells, and are required for either SI or power integrity concerns. We call these P/G cells as *secondary* P/G cells. Examples of secondary P/G cells include pre-drive power cells, reference power cells, and core power cells. In practice, it is required that a *ratio* between the number of I/O cells to the number of neighboring P/G cells (the so-called *signal-power-ground ratio*, or *SPG ratio*) be maintained such that the design can have a reliable power supply. Different SPG ratios may be derived for different groups of signal I/O cells.

3.2 Timing Constraints

It is observed that substrate routes in a package vary significantly. For example, for a typical size chip, the substrate route length can span from $1mm$ up to $21mm$. One of the impacts of such variation is on timing measured from I/O cells to package pins. Through 3D EM simulation [8], we find that ignoring package substrate route length variation per se can impact the timing by more than $70ps$ for 2.5V SSTL_2 [9] I/O cells. In another words, different substrate route lengths result in significantly different delays. If power supply variation and package stackup variation are taken into account, more significant delay variation would exhibit, making timing closure extremely difficult to attain. Therefore, for I/O cells that have critical timing relations like differential pairs, we have to take this delay variation into account when we place them. A common practice is to place differential pairs close to each other so that the corresponding package routes will have similar route length.

3.3 I/O Standard Related Constraints

It is not unusual to see that a number of common I/O interfaces are implemented in the same chip in today's high-speed IC designs (e.g., DDR2, SSTL, PCI-express, Serdes). Each I/O interface has its own specification on the relative timing requirements for signals within that interface (like differential pairs). Moreover, because all signals belonging to the same I/O interface will be very likely routed to the same I/O interface in other chips on the PCB, it is desirable to have the I/O cells belonging to the same interface physically close to each other (or even in a preferred order), which reduces the delay and SI variations between signals of the same interface. In particular, differential signal pairs are usually required to escape and to be routed together on the package. This imposes not only a *closeness* constraint but also *bump assignment feasibility* constraints (e.g., bumps escaped on the same layers).

3.4 Floorplan Induced Region Constraints

Some I/O cells may have *region* preference or constraints that are imposed by either a chip floorplan or PCB floorplan. For example, in a top-down design hierarchy, the placement of I/O signals may have side preferences imposed by a board level floorplan. Or in a bottom-up design style, the placement of core dictates that some I/O cells be placed within certain regions. Without respecting these region constraints, it could result in significant wire length increase and performance and routability degradation.

4. CIOP PROBLEM FORMULATION

We define the area where I/O cells can be placed as *I/O sites* and the area where bump pads locate as *bump arrays*.

I/O cells are connected to bump pads by RDL routing, with the direct bumping treated as a special case of RDL routing. We associate every I/O site with a set of bump pads in the bump array, so that I/O cells placed in this I/O site can connect to these bump pads via RDL routes. Therefore, a general I/O placement problem for chip package co-design consists of three essential sub-problems: (1) the placement of bump arrays, (2) the placement of I/O sites, and (3) the placement of I/O cells, with each sub-problem fulfilling different design constraints. Therefore, we propose a constraint-driven I/O placement ($CIOP$) problem as follows:

FORMULATION 1. $CIOP$ **Problem**: *Given a fixed die size, a chip net-list with I/O cells to be implemented on the die, and a set of design constraints (arising from both package and chip aspects as discussed in Section 3) for I/O placement, determine (1) the placement of bump arrays, (2) the placement of I/O sites, (3) the legal placement of I/O cells with respect to the defined I/O sites, and (4) an assignment of these I/O cells to the bumps, such that the specified design constraints are satisfied.*

The $CIOP$ problem is more difficult than the conventional purely wire length minimization core placement or pure I/O placement problems. Because in addition to finding a legal I/O placement solution, $CIOP$ also needs to satisfy many complicate design constraints. It is not likely that a one-step algorithm can solve such a problem. Therefore, we propose to solve the $CIOP$ problem via the following multi-step algorithm, which consists of two main parts: *constraint driven I/O planning* ($CPPL$) and *constraint driven detailed I/O placement* ($CDPL$).

5. CONSTRAINT-DRIVEN I/O PLANNING

The purpose of $CPPL$ is to determine the placement of bump arrays, I/O sites, and the rough locations of I/O cells, subject to the given design constraints. Figure 2 illustrates one complete output of $CPPL$.

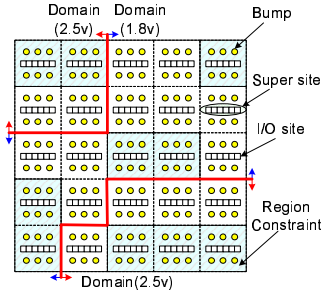


Figure 2: Output of constraint-driven I/O planning.

I/O placement and core placement are conventionally done in two separate steps. A recent study by [2] has shown that traditional global placement techniques (e.g. min-cut based or force-directed placement algorithms) can be extended to handle the co-placement of both core logic and I/O cells, and it has shown that co-placement of I/O and core could achieve better timing closure than the the conventional separated two-step approach. However, such a study is purely based upon wire length (timing-driven) minimization without considering real design constraints. Therefore, in the following we discuss the necessary changes we have made to a conventional wire-length minimization analytic placer in order to handle the design constraints as discussed in section 3. However, note that the focus of $CPPL$ is I/O planning, not the core placement.

To avoid congestion in the core area and to enable escapability and planar routability of bumps on the package, we use a bin-based density metric to measure the uniform distribution of cells for the analytical placer that spreads the core logic cells and I/O cells over the die evenly. Different from [10, 11] where only core logic cells are considered, however, we use different bin sizes for core logic cells and I/O cells during the placement procedure. Figure 3 illustrates the output of a sample run on a small benchmark we have tested, where the dark shapes are I/O cells and light shapes are core logic cells. The dashed lines represent the grid we used to distribute I/O cells.

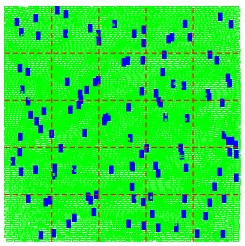


Figure 3: Global co-placement of core and I/O cells.

To consider region constraints, we model cell c_i's region constraint, also known as move bound, by a rectangle $R_i = (\overline{x_{l_i}}, \overline{y_{l_i}}, \overline{x_{h_i}}, \overline{y_{h_i}})$. The constraint is then formulated as a penalty function and added to the objective function for the placer. Cells with region constraints contribute a penalty m_i to the placement objective as given by:

$$m_i(x_i, y_i) = m(x_i, \overline{x_{l_i}}, \overline{x_{h_i}}) + m(y_i, \overline{y_{l_i}}, \overline{y_{h_i}}), \tag{1}$$

where function $m(x_i, \overline{x_{l_i}}, \overline{x_{h_i}})$ (similarly, $m(y_i, \overline{y_{l_i}}, \overline{y_{h_i}})$) is defined as:

$$m(x_i, \overline{x_{l_i}}, \overline{x_{h_i}}) = \begin{cases} (\overline{x_{l_i}} - x_i)^2, & x_i < \overline{x_{l_i}} \\ (x_i - \overline{x_{h_i}})^2, & x_i > \overline{x_{h_i}} \\ 0, & \overline{x_{l_i}} \le x_i \le \overline{x_{h_i}} \end{cases} \tag{2}$$

In other words, cells placed outside their constrained regions will increase the objective function by a quadratic term that is proportional to the distance between cells and its desired region boundaries.

It is usually very hard, if not infeasible, to model *SI and escapability constraints* exactly during placement. However, a high level approximation is still beneficial. We divide the die into grids of suitable size, and a capacity limit is specified on each grid for special types of I/O cells (e.g. differential signal pair that has to be escaped on a certain layer). Also a SPG ratio is specified for each power domain so that the grid assigned to a domain has to reserve enough power/ground I/O cells for SI purpose. Such capacity requirements are translated into density constraints to the placer.

Another important goal in this step is to place I/O cells belonging to the same power domain close to each other and away from I/O cells belonging to a different power domain, thus minimizing the number of potential power plane

cut-lines on the die[1]. To meet this goal, we add a virtual net to connect I/O cells belonging to the same domain and apply the placement algorithm to obtain an initial distribution of I/O cells on the die. We then subdivide the die into appropriately-sized bins. Each bin is assigned to at most one power domain based on the composition of the I/O cells residing within that bin. Adjacent bins assigned to the same domain are merged together. If one power domain is too fragmented (i.e., too many bins are not adjacent), the corresponding virtual net will be given a higher weight and we will re-run the placement algorithm. In order to make this "plane-cutting" process converge, I/O cells belonging to domains that are not severely fragmented may be artificially connected to some anchor points, so that they will be kept more or less in place during the placement iterations.

Having decided a rough location for each I/O cell from the global placement, we then proceed to synthesize bump arrays by combining algorithms from [12] and [13]. As I/O cells are roughly evenly distributed on the die, the bump arrays can also be evenly distributed. Moreover, we also need to reserve some extra bumps for power and ground (P/G) cells, which are needed to satisfy both the signal integrity and power integrity constraints. We then define the exact locations of I/O sites on the die where I/O cells will be finally placed as shown in Figure 2 based upon RDL routability estimation. As RDL routing is essentially a planar routing problem and there is rich literature on this subject, we refer readers to [14] and [15] for the details.

6. CONSTRAINT-DRIVEN I/O PLACEMENT

Finally, we need to assign I/O cells to the specific I/O sites such that no I/O cells overlap with each other. Since every I/O cell has already been assigned to one particular power domain after section 5, we can solve the I/O placement problem on a per domain basis, and we call it constraint-driven detailed I/O placement problem ($CDPL$).

6.1 CDPL Constraints

The design constraints discussed in Section 3 are refined with more detailed information after section 5 and can be formally described as follows. (1) One *primary SPG ratio* constraint ($N_{s,0}$:1:1). For every neighborhood consisting of at most $N_{s,0}$ number of signal I/O cells, there is at least one primary power cell and one primary ground cell. (2) A set of *secondary SPG ratio* constraints ($N_{s,i}$:1). For every $N_{s,i}$:1 constraint, it dictates that there is at least one corresponding secondary power cell for every neighborhood consisting of at most $N_{s,i}$ number of signal I/O cells. (3) A set of *region* constraints (R_i, C_i^R). A region R_i defines a rectangular area within the power domain, and a region constraint (R_i, C_i^R) specifies that the set of I/O cells C_i^R must be placed within the region. Some region constraints may come from board level or chip level floorplan; some may come from power domain definition; and some may come from $CPPL$ as we want to minimize the disturbance to the global optimal $CPPL$ solution. (4) A set of *clustering* constraints (L_i, C_i^L). L_i is a pair of length limits $(\overline{l_{i,x}}, \overline{l_{i,y}})$ and C_i^L is a cluster consisting of a set of I/O cells such that in the final placement the spreading of these cells in the x-axis (respectively y-axis) is within a range given by $\overline{l_{i,x}}$ (respectively $\overline{l_{i,y}}$). (5) A set of *differential pair* constraints (D_i=(c_{i0}, c_{i1})). Cells that form a differential pair (c_{i0}, c_{i1}) should be connected to two bumps that have similar substrate route characteristics. (6) A set of *escape layer* constraints (or bump assignment feasibility constraints) (E_i, C_i^E). E_i are the escape layer properties of bumps as determined from section 5. C_i^E are sets of I/O cells that are required to be escaped and routed on layer E_i in package. Examples of C_i^E may come from I/O cells that form certain I/O standards or may be determined from SI or power integrity analysis.

[1]It is acceptable that I/O cells forming a large power domain may be separated into more than one physically disjoint power domain on the die as shown in section 7.

6.2 CDPL Algorithm

We propose to solve the $CDPL$ problem via the following three-step algorithm. In the first step, we honor the power domain related constraints (Constraint 1 and 2). As the total number of signal cells N to be placed in the power domain is known, we can compute the total required number of primary power cells (and of ground cells) as $\lceil N/N_{s,0} \rceil$, and the total required number of secondary power cells as $\sum_i \lceil N/N_{s,i} \rceil$, respectively. We then insert the required number of primary and secondary power/ground cells into the design and distribute them evenly over the power domain.

In the second step, we formulate an ILP feasibility problem to satisfy Constraint 3, 4 and 5. A straight-forward formulation of ILP would make the problem size too large to be solved efficiently. Therefore, we reduce the ILP problem size by introducing the concept of *super site*, which is an abstraction of a cluster of physically continuous I/O sites. By properly defining the number of I/O sites in a super site (or super site's *granularity*) and formulating the ILP in terms of super site instead of I/O sites directly, we can reduce the ILP problem size greatly without sacrificing too much accuracy. The granularity of super sites is left as a tuning parameter for a particular design.

For a chosen granularity, we group all I/O sites in the power domain into a set of *super sites* $\mathcal{S} = \{S_i\}$ as shown in Figure 2. Each super site S_j is defined by its center location (a_j, b_j) on the die and a set of nearby bumps to which I/O cells assigned to this super site can connect. The number of bumps in S_j defines its bump capacity $\overline{p_j}$, which limits the total number of I/O cells that we can assign to S_j to find a feasible RDL routing solution after I/O placement. Among all bumps in S_j, the number of bumps that have similar substrate route characteristics further defines S_j's differential pair bump capacity $\overline{d_j}$. We define the binary integer variable $x_{i,j}$ for every pair of I/O cell c_i and super site S_j in the domain such that $x_{i,j}$=1 if c_i is assigned into S_j, and $x_{i,j}$=0 otherwise. Moreover, we denote the bounding box of the die by (u_L, u_B, u_T, u_R). Then we have the following ILP feasibility problem:

$$\sum x_{i,j} = 1, \quad \forall c_i \tag{3}$$

$$\sum x_{i,j} \le \overline{p_j}, \quad \forall S_j \tag{4}$$

$$\sum (x_{i0,j} + x_{i1,j}) \le \overline{d_j}, \quad \forall S_j \tag{5}$$

$$l_{i,x}^{min} \le a_j \cdot x_{k,j} + u_R \cdot (1 - x_{k,j}), \quad \forall c_k \in C_i^L, \forall L_i, \forall S_j \tag{6}$$

$$a_j \cdot x_{k,j} + u_L \cdot (1 - x_{k,j}) \le l_{i,x}^{max}, \quad \forall c_k \in C_i^L, \forall L_i, \forall S_j \tag{7}$$

$$l_{i,x}^{max} - l_{i,x}^{min} \leq \overline{l_{i,x}}, \quad \forall L_i \tag{8}$$

$$l_{i,y}^{min} \leq b_j \cdot x_{k,j} + u_T \cdot (1 - x_{k,j}), \quad \forall c_k \in C_i^L, \forall L_i, \forall S_j \tag{9}$$

$$b_j \cdot x_{k,j} + u_B \cdot (1 - x_{k,j}) \leq l_{i,y}^{max}, \quad \forall c_k \in C_i^L, \forall L_i, \forall S_j \tag{10}$$

$$l_{i,y}^{max} - l_{i,y}^{min} \leq \overline{l_{i,y}}, \quad \forall L_i \tag{11}$$

$$x_{k,j} = 0, \quad \forall c_k \in C_i^R, \forall S_j \notin R_i \tag{12}$$

$$x_{i0,j} = x_{i1,j}, \quad \forall D_i, \forall S_j \tag{13}$$

where (3) dictates that one cell can only be placed into one super site; (4) says that every super site cannot hold more I/O cells than its bump capacity; (5) specifies that every super site cannot hold more differential pairs than its differential bump capacity; (6) to (11) together enforce the clustering constraints; (12) captures the region constraints; and (13) requires us to put a differential pair into the same super site. Note that in the above ILP formulation, we do not consider wire length explicitly. Instead, we assume that there is a local region constraint for each I/O cell such that the I/O cell will be confined to a local neighborhood of its original location decided by $CPPL$. In this way the disturbance of $CDPL$ to $CPPL$ is controlled explicitly, and wire length minimization is achieved indirectly.

When the problem size of the above ILP formulation is small, we use the general branch and bound technique to solve the binary ILP problem optimally. When the above ILP problem size gets relatively large, instead of solving the ILP directly, we solve the corresponding linear programming (LP) problem by relaxing the binary variables $x_{i,j}$ to continuous ones, as $0 \leq x_{i,j} \leq 1$, and then round them back to integers afterward. Our experiment results show that such an approximation is very effective in practice and for some of the designs, we can even find the corresponding integer solutions directly.

After we assign I/O cells to super sites, in the final step of our $CDPL$ algorithm, we find a legal I/O placement within each super site with consideration of Constraint 5 and 6. We solve such a problem by formulating a min-cost-max-flow problem for each super site. We first build a bipartite network $G(V_1, V_2, E)$, where V_1 is the set of I/O cells assigned to the super site; V_2 is the set of I/O sites within the super site; E is the connection between V_1 and V_2 and are formed as follows. Because each super site contains a set of bumps and RDL routes that connect I/O sites and bumps after $CPPL$, by querying each bump's escape layer properties and substrate routing characteristics, we know whether it can be used by a given I/O cell without violating Constraint 5 and 6. Based upon such information we can build the edges of E between V_1 and V_2 in G. Moreover, we associate each edge with a cost that measures the preference of assigning one I/O cell to a particular I/O site. The cost is a weighted function of three components that we want to minimize: difference between an I/O cell's current location and its I/O site location, RDL routing length, and mismatch between I/O cells' requirements on substrate routing and bumps' package substrate routing characteristics. The final network flow problem can be obtained by adding one source and one sink into G, connecting the source to every node in V_1 with cost of zero, connecting every node in V_2 to the sink with cost of zero, and associate each edge in the network with capacity as one. It is easy to see that the assignment of I/O cells to I/O sites within each super site can be determined optimally by solving a min-cost-max-flow problem.

7. EXPERIMENT RESULTS

Four test cases derived from real industrial custom designs are used to illustrate the effectiveness of our $CIOP$ problem formulation and solution. Table 1 shows the characteristics of the test cases.

Design	# Signal I/O	# Power Domain	# Constraints
d1	1221	4	1801
d2	504	6	814
d3	450	4	934
d4	641	25	1433

Table 1: Test case characteristics.

We report the experiment results in Table 2 including the number of bumps, the number of physically disjointed power domains, the number of inserted P/G cells (both primary and secondary), and constraint satisfaction ratio (CSR), which is defined as the ratio between the number of satisfied constraints and the total number of given constraints[2]. According to Table 2, we see that our $CIOP$ algorithm can effectively find an I/O cell placement solution and satisfy all design constraints simultaneously in the first run (first-time right).

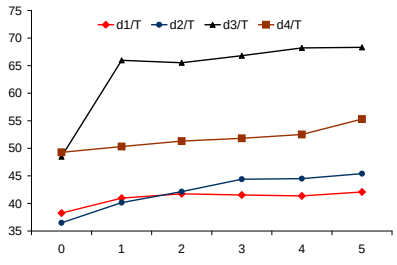
Design	Bumps	Domains	P/G Cells	CSR	Runtime(s)
d1	1560	6	328	100%	538
d2	963	12	445	100%	177
d3	906	6	453	100%	132
d4	1187	71	459	100%	81

Table 2: Experiment results for $CIOP$.

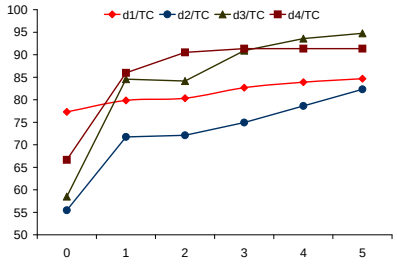
In the following, we compare our results with the conventional two-step approach followed by iterative improvement. In the first experiment, we perform the conventional co-placement of core and I/O cells followed by detailed I/O placement targeting at wire length minimization [2], but without considering the above design constraints. We denoted it as $TIOP$. After that, we measure the quality of design in terms of CSR. Obviously, as design constraints are totally ignored, there may be many design constraint violations. Therefore, we perform a local refinement procedure to remedy this problem by iteratively swapping, shifting or relocating I/O cells to improve CSR. The local refinement's search region is increasingly expanded heuristically after each iteration. As shown in Figure 4(a), before local refinement procedure (zeroth iteration), the CSR is very low (no more than 50%). Local refinement procedure does improve the design, but it is only effective for the first iteration, after that the improvement is very slim. For all designs, the CSR after five iterations of local refinement range from 40% to 70% and start to plateau. This shows that I/O placement without considering design constraints is *impossible to meet all design constraints by relying solely upon the conventional iterative approach.*

In the second experiment, we perform our $CPPL$ followed by conventional detailed I/O placement. We denote it as $TCIOP$. As shown in Figure 4(b), $CPPL$ indeed helps to

[2]Region constraints generated *internally* after $CPPL$ are not included.



(a) *TIOP*



(b) *TCIOP*

Figure 4: Comparison of *CSR*. The *x*-axis is the iteration number and the *y*-axis is *CSR* in percentage, with *CIOP*'s *CSR* being 100%.

obtain better design quality. For the zeroth iteration, the *CSR* is more than 55% and can be up to 77% compared to *TIOP*. After five iterations of local refinement, more than 80% constraints are satisfied, but sill not 100% compared to *CIOP*. Therefore, we conclude that *I/O planning and placement should be constraint driven to be effective in achieving first-time right design closure.*

We further report the wire length as another metric to measure quality of design. We normalize the wire length with respect to the conventional wire length minimization approach (i.e., the zeroth iteration in Figure 4(a). The percentage of wire length increase is shown in Figure 5. It shows that our *CIOP* incurs the largest wire length increases among all designs for the first iteration. However, considering the fact that our *CIOP* satisfies all design constraints while others do not, we believe such a small amount of wire length increase (no more than 8%) is reasonable.

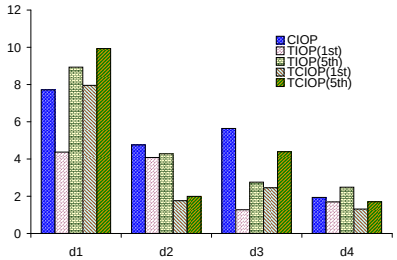


Figure 5: Wire length increase for *CIOP*, *TIOP* and *TCIOP*.

We report the runtime in second for *CIOP* in Table 2. The machine has a 3.0GHz P4 CPU with 2G memory running Linux. It is observed that the total runtime is less than 10 minutes for the largest design. Therefore, our *CIOP* algorithm can provide a quick turn-around time for chip-package co-design by enabling the designer to explore more design alternatives (through different specification of the constraints) in a reasonable amount of time during the co-design process.

8. CONCLUSION AND DISCUSSION

In this paper, we have formally defined a set of common design constraints for chip-package co-design. Based upon these real design constraints, a detailed constraint-driven I/O placement problem (*CIOP*) has been formulated and solved effectively via a multi-step algorithms. Experiment results based on real industry designs have shown that the proposed algorithm can effectively find a large scale I/O placement solution while satisfying all design constraints in less than 10 minutes.

9. REFERENCES

[1] G. Pascariu, P. Cronin, and D. Crowley, "Next generation electronics packaging utilizing flip chip technology," in *Electronics Manufacturing Technology Symposium, IEEE/CPMT/SEMI 28th International*, pp. 423–426, July 2003.

[2] A. Caldwell, A. Kahng, S. Mantik, and I. Markov, "Implications of area-array i/o for row-based placement methodology," in *IC/Package Design Integration, IEEE Symposium on*, pp. 93 – 98, Feb. 1998.

[3] J. Wang, K. Muchherla, and J. Kumar, "A clustering based area I/O planning for flip-chip technology," in *Quality Electronic Design, 5th International Symposium on*, pp. 196 – 201, 2004.

[4] R. Farbarik, X. Liu, M. Rossman, P. Parakh, T. Basso, and R. Brown, "CAD tools for area-distributed I/O pad packaging," in *Multi-Chip Module Conference, IEEE*, pp. 125 – 129, Feb. 1997.

[5] P. Dehkordi, C. Tan, and D. Bouldin, "Intrinsic area array ics: what, why, and how," in *Multi-Chip Module Conference, IEEE*, pp. 120 – 124, Feb 1997.

[6] M. Pedram, K. Chaudhary, and E. Kuh, "I/o pad assignment based on the circuit structure," in *Computer Design: VLSI in Computers and Processors, IEEE International Conference on*, pp. 314 – 318, Oct. 1991.

[7] B. Chen and M. Marek-Sadowska, "Timing driven placement of pads and latches," in *ASIC Conference and Exhibit,Fifth Annual IEEE International*, pp. 30–33, Sept. 1992.

[8] "SpeedXP user manual," in *http://www.sigrity.com/*.

[9] E. I. A. J. S. S. T. Division, "Stub series terminated logic for 2.5 volts (SSTL_2)," in *EIA/JEDEC Standard*, Sept. 1998.

[10] N. Viswanathan and C. C.-N. Chu, "Fastplace: efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model," in *ISPD '04: Proceedings of the 2004 international symposium on Physical design*, pp. 26–33, 2004.

[11] K. Vorwerk, A. Kennings, and A. Vannelli, "Engineering details of a stable force-directed placer," in *ICCAD'04: Proceedings of the 2004 international conference on computer aided design*, pp. 573–580, 2004.

[12] C. Tan, D. Bouldin, and P. Dehkordi, "Design implementation of intrinsic area array ics," in *Advanced Research in VLSI, Seventeenth Conference on*, pp. 82 – 93, Sept. 1997.

[13] M. M. Ozdal and M. D. Wong, "Simultaneous escape routing and layer assignment for dense PCBs," in *Proc. Int. Conf. on Computer Aided Design*, Nov. 2004.

[14] C. E. Leiserson and F. M. Maley, "Algorithms for routing and testing routability of planar vlsi layouts," pp. 69–78, 1985.

[15] J. Cong, M. Hossain, and N. Sherwani, "A provably good multilayer topological planar routing algorithm in IC layout designs," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 70 – 78, Jan. 1993.