Automatic router for the pin grid array package

S.-S. Chen, J.-J. Chen, C-C. Tsai and S.-J. Chen

Abstract: A pin grid array (PGA) package router is described. Given a chip cavity with a number of I/O pads around its boundary and an equivalent number of pins distributed on the substrate, the objective of the router is to complete the planar interconnection of pad-to-pin nets on one or more layers. This router consists of three phases: layer assignment, topological routing and geometrical routing. Examples tested on a windows-based environment show that our router is efficient and can complete the routing task with less substrate layers. Compared to manual routing, this router has a user-friendly graphic interface and can be applied practically to industrial strength VLSI packaging.

1 Introduction

The electronics industry has grown tremendously over recent times. The success has been attributed to revolutionary technology breakthroughs in the fabrication process of integrated circuits. As chips become faster, more complex and more powerful, the reliance on electronic packaging and interconnect technologies is increasing. Hence, packaging plays an important role in determining the overall cost, speed and reliability of chip design. Pin grid array (PGA) and ball grid array (BGA) packages are complex structures with large peripheral I/O counts, multiple routing layers and built-in decoupling capacitors. As a result, development of an industrial-strength CAD tool for high-speed performance package design is indispensable. In the past years, many studies on power dissipation, performance driven, noise and crosstalk for different package layouts have been reported [1–3].

A PGA package structure containing a chip cavity with a number of peripheral I/O pads and a rectangular array of pins was reported by Lewis [4]. The goal of PGA package routing is to connect each chip pad to the corresponding pin on the substrate area with one or more routing layers available. Based on this package structure [4], Tsai and Chen proposed a slope method and a rubber-band-like method for PGA package routing, respectively in [5 and 6]. Recently, Ying and Gu [7] proposed a tool for performance-driven PGA package routing on multiple layer ceramic substrates. These routers [5–7] have the common distinguishing feature that they divide the substrate routing area into four separate zones (namely top, bottom, left and right zones). But sometimes these zones are not independent of each other; for example, one pad and one pin of the same net may not belong to the same zone, or pins may be located at the borderline of two adjacent zones. Therefore, these routers cannot complete the routing task with a limited number of available layers when the input netlist becomes more complicated or there are serious crisscrosses among pad-to-pin nets over the routing area.

To overcome the drawbacks of previous routers [5–7], we propose a windows-based routing tool for the PGA package based on the model proposed in [4]. We cluster a set of grid array pins to form multiple rectangular rings and the substrate area. The structure and conceptual routing environment of this PGA package are shown in Fig. 1. The objective of the package router is to complete pad-to-pin nets routing using fewer substrate layers, as follows: a layer assignment phase is employed to assign nets to two or more distinct layers when these nets cannot be routed using one layer under design-rule constraints. A topological routing phase is used to realise the pad-to-pin net interconnections on each layer in a type of planar sketch, and a geometrical routing phase transforms the planar sketch into a physical wiring layout.

2 Problem formulation

Some studies focused on PGA packages were mentioned earlier. As shown in Fig. 1, a PGA package contains a chip cavity with a certain number of I/O pads distributed on the chip boundary, and has the same number of external pins to form a pin grid array around the chip. In addition, each I/O pad is wired to a pin on the substrate layer. The routing area is viewed as a moat space [8] between the edges of each substrate layer and the chip cavity. Previous routers [5–7] divided the routing area into four separate zones, i.e. top, bottom, left and right zones as shown in Fig. 2. This type of division has drawbacks for routing on a PGA

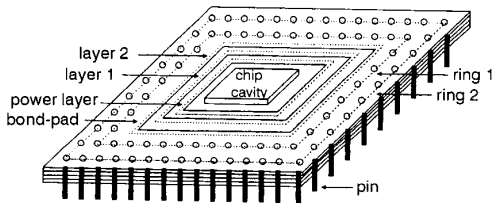


Fig. 1 *Routing environment of a PGA package*

IEE Proceedings online no. 19990797

DOI: 10.1049/ip-cdt:19990797

Paper first received 27th October 1998 and in revised form 23rd September 1999

S.-S. Chen, J.-J. Chen, and S.J. Chen are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

C.-C. Tsai is with the Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan

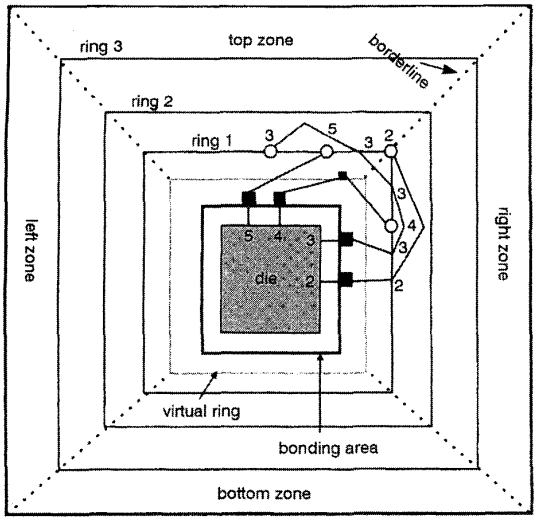


Fig. 2 *The borderline problem in PGA routing*

package. First, if a pad and a pin of the same net are distributed in distinct zones (say nets 3 and 4) or located on the borderline (say net 2) between two adjacent zones as shown in Fig. 2, these routers cannot decide to which zone the pad-to-pin net belongs. These router will either fail to route or need more routing layers. Second, they cannot handle crisscrossing nets on a routing area. Therefore, we partition the routing area into a multiring structure where each rectangular ring consists of a set of grid array pins, to eliminate the drawbacks of previous routers [5–7] for a PGA package. To explain the algorithm, the following assumptions are made based on the routing model of Fig. 1.

- Each ring structure of I/O pads and pins is arranged in a row by starting at the left top corner in a clockwise direction.
- The pins are arranged in a symmetrical square grid and the pin pitches are evenly distributed in vertical and horizontal directions.
- Each net is a two-terminal net, the source terminal is a pad on the boundary of the chip cavity, and the target terminal is a pin distributed on the routing area. We address a net as a pad-to-pin net. The routing sequence for pad-to-pin nets starts from the innermost ring towards the outermost ring and each net path consists of several arbitrary-angle wire segments on the same routing layer.
- One or more layers are available for planar interconnections between the pads and their corresponding pins.
- Each pin location is fixed and no pin assignment [9] is allowed because the package may be compatible with the pin locations of an off-the-shelf processor (e.g. the pin locations of an AMD K6 series processor have to be compatible with the Intel Pentium series).
- The routing results must satisfy design rules which specify the necessary spacing between adjacent wire segments and the capacity between two adjacent pins for wiring segments to pass through.

3 Algorithm overview

The design flow of the proposal PGA package routing system is shown in Fig. 3. Two net lists of pad and pin descriptions are input to the system. Based on the routing model, we first cluster the pins ring by ring to form a

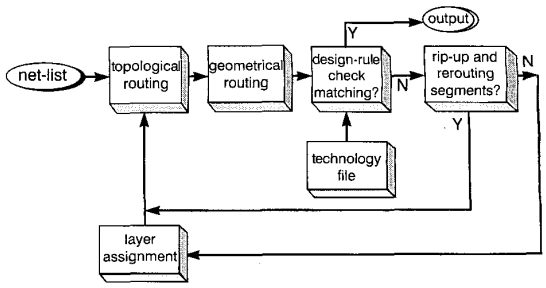


Fig. 3 *Design flow of the package router*

multiple-ring structure from the input data. The topological routing phase having this multiring netlist as input includes three main steps: (1) it transfers each ring into a row of several independent segments and then chooses a better routing direction for each segment; (2) it converts each segment into a direction-constrained single-row routing problem (*SRRP*) by shifting and rotating the corresponding pads to the left or the right; and (3) it determines the topology paths for nets without any crossing and generates a planar sketch. The geometrical routing phase then generates an arbitrary-angle wiring layout incorporating design-rule checking for all nets. The technology file contains the design rule that states the upper-capacity limit of wires passing through the spacing between any two adjacent pins. We use a rip-up and rerouting step to change the routing direction of some segments if design-rule violation or routing cost has to be considered. The layer assignment phase is employed to distribute nets to two or more distinct layers when some existing nets cannot be routed on the current layer after the rip-up and rerouting step. The above phases are repeated until the routing of all nets has been completed.

4 Routing algorithm

The most important phases in the algorithm are layer assignment, topological routing and geometrical routing, as presented in the following subsections.

4.1 Layer assignment

In the topological routing phase nets are routed ring by ring and the priorities of nets are ordered from the innermost ring to the outermost ring. Some routing factors, such as crossing number, detour length and capacity between two adjacent pins, will affect topological routing. As a result, the layer assignment phase is subject to these factors. Here, using a notation similar to the inversion table [10], we assign to each net a weight $W = \alpha V + \beta D + \gamma C$, where V, D and C represent the inversion value, distance and capacity, respectively, and α, β and γ are constants.

4.1.1 Inversion value: R2L (L2R) inversion values and inversion points. Let $(a_n, a_{n-1}, \ldots, a_1)$ be a permutation of sorted data $\{n, n-1, \ldots, 1\}$, then $(d_n, d_{n-1}, \ldots, d_1)$ represent the *R2L* (*L2R*) inversion values of $(a_n, a_{n-1}, \ldots, a_1)$, where every 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 rightmost (leftmost) element of a_i, which is greater (smaller) than a_i.

For example, (3, 2, 4, 1, 5) is a permutation of sorted data {5, 4, 3, 2, 1}, then its *R2L* inversion values are (2, 2, 1, 1, 0), where $d_1 = 0$, means that no element at the right side of 5 is greater than 5; thus, there is no inversion point for 5; $d_4 = 2$, because there are two elements (4 and 5) greater than 2, and the inversion point of 2 is 5. Its *L2R* inversion values are (0, 0, 2, 0, 4), where $d_1 = 4$, because four elements (3, 2, 4, and 1) are smaller than element 5, and its inversion point is 3; $d_3 = 2$, because two elements (3 and 2) are smaller than element 4, and its inversion point is 3.

4.1.2 Inversion distance: While a pin cannot directly be 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 calculate the distance from a pin position to its inversion point position. The inversion distance of pin i is defined as $D = |m - n|$, where m and n are the positions of pin i and its inversion point, pin j, respectively. Note that greater inversion distance causes greater detour length. For instance in Figs. 4 and 5*a*, since pin 9 (at x-coordinate 1) has an inversion point pin 8 (at x-coordinate 3), its inversion distance D is $|1 - 3| = 2$. Similarly, the inversion distance D for pin 7 (at x-coordinate 2) and pin 8 (at x-coordinate 3) are 0.

Comparing pin 7 and pin 9 in the *L2R* inversion table shown in Fig. 5*a*, we find that pin 7 is more suitable than pin 9 to be with pin 8 on the same layer because it will generate shorter detour length; a similar result can be observed from the *R2L* inversion table Fig 5*b*.

4.1.3 Inversion capacity: Since the pin pitch between any two adjacent pins on a PGA package is fixed, the maximum number of nets passing through the pin pitch is limited, and this number will affect the routability. The inversion capacity C_i of pin i is equal to the number of nets which may pass through between pin i and its inversion point pin j. That is, the inversion capacity of pin i is given by $C_i = |i - j| - 1$.

An example illustrating the influence of inversion capacity in a multiring routing model is shown in Fig. 6. Since pin 1 has an inversion point pin 5, its inversion capacity C_1 is $|1 - 5| - 1 = 3$, which means that nets (2, 3 and 4) will pass through the pin pitch between pin 1 and its inversion

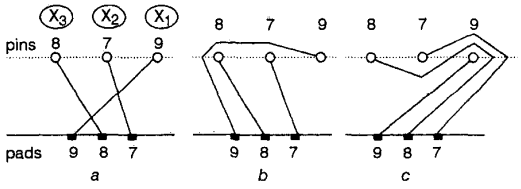


Fig. 4 *Inversion distance calculation*
a Initial routing
b L2R routing
c R2L routing

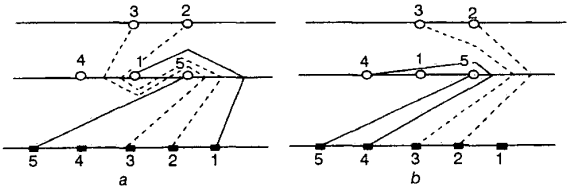
Pin numbers	8	7	9
Inversion distances	0	0	2
Inversion points	0	0	8

a

Pin numbers	8	7	9
Inversion distances	2	1	0
Inversion points	9	9	0

b

Fig. 5 *Inversion tables*
a L2R inversion table
b R2L inversion table



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

c

Fig. 6 *Influences of inversion capacity*
a Inversion capacity = 3
b Inversion capacity = 0
c R2L inversion table

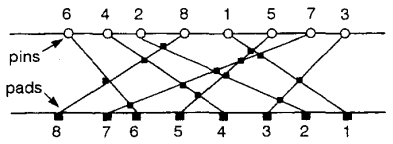
point pin 5. Similarly, the inversion capacity C_4 for pin 4 is $|4 - 5| - 1 = 0$. Assigning pin 4 with pin 5 on the same layer as shown in Fig. 6*b*, is more suitable than assigning pin 1 with pin 5 as shown in Fig. 6*a*, because pin 4 has a smaller inversion capacity than pin 1. Thus, the detour lengths of nets 2 and 3 on the outer ring in Fig. 6*b* will be shorter than Fig. 6*a*.

Finally, we can start to process the layer assignment of nets using a simple comparison procedure as follows. Assume we have only two available layers, then we can assign nets having minimum weights in the *L2R* table to one layer and nets having minimum weights in the *R2L* 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. An example, is shown in Fig. 7, where nets (8, 5, 7, 3) have to be assigned to layer 1 and nets (6, 4, 2, 1) to layer 2.

4.2 Topological routing

The goal of topological routing is to generate a planar routing sketch for all the nets. In this section, we transform the PGA routing into a direction-constrained single-row routing problem (*SRRP*), such that its planar topology layout can easily be obtained.

4.2.1 Partition a ring into several independent segments: The routing of pins on an inner ring should precede pins on an outer ring because the topology routing paths of nets in outer rings have to pass through inner rings. To obtain a shorter detour length, the pins on the same ring are considered simultaneously and routed in



	R2L inversion table								L2R inversion table							
Pin numbers	6	4	2	8	1	5	7	3	6	4	2	8	1	5	7	3
Inversion values	2	3	4	0	3	1	0	0	0	0	0	3	0	3	5	2
Inversion distances	6	5	5	0	3	1	0	0	0	0	0	3	0	4	6	5
Inversion capacities	0	2	0	0	1	1	0	0	0	0	0	1	0	0	0	0
Weights	8	10	9	0	7	3	0	0	0	0	0	7	0	7	11	7
Inversion points	7	7	3	0	3	7	0	0	0	0	0	6	0	4	6	2

$\alpha = 1, \beta = 1, \gamma = 1$

Fig. 7 *An example of nets layer assignment*

a sequential net ordering. Thus, we treat all pins on a ring as a group, namely a ring group. Then, we try to divide each ring group into different independent segments such that a better routing direction for nets in each segment can be chosen. The objective is to obtain a shorter detour length layout and to reduce the number of wiring nets passing through pin pitches.

To choose a better routing direction (to the left or right) for each segment, first we must find the inversion point of each net and calculate its inversion distance D. Then, we sum the inversion distances in the *L2R* and *R2L* inversion tables into two weight-sums. The criterion for choosing a routing direction depends on the weight-sums in the *L2R* and *R2L* inversion tables. A leftward routing direction is made if the *L2R* weight-sum is smaller than in the *R2L* one and vice versa. For example in Fig. 8, a ring group is divided into two independent segments in which segment 1 and segment 2 have a leftward routing and a rightward routing, respectively.

4.2.2 Direction-constrained single-row routing problem: We transform the routing of each segment into a direction-constrained single-row routing problem by shifting and rotating the corresponding row of pads to one side (left or right). Here, the term 'direction-constrained' means that the wiring direction of a pad is upwards only (whereas a pin can move upwards or downwards) and must be away from the chip cavity for our PGA package routing. For instance, a pin-side and pad-side interval diagram and how it corresponds to a single row are shown in Fig. 9*a*, where the wiring direction of a pin can be upwards or downwards while the wiring direction for pads can only be downwards (after transformation). Many studies on solving the single-row routing problem have been undertaken [11, 12]. Here, our PGA package routing can be realised as a special case of the single-row routing problem.

Theorem 1: Since the routing in a PGA package has to be a planar sketch, the routing ordering of nets should be sequential under the direction constraint in a single-row routing.

Proof: For simplicity, we map all pads of a segment to the right of its corresponding pin row. If the routing ordering of nets in a segment is nonsequential, such as the one arranged by (1, 4, 2, 6, 7) as shown in Fig. 9*b* this will cause some extra nodes on the pad-side of the reference line (broken line). For this situation, the wiring directions of some pads have to go towards the bottom, such as pad 4 in Fig. 9*c*. That is, the wiring path of pad 4 will pass through the chip cavity area. Obviously, this will contradict the direction constraint of our PGA routing model. Therefore, the net ordering should be sequential (that is, nets 1, 2, 3, . . . , n) for a direction-constrained single-row routing to generate a planar sketch for all the wiring paths.

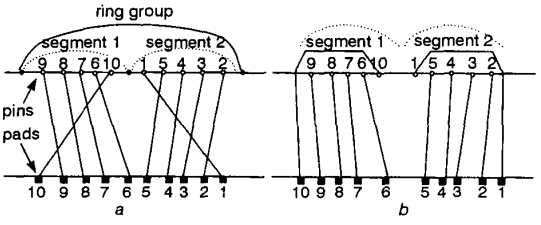


Fig. 8 *A ring divided into several independent segments*
a Partition two segments
b Leftward and rightward routings

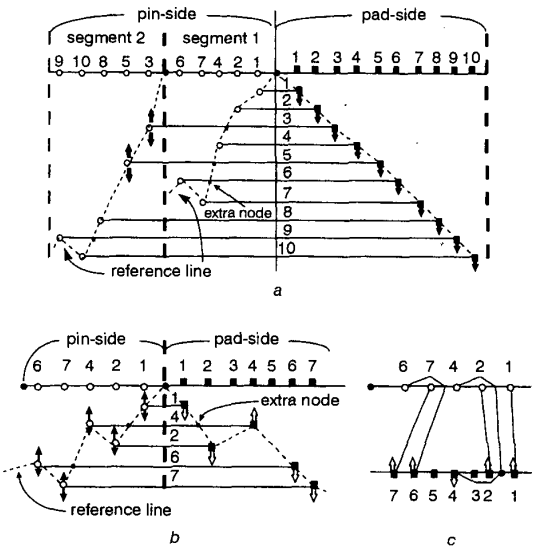


Fig. 9 *Direction-constrained single-row routing problem*
a Transforming segments to a single row
b Nonsequential routing ordering
c Constraints violation

Finally, we can stretch out the reference line for segment 1 in Fig. 9*a* to obtain its direction-constrained Manhattan routing as shown in Fig. 10. In our PGA routing environment, because the routing sequence runs from the inner ring towards the outer ring, a multiring routing can be transformed into a multisegment single-row routing problem. As a result, by means of shifting and rotating pads back to their original positions, we can also achieve a planar topology layout, where the relative coordinates of pins and extra nodes are recorded on ring-grid lists (RGLs). For example in Fig. 11, a planar sketch layout for nets in a multiring routing can be made.

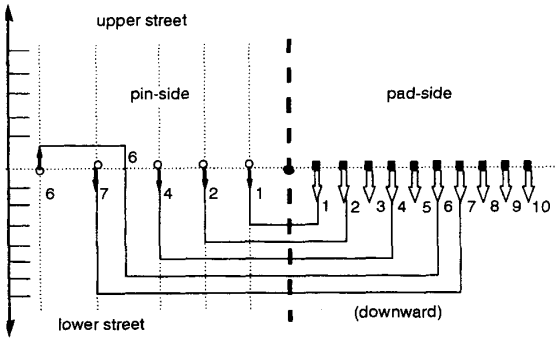


Fig. 10 *Direction-constrained Manhattan routing result*

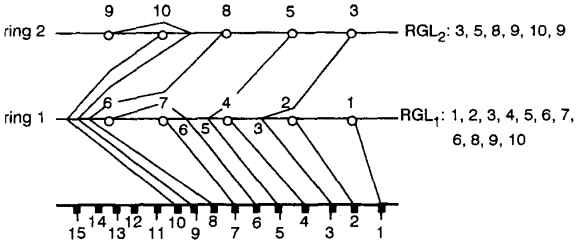


Fig. 11 *Planar sketch layout for fig. 9a*

4.3 Geometrical routing

After the topological routing phase, the extra nodes are distributed over horizontal pin pitches on each ring-grid list. We first attempt to move one or more extra node(s) of a net to another pin pitch to reduce the wiring length or to eliminate redundant nodes. Then, the geometrical layout is routed at an arbitrary angle for all the nets. Finally, the rip-up and rerouting of a segment is performed only if we have to adhere to the design rules or to lower routing cost.

4.3.1 Elimination of redundant nodes: Since the relative positions (not physical positions) of any extra nodes on each ring-grid list are still movable, we can try to move these extra nodes to reduce the wiring length or to minimise redundant nodes generated in the geometrical routing phase. If an extra node of a net can be moved to another pin pitch to reduce the wiring length, many redundant nodes on the vertical grid list will also be reduced. For example, if the extra node a of net 4 and the extra node b of net 3 in Fig. 12 can be moved to their respective pin pitches, the wire lengths of net 3 and net 4 will be reduced and the redundant nodes c and d will be eliminated in this routing phase.

4.3.2 Arbitrary angle routing: Previous work by Dai [13] transformed a topological routing into a geometrical routing using a rubber-band sketch. We hope to have the physical positions of extra nodes evenly distributed between each pin pitch to reduce the crosstalk among nets. Assume that there are m extra nodes located at each pin pitch from left to right arrangement. The physical locations of extra nodes are calculated by the following formulas:

$$d = \frac{S_g - \left[W_h + \sum_{i=1}^{m} W_{net}\right]}{m+l}, X_{E_k} = \sum_{i=l}^{k}(d + W_{net}),$$

$$\text{and } Y_{E_1} = Y_{E_k}, \text{ for } k = 1, 2, \ldots, m$$

where d is the average space between any two adjacent extra nodes located at a pin pitch, S_g denotes the space of a pin pitch, W_h denotes the diameter of a pin hole, W_{net} denotes the width of each net wire, and X_{E_k} and Y_{E_k} are the x- and y-coordinates of an extra node, respectively.

In our geometrical routing phase, vertical grid lists (VGLs) must be created between any two adjacent ring grid lists (RGLs) to form a set of independent routing bin grids. In each bin grid, pins and extra nodes (denoted as RGL numbers) lie on the horizontal axis of the rectangular routing region. The interconnections of all the net segments can be accomplished by using a river router [14] with a hash table, where each hash table slot contains a linked list of RGL numbers. The principle of connecting the segments of each net and net ordering is described as follows: assume that we are working on a ring region formed between the ith and $(i+1)$th rings, and are scanning from the rightmost RGL number of the ith ring to the rightmost RGL number of the $(i+1)$th ring in a clockwise direction. Here, let the head item of each linked list in the hash table be denoted as a hash item and an item is always inserted at the head of the linked list. During scanning, an RGL number is inserted into the hash table if this RGL number is not found in the hash table. Otherwise, case (1): when the hash-item and the RGL number are distributed on the same bin grid, then they are connected by a segment, and this RGL number is inserted into the hash table; case (2): if not the above case, one or more redundant node(s) have been inserted into VGL(s), then we can connect the redundant node(s) to both the hash item and RGL number, and insert this RGL number into the hash table. The above process is continued until a scanned cycle is achieved. Similarly, the routing of other ring regions can also be made. For further explanation, an example of some partial nets routing is shown in Fig. 13.

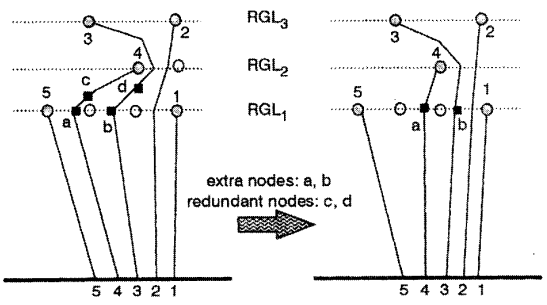


Fig. 12 *Redundant nodes and wiring length reduction*

4.3.3 Rip-up and Rerouting: Since each ring has been divided into several segments, and each segment is an independent group, we can change the routing direction of some segment groups in consideration of design rule checking or routing costs, as shown in Fig. 14. Note that the inversion values in the *L2R* and *R2L* inversion tables indicate the C_{us} (number of redundant nodes) of each pin, that is, the *L2R* inversion value of a pin denotes a C_{us} obtained when we route all nets from the right side (i.e. rightward routing), and the *R2L* inversion value of a pin denotes a C_{us} obtained when we route nets from the left side (leftward routing). And C_b is used to denote the maximum value of the number of nets passing through any two adjacent pins in a segment. Hence, the result of

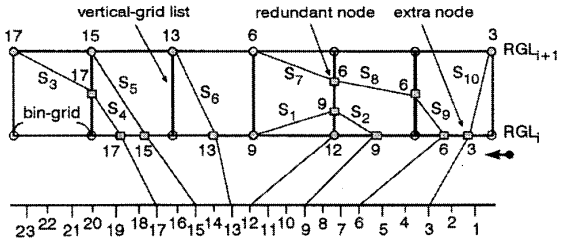
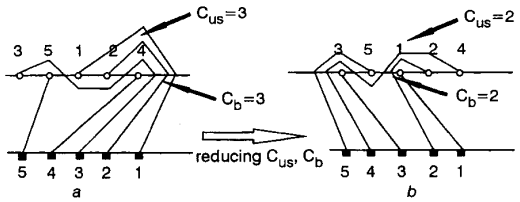


Fig. 13 *An example of any-angle routing of some partial nets*



PIN numbers	3	5	1	2	4
L2R inversion values	0	1	0	1	3
R2L inversion values	2	0	2	1	0

c

Fig. 14 *Change of a segment routing direction*

a Rightward routing
b Leftward routing
c Inversion table

```
Algorithm PGA_Router( )
{ /* A windows-based router for pin-grid-array (PGA) package */
  Read the input net lists of pads and pins;
  Construct a set of pins ring by ring to form a multiple-ring structure from the input data;
  routed_flag = FALSE;        LAYER = 1;
  do {
  /* perform the PGA routing layer by layer */
    Topological Routing Phase:
    for ring ← 1 to MAXRING do {
      Partition a ring into several independent segments of unrouted nets;
      Perform the routing direction choice for each segment;
      Transform the routing of each segment into a direction-constrained single row
       routing problem;
      Record the relative coordinates of pins and extra nodes in this ring into a RGL;
    }
    Geometrical Routing Phase:
    Try to move one or more extra node(s) of a net on each RGL to another pin pitch
     in order to reduce wiring length or to eliminate redundant nodes;
    Invoke a river router to obtain an arbitrary-angle geometrical layout;
    /* Design-Rule Checking */
    if (design-rule checking = = TRUE ) {
      output the geometrical layout;
      if ( all nets are routed )
          routed_flag = TRUE;
    }
    else
      if (rip-up and rerouting of segments can be satisfied)
        goto Topological Routing Phase;
    else {
      Layer_Assignment( ) for unrouted nets;
      LAYER + +; /* increasing the number of layer by one */
      }
  } while (routed_flag = = FALSE);
}/* end of the PGA algorithm */
```

Algorithm I. Summary of PGA algorithm

leftward routing is better than the rightward routing for the example in Fig. 14. Here, we say that a design rule violation occurs when the number of extra nodes (redundant nodes) distributed on any horizontal (vertical) pin pitch is greater than the real wiring capacity of a pin pitch.

An outline of our router is depicted in Algorithm I, which includes layer assignment, topological routing and geometrical routing phases. Computing the inversion value, inversion distance and inversion capacity in the layer assignment phase takes $O(rn^2)$, where r is the number of rings and n represents the average number of pins on a ring. In the topological routing phase, it takes $O(rn^2)$ because there are n pins to be sorted (inserted) into a ring grid list. As in the geometrical routing phase, there are n bin grids in each ring region to be routed in $O(n)$ time using a river router and a hash table. On average, the complexity is $O(rn)$. Therefore, the overall time complexity of our router is $O(rn^2)$.

5 Experimental results

This Section describes the experimental results obtained from the PGA router, which was implemented in Visual C++ language on a Pentium II-300 PC running Windows-98. Because this type of benchmark is not commonly available in the literature [7, 9], some testing examples created by the authors are used to test the router. The data listed in Table 1 show the cavity size of PGA, the number of nets from 84 to 200, the number of rings, the number of routing layers required, and the lengths of any-angle wiring layouts, for twelve PGA packages. A single-layer wiring layout for a 200-pin PGA package (*chip_12*) is plotted in Fig. 15; this had serious crisscrosses among nets before the routing was done. As shown, the proposed router can improve the common drawback of existing routers [5–7], which fail to deal with cases where one pad and one pin of the same net are distributed on distinct routing zones (that is, it causes the borderline problem) and where there are serious crisscrosses among nets. Obviously, the proposed router performs significantly better than previous routers [5–7] and manual designs in terms of routability and productivity, and it has a user-friendly graphic interface for designers in the electronic packaging industry.

6 Conclusions

An innovative windows-based routing tool for PGA package routing on a multilayer substrate has been described. The router includes the following three phases: layer assignment, topological routing and geometrical routing phases. In the layer assignment phase, we calculate the weight values for each net and then apply a comparison procedure to distribute unrouted nets to suitable layers. The topological routing phase generates a planar routing sketch. This was done by converting the topological routing into a direction-constrained single-row routing

Table 1: Routing results of distinct PGA packages

Chip name	Cavity size (pixels)	Number of nets	Number of rings	Number of routing layers	Any-angle length (pixels)
chip_1	104 × 104	84	3	1	8428
chip_2	104 × 104	84	3	1	9251
chip_3	110 × 110	84	3	2	8361
chip_4	110 × 110	144	4	1	12726
chip_5	110 × 110	144	4	1	14689
chip_6	110 × 110	112	4	1	14396
chip_7	110 × 110	112	4	1	14249
chip_8	110 × 110	200	5	1	28240
chip_9	110 × 110	200	5	1	23450
chip_10	110 × 110	200	5	1	23482
chip_11	110 × 110	200	5	1	23452
chip_12	110 × 110	200	5	1	24199

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

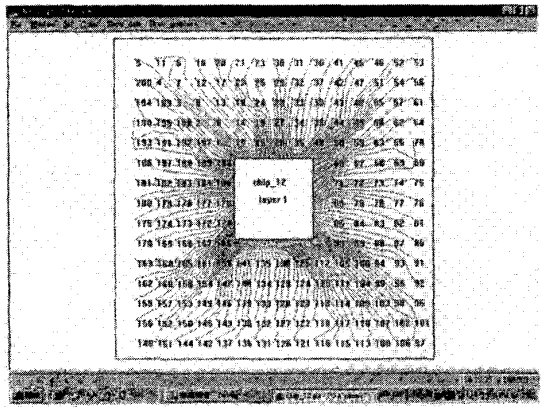


Fig. 15 *Single-layer routing result of a 200-pin PGA package*

problem, where routing strategy with sequential net ordering in a multiring routing environment is adopted to shorten the routing length. Finally, the geometrical routing phase partitions the routing region into several independent bin grids such that we can employ a river router to find an any-angle wiring layout for all the pad-to-pin nets. The router has the following undisputed advantages:

- It can eliminate the borderline problem [5–7] which occurs in PGA packages and handle serious crisscrosses among nets in the routing area.
- The planar routing of all nets could be realisable on a single layer if no constraint exists on the capacity between any two adjacent pins.
- The even-distributed any-angle layout improves the yield as well as reducing the manufacturing cost.

We have shown in [15] how to extend this router to handle the routing problem of ball-grid-array (BGA) packages as proposed in [16] because both PGA and BGA packages preserve equivalent topological characteristics. Moreover, while the number of peripheral I/O pads increases in multichip modules (MCM), applying our package router to an MCM environment will be a new challenge.

7 Acknowledgments

The fruitful comments and suggestions from the anonymous referees are highly appreciated. This work was supported by the National Science Council, Taipei, Taiwan, under Grant no. NSC 89-2213-E-002-097.

8 References

1 SHRIVASTSVA, U.A., and BUI, B.L.: 'Inductance calculation and optimal pin assignment for the design of pin-grid-array and chip carrier packages', *IEEE Trans. Compon. Hybrids Manuf. Technol.*, 1990, **13**, pp. 147–153
2 MORESCO, L.L.: 'Electronic system packaging: The search for manufacturing the optimum in a sea of constraints', *IEEE Trans. Compon., Hybrids Manuf. Technol.*, 1990, **13**, pp. 494–508
3 WANG, D., ZHANG, P., CHENG, C.K., and SEN, A.: 'A performance-driven I/O pin routing algorithm', Proceedings of IEEE Asia and South Pacific *Design Automation* Conference, 1999, Hong Kong, pp. 129–132
4 LEWIS, E.T.: 'The VLSI package-an analytical review', *IEEE Trans. Compon. Hybrids Manuf. Technol.*, 1984, **CHMT-7**, pp. 197–201
5 TSAI, C.C., and CHEN, S.J.: 'Planar routing on a pin grid array package', Proceedings of 3rd International Conference on *CAD & Computer Graphics*, 1993, Beijing, China, pp. 439–445
6 TSAI, C.C., WANG, C.M., and CHEN, S.J.: 'NEWS: A net-even-wiring system for the routing on a multilayer PGA package', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 1998, **17**, pp. 182–189
7 YING, C., and GU, J.: 'Automated pin grid array package routing on multilayer ceramic substrates', *IEEE Trans. Very Large Scale Integr. Syst.*, 1993, **1**, pp. 571–575
8 McGEHEE, R.K.: 'A practical moat router', Proceedings 24th *Design Automation* Conference, 1987, pp. 216–222
9 YU, M.-F., and DAI, W.W.-M.: 'Pin assignment and routing on a single-layer pin grid array', Proceedings of IEEE Asia and South Pacific *Design Automation* Conference, 1995, pp. 203–208
10 KNUTH, D.E.: 'Sorting and searching' (Addison-Wesley, Reading, MA 1973), **3**
11 KUH, E.S., KASHIWABARA, T.K., and FUJISAWA, T.: 'On optimum single row routing', *IEEE Trans. Circuits Syst.*, 1979, **26**, pp. 361–368
12 SHERWANI, N., WU, B., and SARRAFZADEH, M.: 'Algorithms for minimum bend single row routing', *IEEE Trans. Circuits Syst.*, 1984, **39**, pp. 178–183
13 DAI, W.W.-M., KONG, R., and SATO, M.: 'Routability of a rubber-band sketch', Proceedings of 28th *Design Automation* Conference, 1991, pp. 45–48
14 HSU, C.P.: 'General river routing algorithm', Proceedings of 20th *Design Automation* Conference, 1983, pp. 578–583
15 CHEN, S.S., CHEN, J.J., LEE, T.Y., TSAI, C.C., and CHEN, S.J.: 'A new approach to the ball grid array package routing', *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*1999, **E82-A**, (11) pp. 2599–2608
16 YU, M.-F., and DAI, W.W.-M.: 'Single-layer fanout routing and routability analysis for ball grid array', Proceedings of International Conference on *Computer-Aided Design*, 1995, pp. 581–586