PAPER

A New Approach to the Ball Grid Array Package Routing

Shuenn-Shi CHEN[†], Jong-Jang CHEN[†], Trong-Yen LEE[†], Chia-Chun TSAI[††], *and* Sao-Jie CHEN[†], *Nonmembers*

SUMMARY Due to the large number of I/O's in a Ball-Grid-Array (BGA) package, routing becomes more and more an important work. A ring-based router for the BGA package is presented in this paper to interconnect each I/O pad of a chip to a corresponding ball distributed on the substrate area. The major phases for the router consist of layer assignment, topological routing, and physical routing. Using this router, we can generate an even distribution of planar and any-angle wires to improve manufacturing yield. We have also conducted various testing examples to verify the efficiency of this router. Experiments show that the router produces very good results, far better than the manual design, thus it can be applied to the practical packaging of integrated circuits.

key words: *ball grid array (BGA), pin grid array (PGA), inversion table*

1. Introduction

In past years, package routing has been a missing link in the design process of integrated circuits because it was simple enough and could be done manually. Recently, the problem of package routing becomes more and more difficult when the I/O pin count gets larger and larger in array I/O packages, such as Pin-Grid-Array (PGA) and Ball-Grid-Array (BGA) packages, which have a complex structure and multiple substrate layers. PGA packages have played a predominant role in high-density and high-performance packaging design for many years. Nevertheless, the drawbacks of a PGA package are its high cost (of using alloy-42 or copper pins), through-holes on the PCB for mounting (not Surface-Mount-Technology compatible), handling, and pin bending and insertion [1]. Based on the reasons above, PGA packages have been gradually replaced with BGA packages since the later ones use cheaper solder balls, which are easier to attach to the bottom of the chip carrier (SMT-compatible), and allow more I/O's in less area. Therefore, developing space-effective packaging tool for PGA/BGA packages becomes indispensable [2]–[5].

A BGA package with its ball-numbers shown on a substrate and its routing design is given in Fig. 1. Given a larger number of pads around the boundary of the package and an equivalent number of solder balls distributed on the substrate area, the objective of routing on a BGA package is to complete the planar interconnection of pad-to-ball nets using the shorter detour length on one or more layers.

In the past, Yu and Dai [6] proposed an even fanout routing algorithm for a single-layer BGA package. They use a monotonic topological planar router for the fanout routing, where a pad and a ball of the same net have to be assigned to the same sector in a package. That is, the pad-to-ball nets are not allowed to cross one another and they should be symmetrical. As a result, all of the pad-to-ball nets can be easily routed on a single layer by the planar router. But in chip-set or CPU designs, the positions of solder balls are usually fixed such that the new design can reserve the same compatible function as a previous chip-set or CPU. In this case, the planar routing on a single layer for each of pad-to-ball nets becomes more difficult [7] and even causes routing task failure.

To cope with the situation above, we propose in this paper a ring-based router for the BGA package. We first cluster the array of balls to form multiple rectangular rings on the substrate area as our ring routing space [8]. Then, we try to complete the pad-to-ball nets

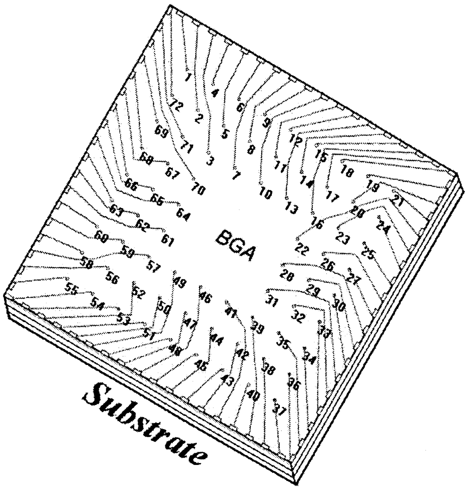


Fig. 1 A Ball-Grid-Array package with its routing shown.

Manuscript received February 1, 1999.
Manuscript revised May 27, 1999.

[†] The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, Republic of China.

[††] The author is with the Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan, Republic of China.

routing using less routing layers. Our proposed router is composed of three major phases: (1) The layer assignment phase is employed to assign nets to distinct layers. (2) The topological routing phase transforms the routing of pad-to-ball nets into a direction-constrained single row routing problem and then realizes the topology paths into a planar sketch. And (3) the physical routing phase performs a physical any-angle wiring for all the nets.

The rest of this paper is organized as follows. Section 2 gives a brief introduction to the problem formulation and routing model. Section 3 first presents an overview of the package router and then each phase of the routing algorithm will be detailed. The experimental results are reported in Sect. 4. Finally, Sect. 5 concludes this paper.

2. Problem Formulation

Assume that a BGA package contains a certain number of I/O pads arranged in a clockwise direction starting at an arbitrary side of the chip, and has the same number of solder balls distributed on the substrate area to form a multi-ring balls grid as shown in Fig. 2. Unlike the previous router [6] which divided the routing area of a BGA package into four symmetric sectors, and restricted the pad and the ball of a same net to be distributed in the same sector, the routing area that we use is a ring routing space [8] inside the chip periphery. The purpose of partitioning the routing area into such a multi-ring structure is to relax the sector restriction [6] and to handle the case of nets crisscross in the routing area. Based on the model of Fig. 2, some assumptions are made to help in developing our routing algorithm.

- The routing environment is a multi-ring structure. And nets can only be routed inside the routing area.
- The balls, each associated with a ball number, are evenly arranged in a symmetrical rectangular grids, which have a fixed ball pitch in both the vertical and horizontal directions.
- Each net is a two-terminal net, the source-terminal is a pad on the boundary of the chip, and the target-terminal is a ball distributed on the routing area. We refer to a net as a pad-to-ball net.
- One or more substrate layers are available for the planar interconnections between the pads and their corresponding balls.
- Power and ground run on individual layers.
- Each ball location is fixed and no pin assignment [9] is allowed because a compatible chip (like the *Intel mobile Pentium II-series* processors) must often have the same solder ball positions as an off-the-shelf processor.
- Two sets of lists, ring-grid lists (*RGL's*) and vertical-grid lists (*VGL's*), are used to record information on how nets are feeding through the horizontal and vertical ball pitches, respectively, and how these net numbers are inserted into corresponding ring-grid lists and vertical-grid lists. The *RGL's* structure is applied in the later topological routing and physical routing phases, while the *VGL's* one is used in the physical routing phase only.

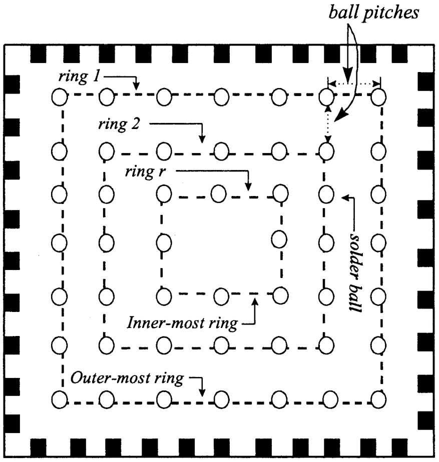


Fig. 2 Routing model of a Ball-Grid-Array package.

3. Algorithm Description

The general structure of our BGA package router is shown in Fig. 3. The position descriptions of both pads and balls are served as the input of the routing system. From the input data, we cluster the balls ring by ring to form a multiple-ring structure. The topological routing phase takes the multi-ring netlist as input and is refined into three steps: (1) divide each ring into several independent segments and then choose a suitable routing direction for each segment, (2) transform each segment into a direction-constrained single row routing problem by shifting and rotating the corresponding pads to the right (or the left) side of the segment, and (3) perform terminal-insertion to determine the topology paths for

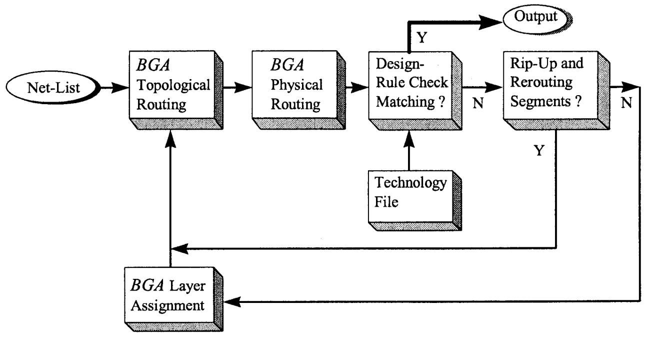


Fig. 3 General structure of the BGA package router.

nets without any crossing and generate a ring-grid list on each ring. Afterwards, the physical routing phase transforms the obtained planar sketch into a physical arbitrary-angle wires adhering to design rules. The technology file contains mainly the upper-capacity limit of wires passing through the ball pitches. We apply a rip-up and rerouting step to change the routing direction of some segments to deal with design-rule violation or to lower routing cost. The layer assignment phase is used to distribute nets to two or more layers if there are still some crossing or overcrowded nets which cannot be routed on the current routing layer after the rip-up and rerouting step. The above phases will be performed repeatedly until the routing of all nets has been done. An algorithm outline is given in Appendix and the algorithm details are depicted in the following subsections.

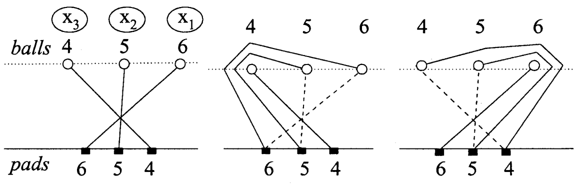
3.1 Layer Assignment

The layer assignment phase is employed to distribute nets to two or more distinct layers when some existing nets cannot be routed even after the rip-up and rerouting segment step. In the topological routing phase (Sect. 3.2), nets are routed from the outer-most ring toward the inner-most ring. Some routing factors, like crossing number, detour lengths, and capacity in a ball pitch, will affect the routing result. Therefore, we must consider the above factors in assigning nets to layers. A net interference graph based approach [10] has been proposed to perform the task of layer assignment for all nets, but it is very complex and time-consuming. Here, using the notation of inversion-table [11], we assign to each net a weight $\boldsymbol{W} = k_1V + k_2D + k_3C$, where V, D, and C represent the inversion value, inversion distance, and inversion capacity respectively and k_1, k_2, and k_3 are constants. Each of the V, D, and C factors will be described in the following subsections.

3.1.1 Measure of Inversion Value

***R-Step (L-Step)* Inversion Values and Inversion points:** Let $(a_n, a_{n-1}, \dots, a_1)$ be a permutation of a list of sorted data $\{n, n-1, \dots, 1\}$, the *R-Step* (*L-Step*) *inversion values* of $(a_n, a_{n-1}, \dots, a_1)$ are defined as $(d_n, d_{n-1}, \dots, d_1)$ where each d_i represents the number of elements located at the right (left) side of a_i and greater (smaller) than a_i. The *right* (*left*) *inversion point* of a_i is the right-most (left-most) element of a_i, which is greater (smaller) than a_i.

For example, $(2, 4, 5, 1, 3)$ is a permutation of sorted data $\{5, 4, 3, 2, 1\}$, then its *R-Step* inversion values are $(3, 1, 0, 1, 0)$, where $d_3 = 0$, means that no element at the right side of 5 is greater than 5, and there is no inversion point; $d_4 = 1$, because there is an element 5 greater than 4, and the inversion point of 4 is 5. And its *L-Step* inversion values are $(0, 1, 2, 0, 2)$, where $d_1 = 2$, because two elements (1 and 2) are smaller than element 3, and its inversion point is 2; $d_3 = 2$, because two elements (2 and 4) are smaller than element 5, and its inversion point is 2.



(a) Initial routing (b) L-Step routing (c) R-Step routing

Ball numbers	4 5 6
Inversion distances	0 1 2
Inversion points	0 4 4

(d) L-Step inversion table

Ball numbers	4 5 6
Inversion distances	2 1 0
Inversion points	6 6 0

(e) R-Step inversion table

Fig. 4 Inversion distance calculation.

3.1.2 Measure of Inversion Distance

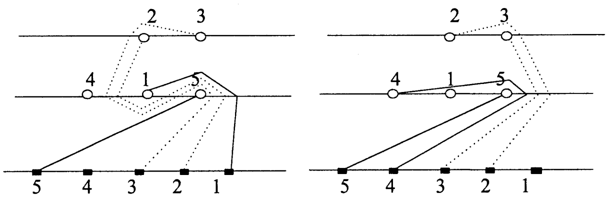
While a ball cannot be directly connected to its corresponding pad, the topological path of this net must detour at the inversion point. Here, we define the term "*inversion distance*" to express the distance from a ball position to its inversion point position. The inversion distance of ball i is defined as $D = |m - n|$, where m and n are the positions of ball i and its inversion point, ball j, respectively. Note that the more inversion distance causes the more detour length. For instance in Fig. 4 (d), since ball-number 6 (at x-coordinate 1) has an inversion point ball-number 4 (at x-coordinate 3), its inversion distance D is $|1 - 3| = 2$. Similarly, the inversion distance D for ball-number 5 (at x-coordinate 2) is $|2 - 3| = 1$.

Comparing ball-numbers 5 and 6 in the *L-Step* inversion table shown in Fig. 4 (d), we realize that ball 5 is more suitable than ball 6 to associate with ball 4 on the same layer because this will generate shorter detour length; and similar result can be observed from the *R-Step* inversion table.

3.1.3 Measure of Inversion Capacity

Since the ball pitch S_g between any two adjacent balls on a BGA package is fixed, the maximum number of nets passing through the ball pitch is limited, and this number will affect the routability. Measuring the inversion capacity C_a of ball a is equal to measuring the number of nets which may pass through between ball a and its inversion point, ball b. That is, the inversion capacity of ball a is $C_a = |a - b| - 1$.

Figure 5 shows an example of the influence of inversion capacity in a multi-ring routing model. Since ball 1 has an inversion point ball 5, thus its inversion capac-



(a) Inversion capacity = 3

(b) Inversion capacity = 0

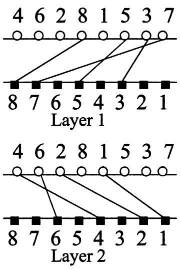
Ball numbers	4	1	5
Inversion capacities	0	3	0
Inversion points	5	5	0

(c) R-Step inversion table

Fig. 5 Relation between inversion capacity and net detour.

	R-Step inversion table								L-Step inversion table							
Ball numbers	4	6	2	8	1	5	3	7	4	6	2	8	1	5	3	7
Inv. value	4	2	4	0	3	1	1	0	0	1	0	3	0	3	2	6
Inv. distance	7	6	5	0	3	2	1	0	0	1	0	3	0	5	4	7
Inv. capacity	2	0	4	0	5	1	3	0	0	1	0	3	0	0	0	2
Weight	13	8	13	**0**	11	**4**	**5**	**0**	**0**	**3**	**0**	9	**0**	8	6	15
Inv. point	7	7	7	0	7	7	7	0	0	4	0	4	0	4	2	4

(a) Inversion tables



(b) Layer assignment

Fig. 6 An example of the layer assignment of nets.

ity C_1 is $|1-5|-1=3$, which represents that nets (2, 3, and 4) will pass through the ball pitch between ball 1 and its inversion point ball 5. Similarly, the inversion capacity C_4 for ball 4 is $|4-5|-1=0$. Assigning ball 4 with ball 5 on the same layer, as shown in Fig. 5 (b), is more suitable than assigning ball 1 with ball 5, as shown in Fig. 5 (a), because ball 4 has a smaller inversion capacity than ball 1. Thus, the detour lengths of nets 2 and 3 in Fig. 5 (b) on the outer ring will be shorter than those in Fig. 5 (a).

Finally, we apply a simple comparison procedure for the layer assignment of nets as follows. Assume we have only two available layers, then we can assign nets having minimum weights in the *L-Step* inversion table to one layer and nets having minimum weights in the *R-Step* inversion table to another layer. If the two weights of a net are equal, assigning this net to the layer having fewer nets will make the number of nets on each layer more even. For example, nets $(8,5,3,7)$ have to be assigned to layer 1 and nets $(4,6,2,1)$ to layer 2 as shown in Fig. 6. Here, the weight of each net is calculated as $\boldsymbol{W}=k_1$ (the inversion value) $+k_2$ (the inversion distance) $+k_3$ (the inversion capacity) and k_1,k_2,k_3 are set to 1.

3.2 Topological Routing

The objective of topological routing is to generate a planar topology path for all the nets. In this subsection, we will show how to transform the BGA routing into a direction-constrained single row routing problem (*SRRP*) such that its planar topology layout can be easily obtained.

3.2.1 Partition a Ring into Independent Segments

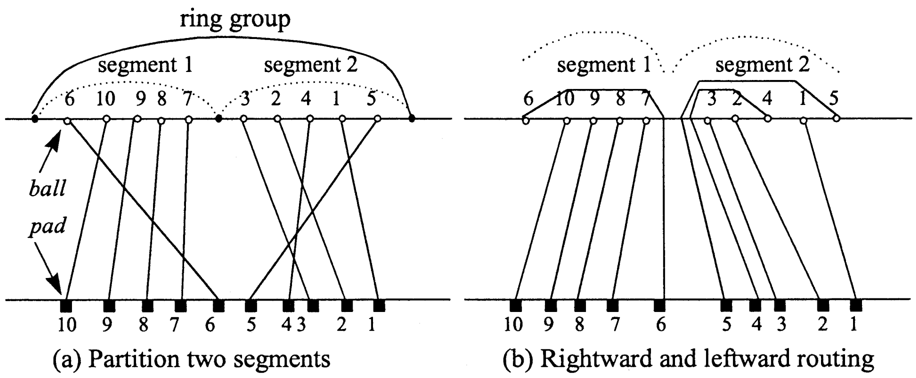
Since the topology paths of all the outer-ring balls have to pass through inner-rings toward pads, the routing of balls on an outer-ring should precede those on an inner-ring. The balls on the same ring are considered simultaneously and routed in a sequential net-ordering. Therefore, we treat all balls on a ring as a group, namely ring-group. Figure 7 shows that a ring-group is divided into two independent segments in which segment 1 and segment 2 have a rightward routing and a leftward routing, respectively. How to choose a better routing direction (toward left or right) for each segment will be described in the following.

The objective of choosing a better routing direction for nets in each segment is to obtain a shorter detour-length layout. It means that the number of wiring nets passing through ball pitches is also reduced, and thus, the electrical characteristics like noise interference, the chance of short-circuit among nets passing through each ball pitch can significantly be decreased. As a result, the packaging yield is further enhanced.

In order to shorten the detour length, we first must calculate the inversion distance D referring to the inversion point of each net. Then the weight-sums of the inversion distances in the *L-Step* and *R-Step* inversion tables will respectively be calculated. The criteria of choosing a routing direction depends on the weight-sums in the *L-Step* and *R-Step* inversion tables. A leftward routing direction is made if the weight-sum in the *L-Step* inversion table is smaller than the one in the *R-Step* inversion table and vice versa. For instance in Fig. 8, the weight-sum in the *L-Step* inversion table is smaller than that in the *R-Step* inversion table; thus, the leftward routing direction will be chosen.

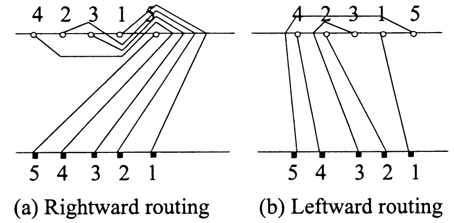
3.2.2 Transformation into a Direction-Constrained Single Row Routing Problem

In this subsection, we show how to transform the routing of each segment in a multi-ring environment into a direction-constrained single row routing problem (*SRRP*) by shifting and rotating the corresponding row of pads to the left (or the right) side of the segments. Here, the term "direction-constrained" means that the wiring direction of a pad is upward only (whereas a ball can move upward or downward) for our BGA package routing. For the example in Fig. 9, its ball-side and pad-side interval diagram and how it corresponds to a single row routing are shown in Fig. 10, where the wiring direction for pads can only be downward (after transformation). Many studies based on the approach



(a) Partition two segments
(b) Rightward and leftward routing

Fig. 7 Partition a ring-group into multiple segments.



(a) Rightward routing
(b) Leftward routing

	R-Step inv. table	L-Step inv. table
Ball numbers	4 2 3 1 5	4 2 3 1 5
Inversion distances	4 3 2 1 0	0 0 1 0 4
Weights sums	**10**	**5**
Inversion points	5 5 5 5 0	0 0 2 0 4

(c) Inversion tables

Fig. 8 Routing direction choice for a segment.

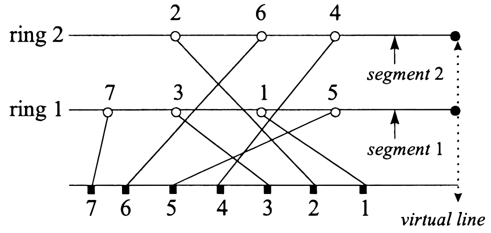


Fig. 9 An initial routing of multi-ring nets.

of interval graph for the single row routing problem have been reported [12], [13]. Here, our BGA package routing will be realized as a special case of the general single row routing problem.

Theorem 1: If there are no crossing nets on each routing segment, the planar BGA routing problem is just the problem of ordering by terminal-insertion in each ball pitch.

Proof: For any ball pitch, there are two ball-numbers P_x and P_y at its left- and right-sides, respectively. Assume that terminal-numbers $t_1, t_2, \ldots, t_k$ (called as extra nodes) are inserted into this ball pitch after the transformation of a direction-constrained single row routing, where $k \in Z^+$. The ordering by terminal-insertion is shown as follows:

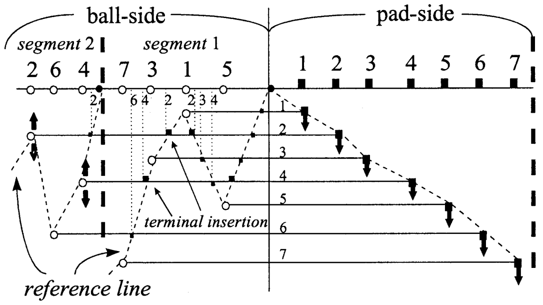


Fig. 10 Transform segments into a direction-constrained single row routing.

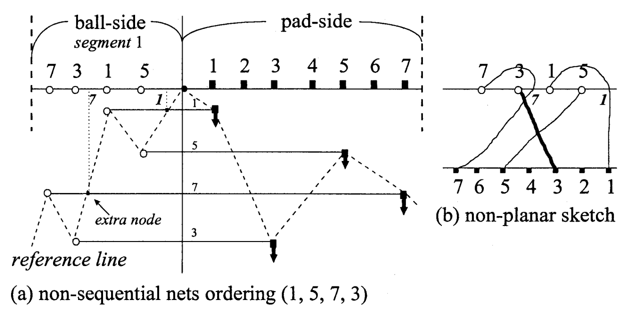
Case 1: $P_x < t_1 < t_2 < \ldots < t_k < P_y$, for ascending terminal-number order;

Case 2: $P_x > t_1 > t_2 > \ldots > t_k > P_y$, for descending terminal-number order.

According to the segment 1 as shown in Fig. 10, terminal-numbers 2, 3, and 4 are inserted into the ball pitch between ball-numbers 1 and 5, this situation is matched to Case 1. On the other hand side, terminal-numbers 6 and 4 are inserted into the ball pitch between ball-numbers 7 and 3, this situation is matched to Case 2. Consequently, the planar BGA routing problem is equivalent to an ordering by terminal-insertion problem under the direction-constrained single row routing model. □

Theorem 2: Since the BGA package routing has to be a planar sketch, the only routing ordering of nets will be sequential (i.e., nets $1, 2, 3, \ldots, n$) under the direction constraint in a single row routing.

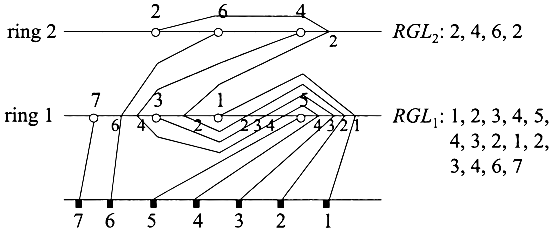
Proof: If the routing ordering of nets in a segment is non-sequential, such as it is in a random permutation $(1, 5, 7, 3)$ as shown in Fig. 11 (a); it will generate some extra nodes on the ball-side of the reference line (broken line). For this situation, an extra node 7 has to be inserted into the ball pitch between balls 3 and 1. Obviously, this will cause a non-planar BGA routing because nets 5 and 7 are crossing net 3 as shown in



(a) non-sequential nets ordering (1, 5, 7, 3)

(b) non-planar sketch

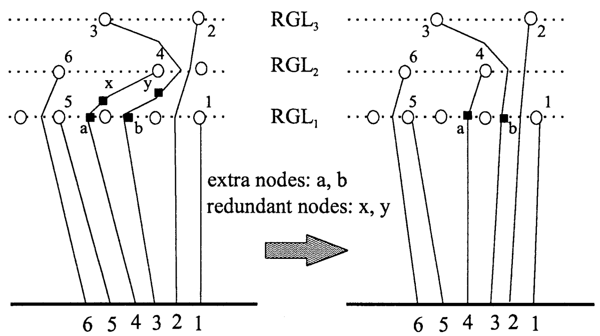
Fig. 11 Non-sequential net order causing a non-planar routing.



RGL_2: 2, 4, 6, 2

RGL_1: 1, 2, 3, 4, 5, 4, 3, 2, 1, 2, 3, 4, 6, 7

Fig. 12 A planar layout of multi-ring nets.



extra nodes: a, b
redundant nodes: x, y

Fig. 13 Redundant nodes elimination deriving wiring length reduction.

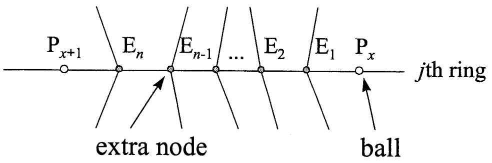


Fig. 14 Distribution of extra nodes on each ball pitch.

Fig. 11 (b). Therefore, the net-ordering should be sequential for a direction-constrained single row routing to generate a planar wiring paths for all the nets. □

Finally, we can stretch out the reference line in Fig. 10, which contains several extra nodes. As a result, we can obtain a planar topology layout by means of shifting and rotating pads back to their original positions, where the relative coordinates of balls and extra nodes are recorded from the right-side to the left-side of a ring-grid list (RGL). Therefore, an entirely planar layout for all nets in a multi-ring routing can be made as shown in Fig. 12.

3.3 Physical Routing

The input net-list has been transformed to a planar sketch and generates some of extra nodes on each ring after the topological routing phase. As balls locations are fixed and evenly distributed on the package while extra nodes are movable. Therefore, we can first attempt to move one or more extra nodes of a net to another ball pitch in order to reduce physical wiring length or to eliminate some redundant nodes. Afterwards, the physical wiring of nets will be routed in arbitrary angle. And the rip-up and rerouting of a segment is performed only if we have to accord with the design rules or to reduce routing cost. The detailed steps of this physical routing phase will be described in the following.

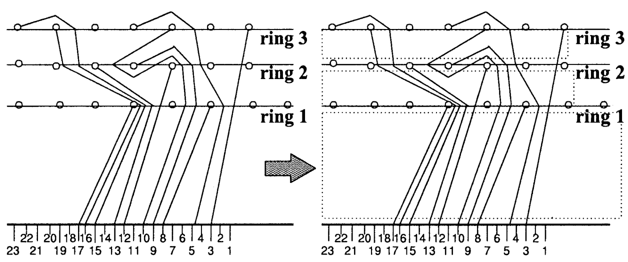
3.3.1 Redundant Nodes Reduction

In the topological routing phase, many extra nodes between ball pitches were generated on the ring-grid lists (RGL*'s*). As the relative positions (non-physical position) of extra nodes on a ring-grid list are still movable, we can try to shift these extra nodes to the right- or the left-pitch for minimizing redundant nodes, and thus reduction of wire length can be achieved in the physical routing phase. For instance, if the extra node a of net 4 and the extra node b of net 3 in Fig. 13 can be shifted to their neighboring right pitches, the wire lengths of net 3 and net 4 will be reduced in this routing phase because redundant nodes x and y have been eliminated by this move.

3.3.2 Arbitrary Angle Wiring

In this step, we hope to have the physical positions of extra nodes evenly distributed between each ball pitch to reduce the crosstalk among nets. Assume that there are some of extra nodes distributed between balls P_x and P_{x+1} on the jth ($j \in Z^+$) ring as shown in Fig. 14. The physical positions of these extra nodes have to be determined by the following formulas: $d = S_g - [W_h + \sum_{i=1}^{n} W_{net}]/n+1$, $X_{E_k} = \sum_{i=1}^{k}(d+W_{net})$ and $Y_{E_k} = Y_{P_x} = Y_{P_{x+1}}$, for $k = 1, 2, \ldots, n$. Where d is the average space between any two adjacent extra nodes located at a ball pitch, S_g denotes the space of a ball pitch, W_h denotes the diameter of a through ball hole, W_{net} denotes the width of a net, X_{E_k} and Y_{E_k} are the x- and y-coordinate of an extra node, respectively.

Assume that a routing region has been partitioned into r sub-regions by the ring-grid lists as shown in Fig. 15, where r is the number of rings in a BGA package. For each sub-region, the vertical-grid lists (VGL*'s*) must be also constructed to form several in-



(a) Original routing region (b) Sub-routing regions

Fig. 15 Partition an entire routing region into sub-regions.

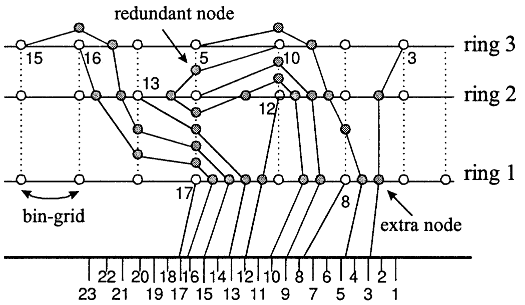
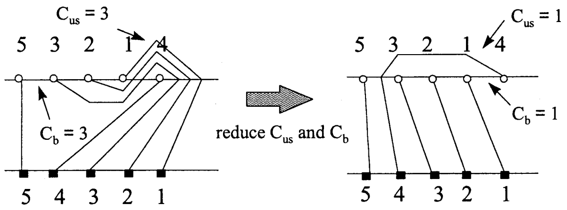


Fig. 16 Any-angle physical wiring result of Fig. 15.

dependent bin-grids rather than using the rubber-band sketch method proposed by Dai [14]. In each bin-grid, balls and extra nodes lie on the horizontal axis of the rectangular routing region (or on the ring-grid lists, *RGL's*). Additionally, there is no obstacles in the bin-grids and each segment of a net is a two-terminal net. Therefore, the routing problem in each bin-grid looks like a river routing [15] problem and can be solved by using a stack data structure. Whenever the path of any segment passes through a vertical-grid, insert its net number (named as a redundant node) into the vertical-grid lists, we call this a range insertion problem. Therefore, each net in the bin-grid can be routed by using a straight line to connect its two terminals. In such a way, an any-angle physical wiring result can be generated as shown in Fig. 16.

3.3.3 Rip-Up and Rerouting Segment

If the physical wirings of nets violate design rules, the routing direction for their corresponding segments should be adjusted. Recall that each ring has been divided into several segments and each segment is viewed as an individual group. Therefore, we can adjust the routing direction (leftward or rightward) of nets in a certain segment group in order to adhere to design rules to reduce routing cost. As shown in Fig. 17, the maximum non-zero value among the *L-Step* inversion values denotes the value of C_{us} (a maximum value of the number of vertical redundant nodes among ball pitches) obtained when we perform a rightward routing for nets in this segment. On the other hand, the maximum non-zero value among the *R-Step* inversion values denotes the value of C_{us} obtained when we perform a leftward routing for nets in this segment. Here, C_b is denoted as the maximum value of the number of nets passing through any ball pitch in this segment. From the figure, the values of C_{us} and C_b in the leftward routing are smaller than the rightward routing for this segment and the former has shorter detour length and lower wiring crowd among ball pitches, thus adheres better to design rules. Here, we say that a design-rule violation occurs when the number of extra nodes (redundant nodes) distributed on any horizontal (vertical) ball pitch is greater than the wiring capacity of a ball pitch.



(a) Rightward routing (b) Leftward routing

Ball numbers	5	3	2	1	4
L-Step inversion value	0	0	0	0	3
R-Step inversion value	0	1	1	1	0

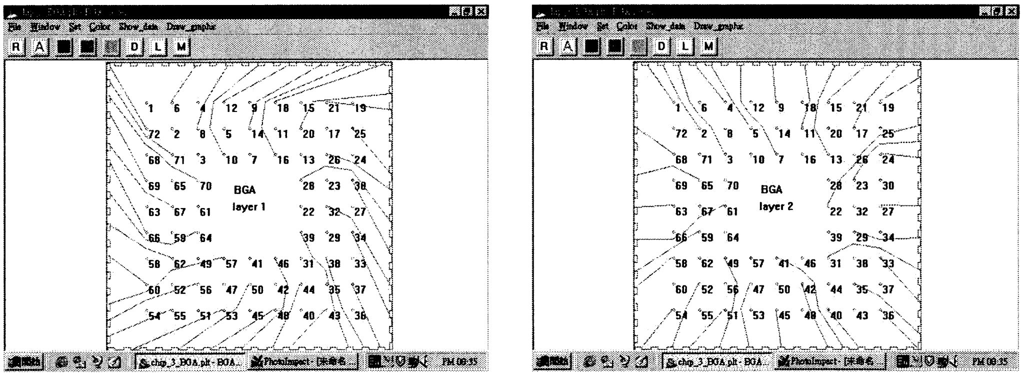
(c) Inversion table

Fig. 17 Change the routing direction of nets in a segment.

Before closing this subsection, the complexity of our package router is calculated according to the algorithm outlined in Appendix. Computing the inversion values, inversion distance, and inversion capacity in the layer assignment phase takes $O(rn^2)$, where r is the number of rings in a BGA package and n represents the average number of balls on a ring. The phase of topological routing has a complexity of less than $O(rn^2)$ because there are n balls to be sorted (inserted) into a ring-grid list. As in the physical routing phase, there are n bin-grids in each sub-region to be routed in $O(n^2)$ time using a river router and the range search operation. On average, the complexity is $O(rn^2)$. Therefore, the overall time complexity of our routing system is bounded on $O(rn^2)$.

4. Experimental Results

The routing system for the BGA package was implemented on a *Pentium II*-266 PC in Visual C++ language running Windows-98 operating system. Since no this type of benchmarks are available from the literature, a set of testing examples created by the authors are used to verify the efficiency of our BGA router. Tested results for the BGA router are reported in Table 1, which shows the number of pad-to-ball nets from 72 to 200, the number of rings, the number of routing



(a) Wiring result on layer 1

(b) Wiring result on layer 2

Fig. 18 Multi-layer wiring result of *chip*_3.

Table 1 Routing results of different BGA package examples.

chip name	# nets	# rings	# routing layers	any-angle length (pixels)	cpu-time (second)
chip_1	72	3	1	7451	0.827
chip_2	72	3	1	8050	0.832
chip_3	72	3	2	7689	0.847
chip_4	72	3	1	10383	0.842
chip_5	112	4	1	10669	0.924
chip_6	112	4	1	10172	0.927
chip_7	112	4	1	11509	0.931
chip_8	144	4	1	15021	1.012
chip_9	144	4	1	17014	1.025
chip_10	144	4	1	15798	1.108
chip_11	200	5	1	21147	1.392
chip_12	200	5	1	21012	1.390

Note: routing layers do not include power and ground layers.

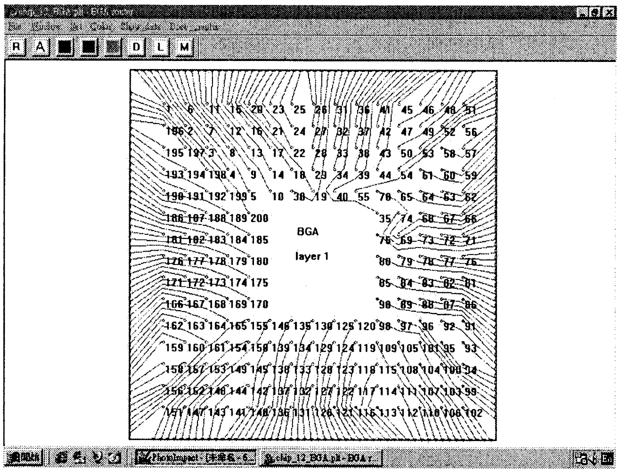


Fig. 19 Single-layer wiring result of a 200-ball BGA package (*chip*_12).

layers required, the lengths (in pixels) of arbitrary-angle wiring layout, and running time for twelve BGA packages. In all our experiments, we succeed to create a planar routing (that is, net crossing does not appear) on each routing layer. Figure 18 shows the multilayer routing results of a BGA package (*chip*_3). The single-layer wiring result of a 200-ball BGA package (*chip*_12) is plotted in Fig. 19, where some routing area is overcrowded because this chip essentially has serious crisscross among nets. Experimental results show that our router produces very good results, far better than the manual design in terms of routability and productivity.

5. Conclusions

In this paper, an innovative ring-based routing tool for BGA packages is proposed. First, topological routing is used to transform each of the routing segments on a ring into a direction-constrained single row routing problem (*SRRP*) which can be solved by terminal-insertion. Then, physical routing is applied to generate an even-distributed and any-angle wiring layout by using a river router and the range search technique for all the pad-to-ball nets. If layer assignment is needed, we calculate for each net a weight which is a linear combination of inversion value, inversion distance, and inversion capacity. According to these net weights, we apply a comparison procedure to assign nets to suitable layers.

The major contributions of our proposed package router are given as follows:

- The router has an outstanding feature, that is, it can handle the case of nets seriously overcrowded in a BGA routing area.
- Most of the tested examples can be planarized and the physical wiring be realized on a single layer under design-rule (ball-pitch capacity) consideration.
- Efficient data structures, such as grid lists *RGL's* and *VGL's*, are used from the layer assignment phase to the physical routing phase in our package router, rather than the net interference graph [10] or the complicated triangulation data structure [14].

How to accomplish the interconnection between a next-level printed circuit board (PCB) and BGA packages has been studied [16]. To extend our proposed router to generate a good BGA package routing with minimal impact to the cost and the productivity of a PCB layout will be a challenging future work.

6. Acknowledgments

The authors are grateful to the anonymous referees for the fruitful comments and suggestions on this article. This work was supported by the National Science Council, Taipei, Taiwan, R.O.C., under Grant no. NSC 88-2215-E-002-037.

References

[1] J.H. Lau, Ball Grid Array Technology, McGraw-Hill, 1995.

[2] U.A. Shrivastsva and B.L. Bui, “Inductance calculation and optimal pin assignment for the design of pin-grid-array and chip carrier packages,” IEEE Trans. Components, Hybrids, Manufacturing Technology, vol.13, pp.147–153, 1990.

[3] C.C. Tsai, C.M. Wang, and S.J. Chen, “NEWS: A net-even-wiring system for the routing on a multilayer PGA package,” IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol.17, pp.182–189, 1998.

[4] T. Hameenanttila, J.D. Carothers, and D. Li, “Fast coupled noise estimation for crosstalk avoidance in the MCG multichip module autorouter,” IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol.4, pp.356–366, 1996.

[5] D. Wang, P. Zhang, C.K. Cheng, and A. Sen, “A performance-driven I/O pin routing algorithm,” Proc. IEEE Asia and South Pacific Design Automation Conference, Hong Kong, pp.129–132, 1999.

[6] M.-F. Yu and W.W.-M. Dai, “Single-layer fanout routing and routability analysis for ball grid array,” Proc. International Conference on Computer-Aided Design, pp.581–586, 1995.

[7] D. Richards, “Complexity of single-layer routing,” IEEE Trans. Comput., vol.C-33, pp.286–288, 1984.

[8] D.C. Wang, “Pad placement and ring routing for custom chip layout,” Proc. 27th Design Automation Conference, pp.193–199, 1990.

[9] M.-F. Yu, J. Darnauer, and W.W.-M. Dai, “Interchangeable pin routing with application to package layout,” Proc. International Conference on Computer-Aided Design, pp.668–673, 1996.

[10] J.D. Cho, M. Sarrafzadeh, M. Sriram, and S.M. Kang, “High-performance MCM routing,” IEEE Design and Test of Computers, vol.10, no.3, pp.27–37, 1993.

[11] D.E. Knuth, Sorting and Searching, vol.3, Reading, Addison-Wesley, MA, 1973.

[12] E.S. Kuh, T.K. Kashiwabara, and T. Fujisawa, “On optimum single row routing,” IEEE Trans. Circuits & Syst., vol.26, pp.361–368, 1979.

[13] T.T.-K. Tarng, M. Marek-Sadowska, and E.S. Kuh, “An efficient single row routing algorithm,” IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst., vol.CAD-3, pp.178–183, 1984.

[14] W.W.-M. Dai, T. Dayan, and D. Staepelaere, “Topological routing in surf: Generating a rubber-band sketch,” Proc. 28th Design Automation Conference, pp.39–44, 1991.

[15] C.P. Hsu, “General river routing algorithm,” Proc. 20th Design Automation Conference, pp.578–583, 1983.

[16] R. Johnson, A. Mawer, T. McGuiggan, B. Nelson, M. Petrucci, and D. Rosckes, “A feasibility study of ball grid array packaging,” Proc. NEPCON East, pp.413–422, 1993.

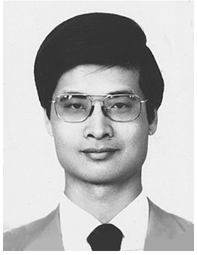
Appendix

Formally stated, the BGA package router is outlined as follows:

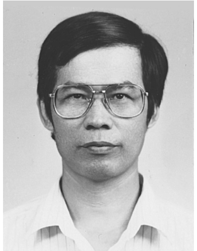
```
Algorithm BGA_Router( )
{ /* A windows-based router for the ball-grid-array (BGA) package */
  Construct a multiple-ring structure from the input
  descriptions of pads and balls;
  LAYER = 1; Complete_Flag = FALSE;
  do {
  /* perform the BGA routing layer by layer */
    Topological Routing Phase:
    for ring ← 1 to MAXRING do {
      (1) Partition a ring into several independent segments of unrouted nets and then choose a
          better routing direction for each segment;
      (2) Transform the routing of each segment into a direction-constrained single row routing
          problem;
      (3) Perform the topological planar routing and record the relative coordinates of balls and
          extra nodes on this ring into a ring-grid list (RGL);
    }
    Physical Routing Phase:
    Try to move extra nodes from one ball pitch to another ball pitch on a ring-grid list in order to
    reduce wiring length as well as eliminate redundant nodes;
    Make the physical positions of extra nodes evenly distributed in a ball pitch;
    Divide the routing region into several sub-regions and then construct vertical-grid lists (VGL's);
    Invoke the river router and the range insertion operation to obtain an any-angle wiring layout;
    /* Design-Rule Checking */
    if ( design-rule checking == TRUE ) {
      output the physical layout;
      if ( all nets are routed )
        Complete_Flag = TRUE;
    }
    else
      if ( rip-up and rerouting segments can be accepted )
        goto Topological Routing Phase;
      else {
        Layer_Assignment( ) for unrouted nets;
        LAYER ++; /* increasing the number of layer by one */
      }
  } while ( Complete_Flag == FALSE );
} /* end of the BGA routing algorithm */
```

Shuenn-Shi Chen received the B.S. degree in electronic engineering from the National Taipei Institute of Technology, Taipei, Taiwan, and the M.S. degree in computer and information science from the National Chiao Tung University, Hsinchu, Taiwan, in 1984 and 1993, respectively. He is currently working toward his Ph.D. degree in the Institute of Electrical Engineering at the National Taiwan University, Taiwan. In 1988, he served in the National Corporation, Taipei, Taiwan, as an Information System Engineer. His research aspects of interest are on VLSI physical design algorithms, genetic algorithms, and hardware-software codesign. He is a member of the Association for Computing Machinery (ACM) and the IEEE Circuits and Systems Society.

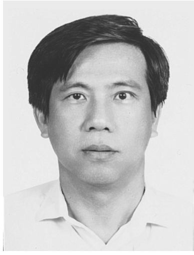
Jong-Jang Chen received the B.S. degree in electronic engineering from the Tung Nan Junior College of Technology, Taipei, Taiwan, and the M.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1988 and 1998, respectively. In 1993, He joined the Chunghwa Telecom Corporation, Taipei, Taiwan, as a Senior Computer Engineer. His research interests include VLSI physical design algorithms and telecommunication networks.



Trong-Yen Lee received the B.S. degree and M.S. degree from the National Taiwan Normal University, Taiwan, R.O.C, in 1981 and 1988, respectively. Currently he is a Ph.D. candidate in the Institute of Electrical Engineering, National Taiwan University. His current research interests include parallel computer architectures, simulation, design automation systems, and VLSI physical design automation.



Chia-Chun Tsai received the B.S. degree in industrial education from the National Taiwan Normal University, Taipei, Taiwan, R.O.C, in 1978, and the M.S. and Ph.D. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, R.O.C, in 1987 and 1991, respectively. From 1978 to 1989, he was a specialist teacher of electronic maintenance at Taiwan Provincial Hsin-Hua and Taipei Municipal Nan-Kang Technology High Schools, respectively. Since 1989 he has been with the Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C, where he is currently a Full Professor. From 1994 to 1995, he was on leave from the National Taipei University of Technology for a one-year postdoctor research at the University of California, San Diego, U.S.A. His current research interests include VLSI design automation, mixed-signal IC design, computation geometry, and FPGA applications.



Sao-Jie Chen received the B.S. and M.S. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, R.O.C, in 1977 and 1982, respectively, and the Ph.D. degree in electrical engineering from the Southern Methodist University, Dallas, U.S.A, in 1988. Since 1982, he has been a member of the faculty in the Department of Electrical Engineering, National Taiwan University, where he is currently a Full Professor. From 1985 to 1988, he was on leave from the National Taiwan University and working toward his Ph.D. degree at the Southern Methodist University. During the fall of 1987, he held a visiting appointment at the Department of Electrical and Computer Engineering, University of Wisconsin, Madison. His current research interests include VLSI physical design automation, object-oriented software engineering, and multiprocessor architecture design and simulation. Dr. Chen is a member of the Chinese Institute of Engineers, the Association for Computing Machinery, and the IEEE Computer Society.