設 計 自 動 化 58－4
(1 9 9 1. 7. 12)

PRIDE: A Printed Wiring Board Designing System for Analog Circuits by Graph-Planarization and Rectangular-Dualization

Keiichiro Iwamoto, Toshimasa Watanabe, Takuya Yasui and Kenji Onaga

Faculty of Engineering, Hiroshima University
4-1, Kagamiyama 1-chome, Higashi-Hiroshima, 724 Japan

The purpose of this paper is to outline a printed wiring board designing system for analog circuits, PRIDE(PRInted wiring board DEsign system). The main point is that placement and routing are based on graph-planarization and rectangular-dualization. Only single-layered board design is considered. Given a set of net lists of an analog circuit, a graph representing not only connection requirements but also some physical conditions is constructed from the net lists. Then we find a planar spanning subgraph as well as a set of jumpers. Connection requirements represented by this planar subgraph will be embedded without violating physical conditions or increasing total wire length. In order to consider a placement together with connection requirements, this planar subgraph is rectangular-dualized by adding some new edges and vertices so that the resulting graph may have a rectangular-dual (a partition of a whole rectangle into some subrectangles). Rectangles are provided for parts, terminals, and areas for wiring. Actual sizes of subrectangles are determined by means of a linear programming. In PRIDE, compaction is incorparated in this step. The detailed routing is separated into two stages: interconnection among terminal-rectangles of distinct parts; connection between every pair of an actual terminal and its terminal-rectangle within each part-rectangle. The latter may require some part-rectangles to be enlarged by repeating a linear programming.

PRIDE：グラフの平面化と矩形双対グラフを用いた プリント基板設計システム

岩元圭一郎　渡辺敏正　安井卓也　翁長健治
広島大学　工学部
〒724　東広島市鏡山１－４－１

概要

本稿の目的は平面グラフ化と矩形双対グラフを利用した、アナログ回路用プリント基板設計システムPRIDEの概要を述べることである。以下では一層基板設計を扱うが、提案手法は多層基板設計への基礎であると同時に応用可能ないくつかの手法も含んでいる。プリント基板設計において，部品配置、配線，及びコンパクションは重要なステップである。本論文では部品配置と配線を取り上げるが、コンパクションは配置・配線において組み込まれている。まず，ネットリストとして与えられる回路図より接続要求をグラフモデル化し、そのグラフから平面部分グラフを抽出する。この際に除去される辺がジャンパー線となる。得られた平面グラフ（平面埋め込み）に新しく辺あるいは点を付加することにより矩形双対グラフを構成する。更に、部品矩形、端子矩形、及び配線領域矩形の縦横長の下限値を制約条件として、線形計画法により、制約条件を満たす最小矩形分割を得る。（ここで一種のコンパクションが組み込まれている。）前述の平面グラフで表現された接続要求をこの配置に埋め込み、配線を決定する。詳細配線は異なる部品の端子矩形間配線と各部品矩形内における実端子と対応する端子矩形間配線の二種類に分けられる。前者は矩形双対グラフの作り方より容易である。後者は部品矩形サイズの拡大を要求することになるかもしれない。その場合には線形計画法を再度実行して各矩形サイズを決定する。

1. Introduction and Motivation

The purpose of this paper is to outline PRIDE(PRInted wiring board DEsign system), a printed wiring board designing system for analog circuits. The main point is that placement and routing are based on graph-planarization and rectangular-dualization.

The motivation of this research is as follows. Designing printed wiring boards for digital circuits has been well investigated, while for analog circuits it seems that there is still much room to be developed. There may be various reasons why automatically designing analog circuits faces more difficulty than in the design of digital circuits. One of the main reasons is that functionality of circuits is sensitively affected by layouts. Layout designers have to consider carefully how placement and routing affect circuit functionality. Although a knowledge-based system may manipulate almost all conditions required in designing analog circuits, it seems too slow to be practical. It is a much faster method that is required in practical designing processes.

There are so many constraints to be considered in designing boards for analog circuits, and some of them are hard to be handled algorithmically. We select some of main constraints to be handled by PRIDE and leave others to decision done by designers. PRIDE utilizes designers' knowledge interactively so that desired printed wiring boards can be produced in reasonable computation time. This requires that each automatically executable part (or each algorithmic process) in PRIDE to output optimum or near-optimum subsolutions as soon as possible. Although multi-layered (two to four layered) boards are currently available, we consider here single-layered ones. This is because placements and routing on multilayers are based on those on single layers and because devising good algorithms to single-layered cases may be very helpful in designing multi-layered ones.

The designing flow is almost the same as that of VLSI. Each steps, however, has some differences from the counterpart in designing VLSI: they appear in (1) graph modeling, (2) finding planar subgraphs, (3) placement and (4) routing.

(1) **Graph modeling.** Since sizes of parts are fixed in an analog circuit, a graph model is required to handle explicitly not only a part but also its terminals, where they are to be attached to the part in a specified order.

(2) **Finding spanning planar subgraphs.** Each of many parts has a specified side to be faced to the board in actual mounting. These physical conditions have to be handled in finding a spanning planar subgraph of a given graph model. There have been existing some algorithms for finding a spanning planar subgraph with maximal or almost maximal number of edges [4, 5, 11, 14, 19, 20]. Unfortunately they are unlikely to be useful in the practical design processes due to lack of such preservability. This motivate us to provide a new algorithm PLAN-PWB in [2] for finding a planar spanning subgraph of a given graph G and for handling such parts.

(3) **Placement.** Connection requirements represented by this planar subgraph are be embedded without violating physical conditions or increasing total wire length. In order to consider a placement together with connection requirements, this planar subgraph is rectangular-dualized by adding some new edges and vertices so that the resulting graph may have a rectangular-dual, producing a desired placement.

A rectangular-dual R(G) of a connected plane graph G corresponds to a dissection of a whole rectangle into a set of subrectangles each of which represents a vertex of G, and two subrectangles adjoin if and only if corresponding two vertices are adjacent in G. Rectangular duals have been used for floor-planning in VLSI design [10, 12, 15, 23, 24], and their graph-theoretical properties and related algorithms are investigated in [3, 13, 21, 22, 25]. Making two rectangles adjacent is done by adding edges between corresponding vertices in G, and conversely placing two subrectangles apart can be realized by creating a path of appropriate length between corresponding vertices of G (addition of new vertices or edges may be required). Rectangles are given for parts and their terminals, and they are called part-rectangles and terminal-ones, respectively. Each of length and width of a part-rectangle has to be layer that those of the corresponding part, and a terminal-one must be wider than the width of the connecting wire. Similarly we can provide space for wires as subrectangles (called wire-rectangles) that are wide enough for all required ones to pass through. This capability of controlling placement is very useful to our purpose. Actual sizes of subrectangles are determined by means of a linear programming with constraints on sizes of rectangles. This means that a kind of compaction is done in this stage: if another placement is required then we change

(i) a rectangular dual R(G),

(ii) selection of corner rectangles (rectangles to be placed at corners of a board), or

(iii) constraints on length and width of rectangles (corresponding to 90° rotations), followed by execution of a linear programming

(4) **Routing.** The detailed routing is separated into two stages: interconnection among terminal-rectangles (a rectangle representing a terminal) of distinct parts, and inside each part-rectangle(a rectangle representing a part). The former is rather easy because wire-rectangles (rectangles for wires to pass through) are provided and they are placed adjacent to terminal-rectangles to be connected. The latter may require repeating a linear programming: if any mounting of an actual part into the corresponding part-rectangle fails to obtain a desired wiring then this part-rectangle has to be enlarged. This is done by changing constraints concerning sizes of the rectangle of a linear programming to be repeated.

2. Physical Conditions and Graph Models

2.1. Physical conditions

Placement and routing are considered under the following conditions:

C1. There are two kinds of parts; *up-sided* ones (each having a specified side which has to be faced to the board in actual mounting) and *free* ones (otherwise).

C2. parts are to be placed on one side of the board and up-sided parts should be placed as specified.

C3. Non-jumpers are assigned on one side of the board and any two of them can cross each other only at specified terminals.

C4. Routing through *part areas* (an area on the board to be occupied by one part) is prohibited. (We add this condition in order to simplify the discussion. There exist some parts which allow such routing. However the problem of finding a routing allowing passage through parts areas without violating capacity constraints has been shown NP-complete in [17]. Hence solving both problems simultaneously is too hard, and we handle such one by means of post-processing.)

C5. Each terminal is represented as a rectangle in placement, where overlapping of two rectangles are prohibited.

2.2. Graph models

Jumpers are determined by finding a spanning planar subgraph of the following graph G(C). Given a circuit C represented by a set of net lists (see Fig. 1), we construct a graph G(C)=(V,E) as follows, where $E=E_1 \cup E_2 \cup E_3$. A set of terminals requiring electrical connection among them is called a *net,* where distinct nets have no electrical contact.

1° Place one vertex for each terminal. Connection requirements among terminals in each net are represented by simple edges, and there may be some other edges determined by the graph model mentioned later.

2° Represent each of two-terminal parts and free parts as a wheel: the terminals in such a part are connected as a cycle having a new vertex inside which is adjacent to all other vertices. (In a single-layered board, we may assume that any part with at least three terminals is an up-sided one and, therefore, free parts are excluded in Section 4.)

3° Represent each of up-sided parts as a clockwise-directed cycle containing all the terminals, which are placed in the same order as we see them clockwise from above.

4° Let V be the set of vertices introduced in these steps. Let E_1, E_2 or E_3 denote the set of edges introduced in 1°, 2° or 3°, respectively, where every edge of E_3 is directed and no edges in $E_2 \cup E_3$ can be removed (they are called non-removable ones).

PRIDE adopts a graph modelling called the *star-shaped spanning tree modelling*: for each net having at least three terminals, add a new vertex and connect this vertex to each of these terminals. In this graph model, nonplanar edges to be found are actual jumpers. (See Fig. 2. For example, the vertex 56 is an inserted vertex for the multi-terminal net {46, 14, 12, 16, 18, 24}.)

3. Finding a Planar Spanning Subgraph and Jumpers

Jumpers are determined by finding a planar spanning subgraph of G(C). This is done by means of PQR-trees. The subject of this section is stated graph-theoretically as follows:

"Find a spanning planar subgraph M(C)=(V,E') of G(C) satisfying (a) $E' \subseteq E_1$, and (b) there exists a plane embedding M'(C) of M(C) such that every directed cycle is drawn clockwise without edges existing inside."

In a graph constructed from the net lists of Fig. 1, directed cycles are used to force up-sided parts to be placed as specified. Maintaining clockwise directedness is done by R-nodes of PQR-trees to be introduced. Routing through part-areas are prohibited by the introduction of wheels and by avoiding edges inside directed cycles: the latter is handled during the reduction process of PQR-trees. G(C) is denoted by G for simplicity.

3.1. PQR-trees

A PQR-tree, introduced in [16], is a directed ordered rooted tree consisting of four kinds of nodes: P-nodes, Q-nodes, R-nodes and leaves. It is a variation of the well known PQ-tree of [4]. All nodes except R-nodes are elements of PQ-trees. Two PQR-trees T and T' are *equivalent* (denoted by $T \equiv T'$) if and only if T' is obtained from T by repeating any one the following two transformations: (i) changing the order of children of a P-node arbitrarily, and (ii) reversing the order of children of a Q-node. Note that the order of children of any R-node cannot be changed. Let F(T) denote a sequence defined by concatenating leaves of a PQR-tree T from left to right. F(T) is called a *frontier* of T and represents a permutation. Let $\mathrm{con}(T)=\{F(T') | T' \equiv T\}$. A set S consisting of (not necessarily all) leaves of T is called a *leaf set* of T. Given a certain leaf set S of T, a *reduction* of T for S is a procedure to construct a PQR-tree T' such that $T' \equiv T$ and all elements of S appear in F(T') consecutively.

If no reduction is possible then the current subgraph is nonplanar, and we search the present PQR-tree for a "minimum" set of edges whose deletion recover planarity. Our searching method is based on the one proposed by Ozawa and Takahashi in [19,20] for PQ-trees, and some adaptations are incorporated in order to handle R-nodes. A reduction is done by one of template matchings similarly to those for PQ-trees.

3.2. An algorithm PLAN-PWB

A new algorithm PLAN-PWB for finding a spanning planar subgraph of a given graph was proposed in [2] and its refinement was reported in [7, 8, 9].

The main point of PLAN-PWB is that we can determine in $O(|V|+|E|)$ time a "minimum" set of nonplanar edges (jumpers) whenever the reduction of the PQR-tree becomes impossible. This procedure will be repeated at most $O(|V|)$ times and, therefore, a set of jumpers can be obtained in $O(|V||E|)$ time by the union of such "minimum" sets. Although minimality of a solution by PLAN-PWB is not guaranteed theoretically, we can expect it in almost cases, because the experimental evaluation given in [20] for planarization by PQ-trees shows that non-minimality rarely exists and that optimum solution are found in more than 75% cases. On the other hand [16] proposed an $O(|V|)$ algorithm for a planarity testing by means of PQR-trees. A minimal set of jumpers can be determined by repeating addition of an edge followed by this planarity testing. We call this algorithm REPEAT-PLAN, and its time complexity is also $O(|V||E|)$. Experimental evaluation through practical data (audio circuits) shows many cases where PLAN_PWB produces smaller solutions than REPEAT-PLAN and the former computation time is much less than the latter [2,7,8]. Recently [18] proposed an $O(|v|^2)$ algorithm for finding a minimal set of nonplanar edges. Its capability, however, is unknown, since no experimental or theoretical evaluation is given: minimality dose not imply a good approximate solution.

In this section we assume that G is a 2-connected graph with a vertex set $V=\{1, \cdots, n\}$, and outline PLAN-PWB based on [7,8].

```
<PLAN-PWB>
Step 1. Apply the st-numbering algorithm [6] to G.
  For simplicity, we consider the vertex $i \in V$ is the i-
  th st-numbered one. Construct a PQR-tree T
  consisting of only the vertex 1. $N \leftarrow 1$.
Step 2. (1)Construct a PQR-tree $T_N$ for the vertex N,
  where directed edges are handled carefully to
  avoid edges to be placed inside directed cycles.
  {Leaves of $T_N$ correspond to vertices adjacent to N
  in G.}
  (2)Delete all copies of the vertex N appearing as
  leaves of T, and add $T_N$ into T by making the root of
  $T_N$ as a child of the node to which those deleted
  copies were adjacent. Let T denote the resulting
  PQR-tree.
  (3)Assign a number $K_v$ for each node v of $T_N$ such
  that x is the parent of y if and only if $K_x < K_y$.
  (4)$N \leftarrow N+1$.
Step 3. Let S be the set of those leaves of T
  corresponding to the vertex N of G. If a reduction
  of T for S can be done then goto Step 5.
Step 4. Find a minimum set of leaves of T such that,
  after edges that are incident upon those leaves are
  deleted from T, we can resume a reduction of the
  resulting PQR-tree for S.
Step 5. In the PQR-tree obtained after one reduction
  of Step 4, coalesce all elements of S, which appears
  consecutively, into one leaf {this leaf corresponds
  to the vertex N}, and let T denote the resulting PQR-
  tree. If N=n then halt else goto Step 2.
```

Fig. 2 show a planar spanning subgraph found by PLAN-PWB for the graph of the circuit of Fig. 1. The correctness of PLAN-PWB is omitted due to shortage of space: see [2,8] for the detailed proof and the detailed explanations of Step 4.

The most time-consuming part of PLAN-PWB is the computation of the minimum number of leaves to be deleted. This part takes time proportional to the number of nodes in a PQR-tree T, and at most $O(|V|+|E|)$ nodes exist in any PQR-tree. Since there is $O(|V|)$ repetitions, time complexity of PLAN-PWB is $O(|V|(|V|+|E|))$.

3.3. Placing specified parts on the boundary of a board

There are various situations where some specific parts are required be mounted on the boundary of a printed wiring board under consideration. This constraint can easily be handled in PLAN-PWB. If we are given the ordering of terminals of parts to be placed on the boundary, then we add special edges to a graph model G(C) so that these terminals form a cycle in which they appear as specified. We first set all special edges as non-removable ones (that is, they cannot be jumpers) in finding a spanning planar subgraph, and then choose the two vertices *s*, *t* from this newly defined cycle, where *s* and *t* are the first and the last vertex of the s-t numbering. PLAN-PWB finds a spanning planar subgraph having a plane embedding in which the added cycle form the boundary of the exterior face. Clearly this embedding is a desired one. These additional steps do not increase time complexity of PLAN-PWB.

4. Rectangular-Dualization

Rectangular-dualization of a planar graph M(C) obtained in Section 3 is considered in this section. It is shown that rectangular-duals are very useful in designing printed wiring boards, especially for analog circuits, in the sense that terminals and their adjacency to corresponding parts can be handled explicitly.

4.1. Rectangular-duals

Let G be a plane graph (a plane embedding). A dissection of a rectangle into some subrectangles is called a *rectangular dual* of G if and only if it satisfies (1) and (2):

(1) there is a one-to-one correspondence between vertices of G and subrectangles of the dissection,

(2) two vertices of G are adjacent if and only if the corresponding subrectangles adjoin.

Every plane graph does not always have a rectangular-dual, and even if it exists, more than one rectangular-dual of a plane graph G may be possible. A graph G is called a *properly triangulated planar* (PTP) graph if G has a plane embedding satisfying the following[13]:

P1. Every internal face is a triangle.

P2. Any cycle that is not a boundary of any face has length≥4.

Note that any internal vertex (a vertex not on the boundary of the external face) of a PTP graph has degree≥4 [3].

A necessary and sufficient conditions for a given plane graph G to have a rectangular-dual is given in [13]. Before stating it we need some definitions. If v is a cutvertex of G then the total number of connected components after the deletion of v from G is called the *separation degree* of v. A *biconnected component* of a graph G is a maximal subgraph having no cutvertices. If G has a cutvertex then a biconnected component containing exactly one cutvertex of G is called a *pendant*. Let H be any biconnected component of a plane graph G. For a (u,v)-path P that is a subgraph of the boundary of the external face of H, an edge $(u,v) \in E(G)-E(P)$ is called a *shortcut*. If no vertex of P except u and v has an incident shortcut then P is called a *corner implying path* of H. A *critical* corner implying path of H is the one not containing a cutvertex of G. Now we state the following theorem.

Theorem 5 [13]. A connected plane graph G has a rectangular-dual if and only if the following holds.

(1) G is a PTP graph.

(2) One of the following (i) and (ii) holds:

(i) G is biconnected and has at most four corner implying paths.

(ii) G has a cutvertex, the separation degree of every cutvertex is equal to two, each pendant has at most two critical corner implying paths, and any non-pendant does not contain a critical corner implying path.

[3] proposed an $O(|V|)$ algorithm to determine if a

given planar graph G has a rectangular dual and to obtain a rectangular-dual if it exists. [22] proposed an $O(|V||R|)$ algorithm for obtaining all rectangular-duals of G, where $|R|$ is the total number of rectangular duals of G.

4.2. Rectangular-dualization of M(C)

We describe how to construct a plane graph M"(C) having a rectangular-dual from a planar graph M(C) obtained in Section 3. This is done by adding new vertices and edges. Note that addition of a vertex corresponds to inserting a subrectangle and that adding an edge forces two rectangles representing its endvertices to adjoin.

First we construct a planar graph M'(C) from M(C) by the following procedure (1). Note that any part with at least three terminals are assumed to be an up-sided one.

(1) For each up-sided part with at least three terminals, add a new vertex inside the directed cycle and connect this vertex and each vertex on the cycle by an edge.

A *triangulation* is to make every face of a planar graph a triangle by adding appropriate number of edges without violating planarity. Note that every cycle representing a set of all terminals of one part has been triangulated in M'(C). The remaining task is to construct plane graph M"(C) from M'(C) by the following procedures (2)-(6).

(2) Find all faces of M'(C) by means of any plane embedding algorithm (for example, see [5]).

(3) If there is any face of length≥ 4 then triangulate it, where we try to avoid adding an edge between terminals of two parts that should be mounted apart.

(4) If there is a cycle of length 3 that is not a face then remove it by inserting a new vertex on the cycle (and by adding some incident edges if necessary).

(5) If there are any two parts that should be mounted apart then choose one terminal from each part and add a path of appropriate length connecting the vertices that represent these terminals, where planarity has to be kept.

(6) If there is any cutvertex with the separation degree≥ 3 then add one edge (to each cutvertex) so that the separation degree may be equal to 2.

Let M"(C) denote the resulting graph. M"(C) is a PTP graph and satisfies the conditions of Theorem 5: M"(C) has a rectangular-dual. Fig. 3 shows M"(C) constracted from M(C). There are several algorithms for obtaining a rectangular-dual (or all ones if necessary) (see [3,13,15,21,22,23,24]), where directed edges can be treated as undirected ones in executing any such algorithm.

4.3. Incorporating physical conditions

The last step that we are going to do is to incorporate physical conditions, such as length and width of rectangles, into M"(C). Each rectangle represents one of the following:

(1) terminals of a part (terminal-rectangles),
(2) a virtual vertex added within a directed cycle (part-rectangles),
(3) a virtual vertex added as the inner vertex of a star-shaped spanning tree (wire-rectangles),
(4) a virtual vertex inserted at the middle of an edge on a cycle of length 3 that was not the boundary of any face (wire-rectangles if the edge represents a connection requirement; virtual-rectangles otherwise).
(5) a virtual vertex existing on those path added to make some parts to be mounted apart.

The next constraints (1)'-(6)' are put in determining actual sizes of rectangles.

(1)' If the corresponding part has exactly two (at least three, respectively) actual terminals then one of rectangle (each rectangle) satisfies lower bounds of the length and width of the part (each terminal) .

(2)' Each part-rectangle satisfies lower bounds of the length and width of this part.

(3)' The length and width of each wire-rectangle is proportional to the total sum of width of wires to be placed so that a sufficient routing area may be assured.

(4)' Any virtual-rectangle has nonnegative lower bound of the length or width.

(5)' For each edge representing a connection requirement, the shared line of the two rectangles representing the endvertices has length no less than the total width of the corresponding wire.

(6)' If there is any part-rectangle having the two terminal-ones that are adjacent to each other at its corner (see Fig. 4) then, for the length x (y, respectively) of the line shared by the part-rectangle and one of the terminal-rectangle (12). (the part-rectangle and the other (11)), either ($x\geq a$ and $y\geq a+b$) or ($x\geq a+b$ and $y\geq b$) holds, where a (b, respectively) is the total width of the wire to be connected to the former (12) (to the latter (11)). We choose one of them arbitrarily as a constraint.

We determine the length and width of all rectangles so that the sum of length and width of the whole rectangle may be minimized: this is done by repeating a linear programming with constraints (1)'-(6)'. A linear programming is also used in [15,21,23,24].

Fig. 5 shows a placement, with actual sizes of rectangles, given by a linear programming. There are at least three ways for reducing the sum of length and width of the whole rectangle.

(i) to change the choice of vertices *NW*, *NE*, *SW* and *SE* corresponding to four corner-rectangles: rectangular-duals depend upon choice of such vertices;

(ii) to change the roles of widths and lengths of some part rectangles: this is done by replacing the corresponding constraints with each other (this replacement means 90° rotation of an actual) part, and

(iii) to change the roles of part rectangles for two-terminal parts: each of such parts are represented as a pair of terminal-rectangles, one of which possesses the size of the corresponding part and plays the role of the part-rectangle. (This is done by changing the size constraints of the corresponding rectangles.)

Repeating a linear programming with these changes incorporated is very likely to produce a placement on a smaller rectangle. The placement show in Fig.4 is produced by the combination of these three methods.

4.4. Mounting parts and routing

Actual parts can be put into corresponding subrectangles determined so far, where we try to mount them so that routing among the actual terminals and corresponding terminal-rectangles can be done as much as possible within each part-rectangle. Routing is separated into two stages: interconnection among terminal-rectangles of distinct parts, and connection inside each part-rectangle. The former is rather easy because wire-rectangles are provided and they are placed adjacent to terminal-rectangles to be connected. The latter may require repeating a linear programming: if any mounting of an actual part into the corresponding part-rectangle fails to obtained a desired routing then this part-rectangle has to be enlarged. This is done by changing constraints concerning (1)' of a linear programming to be repeated.

5. Experimental Results

PRIDE has been implemented in C programming language and runs on a workstation NEC EWS-4800/30. MINOS (Ver. 5.1), a linear programming developed by Stanford University, CA, U.S.A., is used in PRIDE. Table 1 shows some of our experimental results. Fig. 1 shows an actual circuit DATA 1 and the net lists. In Fig. 5, which is a result given for DATA 1, parts and wires are denoted by halftone rectangles and solid bold lines with a unit width, respectively. Solid fine lines denote rectangles of the rectangular-dual. Small open circles denote actual terminals as well as their locations.

6. Concluding Remarks

We can conclude that one deficiency found in the previous prototype of PRIDE [1] is overcome by incorporating placement and routing through graph-planarization and rectangular-dualization. Its implementation has been finished. Some problems left for future research are as follows:

(1) Refinement of the current PRIDE. This includes incorporating capability of handling more constraints specific to analog circuit design.

(2) Handling of multi-layered designing of printed wiring boards.

7. Acknowledgments

The authors would like to thank Mr. Eiki Kida, who was the Department General Manager, Computer Aided Design Center, Audio Systems Group, SHARP Corporation, Higashi-HIroshima, 724 Japan, for his suggestion. The research of T. Watanabe is partly supported by the Grant in Aid for Scientific Research of the Ministry of Education, Science and Culture of Japan under Grant: (A) 02302047; by the Telecommunications Advancement Foundation (TAF), Tokyo, Japan; and by CSK Research Grant for Information Communication, Tokyo, Japan.

References

[1]Araki, T., Watanabe, T., Iwamoto, K., and Onaga, K. : "Printed Wiring Board Design System PRIDE for Analog Circuits", Technical Research Reports 90-DA-52-7 IPS of Japan (1990)

[2]Araki, T., Iwamoto, K., Watanabe, T., and Onaga, K. : "Finding a Minimal Set of Jumpers in the Design of Printed Wiring Boards for Analog Circuits", Technical Research Reports 90-D A-52-8 IPS of Japan (1990)

[3]Bhasker, J. and Sahni, S. : "A Linear Algorithm to Find A Rectangular Dual of a Planar Triangulated Graph", Proc. 23rd DAC, pp. 108-114.

[4]Booth,K.S. and Lueker,G.S: "Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms", J. Comput. & Syst. Sci., 13, pp.335-379

[5]Chiba, N., Nishizeki, T., Abe, S. and Ozawa, T. : "Embedding Planar Graphs Using PQ-Tree Algorithms", IEICE Trans. Vol. J67-A NO.2, pp.87-94(1984).

[6]Even, S. and Tarjan, R. E. : "Computing an st-numbering", Theoretical Computer Science, 2, pp.339-344(1976)

[7]Iwamoto, K., Araki, T., Watanabe, T. and Onaga, K. : "Minimizing Jumpers in the Design of Printed Wiring Boards for Analog Circuits by means of Maximal Planar Subgraph Extraction", 1990 Autumn National Convermtion Record, IEICE of Japan, PART6(Information and System), SD-1-1, pp. 445-446(1990-10).

[8]Iwamoto, K., Araki, T., Watanabe, T., Yasui, T., and Onaga, K.:Designing Printed Wiring Boards by Graph-Planarization and Rectangular-Dualization, Proc. 4th KARUIZAWA Workshop on Circuits and Systems, IEICE of Japan, pp81-86 (1991-04).

[9]Iwamoto, K., Watanabe, T., Araki, T., and Onaga, K. : "Finding Jumpers in Printed Wiring Board Design for Analog Circuits", Proc. 1991 IEEE ISCAS, (June 1991), pp. 2854-2857.

[10]Jabri, A., M. : "Automatic Building of Graphs Rectangularly Dualisable for Use in IC Floorplanning", Proc. ISCAS '88, pp. 1683-1686.

[11]Jayakumar, R., Thulasiraman, K., and Swamy, M.N.S. : "O(n^2) Algorithms for Graph Planarization", IEEE Trans. Computer-Aided Design, Vol. 8, No. 3, March 1989.

[12]Koike, K., Yoo, M., Tani, K., Tsukiyama, S. and Shirakawa, I. : "A Chip Floor-Plan Technique with the Use of Rectangular Dual Graph", Technical Research Report CAS85-144 IEICE of Japan pp93-100(1985)

[13]Kzominski, Krzystof A. and Edwin, K. : "Rectangular Dualization and Rectangular Dissections" , IEEE Transactions on Circuits and Systems, Vol. 15 pp. 1401-1416, November, 1988.

[14]Lempel, A., Even, S. and Cederbaum, I. : "An algorithm for planarity testing of graphs", in: Theory of graphs:International Symposium (P.Rosenstiehl, Ed.) Rome, July, 1966 pp.215-232, Gordon and breath, New york, 1967

[15]Lokanatan, B. and Kinnen, E. : "Performance Optimized Floor Planning Graph Planarization", Proc. 26th DAC, pp. 116-121.

[16]Masuda, S., Kashiwabara, T. and Fujisawa, T. : "A Wiring Problem on Single Layer Printed Circuit Board without Mounting parts Upside-Down", IEICE Trans. Vol. J66-A NO.3, pp.235-242(1983).

[17]Masuda, S., Kashiwabara, T. and Fujisawa, T. : "On Single Layer Routing under the Condition that Routing is Allowed in the Area Occupied by the part", Technical Research Report CAS82-114 IEICE of Japan pp.35- 40(1982).

[18]Matsumoto, A., Yamaguchi, K., Kashiwabara, T., Masuda, S., and Taki, M.:A Planalization algorithm in the Layout Design of Simple Layer Printed Circuit Board, Tech. Res. Rep. COMP91-21, IEICE of Japan, pp.79-88 (1991-05).

[19]Ozawa, T. and Takahashi, H. : "An algorithm for Planarization of Graphs Using a PQ-tree", Technical Research Report CAS79-150 IEICE of Japan pp25- 30(1979)

[20]Ozawa, T. and Takahashi, H. : "A graph-planarization algorithm and its application to random graphs", in Graph Theory and Algorithms, Lecture Notes in Computer Science, Vol. 108, Springer-verlag pp. 95-107, 1981

[21]Tang. H. and Chen, W. : "Generation of Rectangular Duals of a Planar Triangulated Graph by Elementary Transformations ",Proc. 1990 IEEE ISCAS (May 1990), pp. 2857-2860.

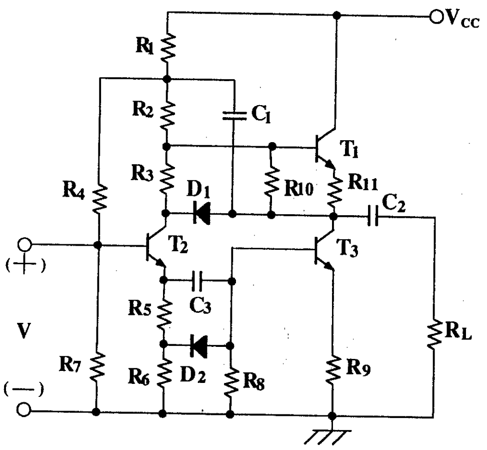
[22]Tani, K., Tsukiyama, S. and Shirakawa, I. : "An Algorithm to Enumerate All Rectangular Dual Graphs", Technical Research Report CAS86-178 IEICE of Japan pp31-38(1986)

[23]Tani, K., Tsukiyama, S. and Shirakawa, I. : "Area-Efficient Drawings of Rectangular Dual Graphs for VLSI Floor-Plan", Proceedings of the 8th Mathematical Programming Symposium, Japan(November, 1987)

[24]Tani, K., Tukiyama, S., Maruyama, S., Shirakawa, I. and

Ariyoshi, H. : "Area- Efficient Drawings of Rectangular Duals for VLSI Floor-Plan", Proc. ISCAS '88, pp. 1545-1548.

[25]Tukiyama, S., Maruyama, T., Shinoda, S. and Shirakawa, I. : "A Condition for a Maximal Planar Graph to Have a Unique Rectangular Dual and Its Application to VLSI Floor-Plan", Proc. ISCAS '89, pp. 931-934.



```
NET_LIST
R1 2 R      R2 2 R      R3 2 R      R4 2 R
   1 2         2 3         3 4         2 7
R5 2 R      R6 2 R      R7 2 R      R8 2 R
   9 11        11 13       7 13        10 13
R9 2 R      R10 2 R     R11 2 R     RL 2 R
   12 13       3 5         8 5         6 13
C1 2 C      C2 2 C      C3 2 C      D1 2 D
   2 5         5 6         9 10        4 5
D2 2 D      T1 3 T      T2 3 T      T3 3 T
   11 10       1 5 3       4 9 7       5 12 10
VCC 1 V     V 2 V
   1           7 12
```

Fig.1. An example of a circuit and its net lists(DATA 1 in Table 1), where only connection requirements are given.

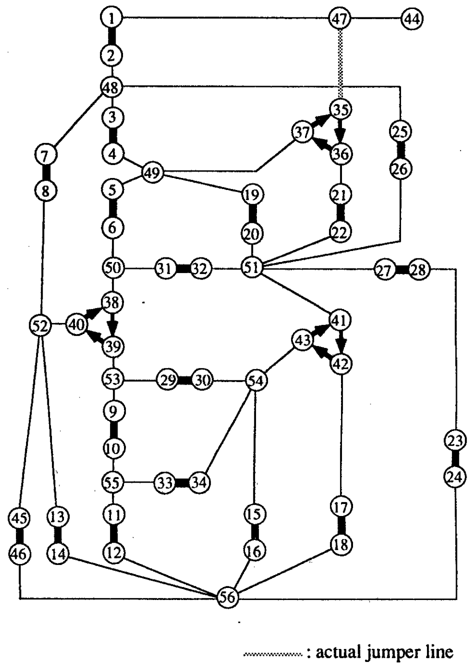


Fig.2. The graph G(C) representing DATA 1 of Table 1, where jumpers are denoted by halftone lines.

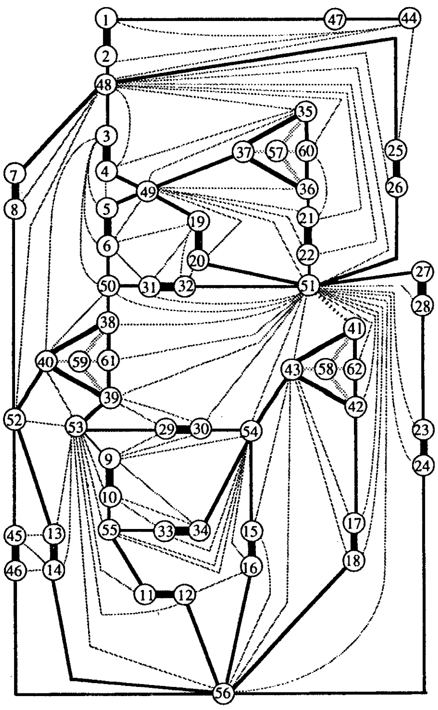


Fig.3. The graph M"(C) constructed from the planar subgraph M(C), where M(C) is given by deleting jumpers from G(C) of Fig. 2.

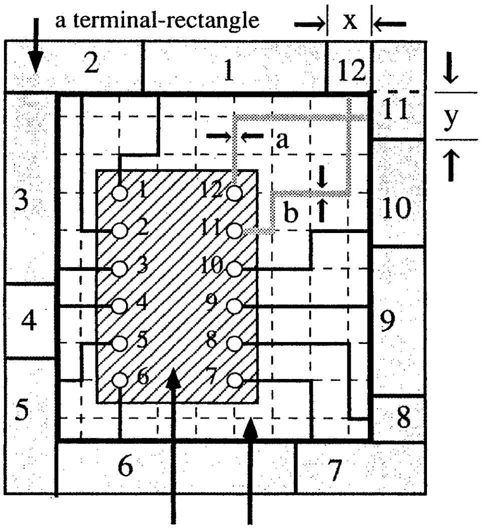


Fig.4. A part-rectangle with two terminal-ones adjacent to each other at a corner.

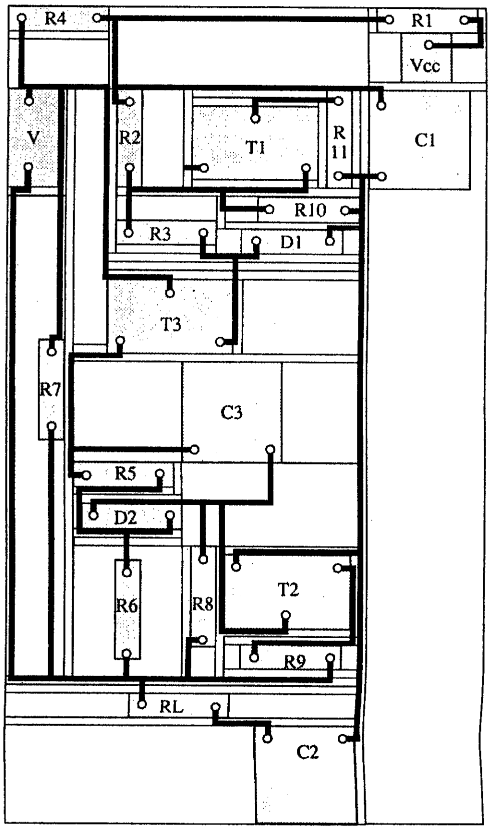


Fig.5. A placement with actual sizes of rectangles incorporated. Detailed routing is also shown, where every wire has a unit width.

Table1. Some of experimental results by PRIDE on a workstation NEC EWS 4800/30 (CPU:68030 50MHz).

Data	# Parts	# Terminals	|V|	|E|	# Jumpers	# Time (s)	Board size (mm × mm)
DATA1	22	46	56	70	1	2.699	57 × 96
DATA2	31	77	85	101	0	8.249	118 × 228
DATA3	42	100	116	140	7	12.316	134 × 278
DATA4	28	64	103	157	2	7.916	135 × 108
DATA5	24	95	94	94	2	12.399	265 × 88
DATA6	36	80	100	124	4	10.432	139 × 89
DATA7	54	115	141	173	6	19.765	168 × 104
DATA8	92	202	235	297	9	53.614	331 × 174