A METHOD FOR GRIDLESS ROUTING OF PRINTED CIRCUIT BOARDS

A. C. Finch, K. J. Mackenzie, G. J. Balsdon, G. Symonds

Racal-Redac Ltd., Newtown, Tewkesbury, Glos., England.

ABSTRACT

The introduction of fine-line technologies to printed circuit board design and manufacture has stretched most previous automatic routing algorithms to the limit. A different approach designed specifically to meet the present needs is detailed.

INTRODUCTION

A program is described which has as its objectives: "To produce an automatic printed circuit board router capable of completely routing a range of different board types. These should cover the spectrum from 50 mil gridded technology designs through to present day fine-line technology, from single layer to multi-layer, with plated through or buried vias. In particular the router should be able to route effectively on boards with surface mounted components. The final product should be compatible with available CAD systems as well as future developments".

DISCUSSION

Most commercially available automatic routers are based on algorithms that rely heavily on the concept of routing grids. Their internal data structure, or model of the printed circuit board, complements the methods used. This is exemplified by the cell structures used for the "Lee" or "flood" type routers (Ref.1). As early CAD systems had gridded basic data structures this caused few problems.

Gridded modelling becomes difficult to implement successfully as the routing becomes finer. The data size increases by a square law, giving longer set-up and access times. The selection of a suitable grid itself becomes more difficult as pad and plated through hole sizes become larger compared to the track widths. Some current implementations utilise special purpose hardware to improve run times by providing a dedicated central processor and plenty of memory to overcome the data size problem. (Ref.2). It is not however reasonable to reduce the grid size down to the order of 1 mil or less to cope with current developments in the field of printed circuit board manufacture.

Fine-line routing requires a different data structure and a different algorithm suitable for true gridless routing. A data structure for gridless routing was suggested by U. Lauther (Ref.3), and this is the basis of the data structure used by the router described here.

METHOD

Before describing the method a definition of some of the terms used is given, where these may not have their obvious meanings. This is followed by a brief description of the internal data structures, an overall method and a detailed description of the method of routing a single connection.

DEFINITIONS OF TERMS USED

EDGE
: An item which can be expanded into an expansion area. Edges include source pads, source track segments, source vias, free edges, blocked edges or via site rectangles (figs 1-3). These items are defined by an enclosing rectangle, a layer, a cost point, the cumulative cost of reaching the cost point and a direction or directions in which they should be expanded.

DIRECTION
: The direction in which an edge may be expanded. This may be any one of the directions N, S, E, W, NW, NE, SW, SE or all four of the cardinal directions.

FREE EDGE
: An edge of an expansion rectangle forming part of its perimeter that does not abut onto any other obstacle. Free edges may continue around one corner.

FIXED EDGE
: An edge of an expansion rectangle forming part of its perimeter that abuts onto a fixed obstacle.

BLOCKED EDGE
An edge of an expansion rectangle forming part of its perimeter that abuts onto a track segment or via of another net, that is onto an "unfixed" obstacle. Blocked edges may be formed by merging adjacent free and blocked edges that are individually too short to produce a long enough edge to be usable.

VIA SITE RECTANGLE
An edge of an expansion rectangle, within or partially overlapping it, which could hold a valid via.

FIXED OBSTACLES
Obstacles to routing that must not be violated as they will not be altered on subsequent passes. These include the board outline or defined routing area, pads and user defined fixed track segments and vias.

UNFIXED OBSTACLES
Obstacles to routing that can be violated if necessary as they may be altered subsequently on the same or later passes. These are track segments and vias on other nets.

COST POINT
A point associated with an edge used in the computation of the cost. It is usually the closest point on the current edge to the cost point on the previous edge.

COST
Consists of a basic cost and a notional target cost. The basic cost is a function of the distance traversed between the source point and the current cost point, and additional costs for using vias and blocked edges. The notional target cost is the lowest cost from the current cost point to any target, ignoring obstacles but including necessary vias.

NET
Consists of component pads, track segments and vias that are electrically common.

ROUTED SUBNET
Consists of a single pad, a single fixed route segment or via; or two or more of these items completely connected by route segments and vias.

TARGET ITEM
Any item in the net being routed that is not included in the current source subnet.

CONNECTION
An implied joining of two routed subnets. The router does not have as input a defined "from-to" list and routes may be found between any two items on routed subnets of the same net.

MIL
One thousandth of an inch.

FIG. 1: EXPANSION ABOUT A SOURCE PAD

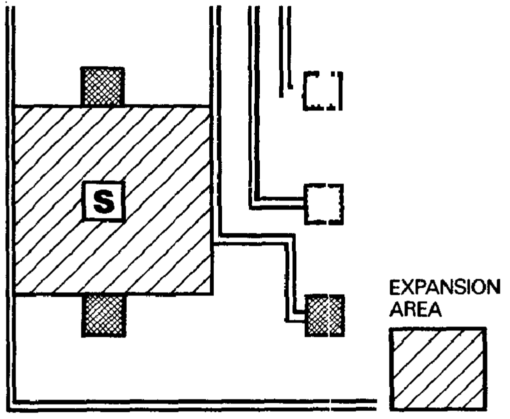


FIG. 2:FREE EDGES AND BLOCKED EDGES FROM EXPANSION IN FIG. 1

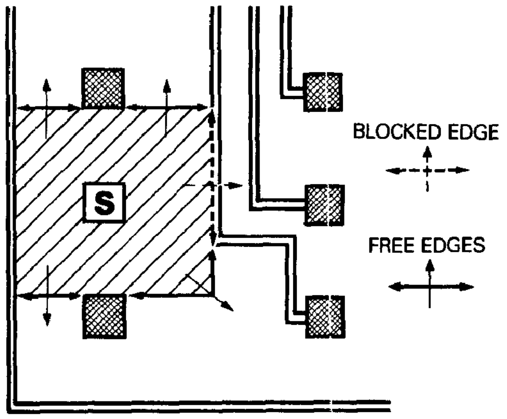
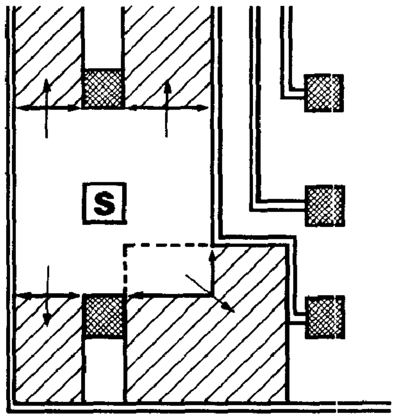


FIG. 3: EXPANSION FROM FREE EDGES IN FIG. 2



OVERALL DESCRIPTION

The router is a single or multi-pass, cost-driven, multilayer, flood and backtrack type router. Its major difference to conventional flood type routers being the fact that it has no defined cell size. Each cell, or expansion area, is as large as possible. Movement from one area to another is across an "edge" forming part of the common perimeter, or through a possible via site. Edges may also be invalid, or blocked edges, allowing for invalid routes to be found where no valid one is possible. This is utilised in the multi-pass mode. The flood part of the routing makes no detailed decisions about track paths, merely defining possible paths in terms of abutting or overlapping expansion areas. As the areas are always rectangular the router could perhaps be described as a rectangle-probe router.

SINGLE-PASS MODE

A simple fast method of testing the routability of a given placement for a particular printed circuit board design is necessary. At present the only real test is to attempt to route it. As the "place – test for routability" process is necessarily iterative, the test for routability itself needs to be effective and fast. Using a router to do this requires a fast router giving a reasonable completion rate. A simple single-pass of the connections through a router that has a multi-pass mode should give answers that are representative of the final routing. Given a high completion rate the designer has the opportunity to complete the board manually if he wishes to, or pass it to the multi-pass router to complete it automatically. Most currently available routers leave him no alternative as they have only a single-pass mode.

MULTI-PASS MODE – LEAST INVALID ROUTING

The router described here has a multi-pass mode of use. This will continue to attempt to route a board design until 100% completion is attained, then it will tidy the routing produced, reducing the total length of tracks, number of via holes and number of track segments and generally finishing with a truly manufacturable board.

There is a fundamental difference between the two modes of use that should be explained in more detail here. The single-pass mode does not route connections for which it cannot find a valid path. This is common to most automatic routers. The multi-pass router however, always finds a path for each connection on each pass; if it cannot find a valid path it will return an invalid path, that is a path which impinges upon other nets. For example the single-pass router will route 90% of the connections of a given board with 10% missing; the multi-pass router will route 85% of the connections validly and the remaining 15% with minimum crossovers or spacing violations. Hence the single-pass mode is simpler than the first routing pass of the multi-pass mode and may even produce an apparently higher completion rate. The multi-pass mode uses the result of the previous pass as the starting point for its next pass. It is believed that "least invalid routing" is an important key to improving the routing success on subsequent passes, as opposed to simply ignoring the failures.

DATA STRUCTURES

Two main internal data structures are used by this router, the obstacle data structure being the "model" of the PC board and the activity array, or "frontier" list.

Obstacle Data Structure – the obstacle data structure used is a 4-dimensional binary tree based on one proposed in Ref.3. This was described as:

"... trees are defined by the following properties:

- The tree consists of nodes and leaves.
- Each node has exactly two successors, the left son and the right son, which may be nodes (subtrees) or leaves.
- All objects the router has to deal with (wire-segments, vias, obstacles, ...) are stored as rectangles in the leaves.
- Each node contains the circumscribed rectangle of the corresponding subtree.
- Each node defines a partitioning of the corresponding subtree according to one of the four coordinate directions.
- For all rectangles R in the right subtree and for all rectangles L in the left subtree the inequalities

$$R[D] < Z$$
$$L[D] >= Z$$

hold, where D is one of the four directions and Z is the partitioning value. The coordinate Z is the same as the D component of the circumscribed rectangle in the right son and the partitioning direction D is "west" for the root of the tree, "south" for nodes on the first level and so on (repeating the directions cyclically). ..."

The structure used differs slightly from that proposed in that a single tree is used to hold obstacles on all layers, this simplifies the searching for via sites on multi-layer designs. Additionally the connectivity information is held in the tree by "ringing" all items on the same net and on separate rings all items on the same routed subnets. This is done so that all the unfixed items on a single net can easily be identified for deletion, or for marking as targets all non-source items. This structure has the advantage that its size is dependent upon the number of obstacles it holds and not the size of individual obstacles, or more particularly any defined grid size.

Activity array – this array is similar in use to the "frontier" list of a classical flood router. It contains details of each edge before and after expansion, including costs and cost points, back pointers, and pointers to expansion areas in the obstacle data structure.

OVERALL METHOD

The single-pass mode is implemented as a single pass of the router with "least invalid routing" disabled. The following method describes the multi-pass router. The overall program structure can be written as:

```
WHILE the number of invalid routes is greater than
zero:
    Order the nets

    FOR each net in turn:

        Remove any unfixed track segments and
        vias from the net and the obstacle array.

        WHILE the net is not fully routed:

            Select a routed subnet and mark remain-
            ing subnets as targets.
            Route a single connection (see next
            section).
            Add the route found to the obstacle data
            structure.

        END
    END
END
```

The router has two stages; multiple routing passes and smoothing passes. For all passes after the first the nets are ordered by decreasing total net costs on the previous pass. Routing passes terminate when all nets have been routed without conflicts, or after a user defined maximum number of passes. If the routing was completed, a user defined number of smoothing passes is done. Each pass involves a rip-up and re-route of each net in turn, whether it was routed validly or invalidly on the previous pass.

ROUTE A SINGLE CONNECTION

1. Set the "current lowest route cost" to an impossibly high value. Add each item of the source routed subnet to the activity array as edges to be expanded, with zero costs and expansion directions set to "all directions."

2. Expand the next unexpanded item in the activity array with the lowest cost. Expansion consists of moving one side of the rectangle defining the edge, in the expansion direction, until an obstacle is met. If more than one direction is implied the order in which this expansion takes place is dependent upon the layer concerned and the size of the edge rectangle. The result of an expansion is a free expansion rectangle about or away from the defined edge. This expansion rectangle is added to the obstacle array as an expansion area. If the edge was a blocked edge it is expanded by finding the free elements of its opposite side and also the end edges, where these are made wide enough, if possible, for the current track width and associated spacings. This allows a route to be found either crossing a previous route segment or following its path. These new edges are added to the activity array with costs associated with routing with conflicts.

3. The expansion area represents an area within which route segments can be placed. At this stage no decisions are made as to actual positions for route segments. Provided that the area is larger than the current track width and associated spacing in both directions, the definition of track segments can be deferred until the backtrack phase. The expansion rectangle is then examined for "escape" edges, that is edges from which it can be exited. These are free edges, target edges and blocked edges around its perimeter and target via sites and via sites within it. If vias are allowed in the design, the expansion area is examined for possible via sites that could be placed to reach target track segments, and for possible via sites to reach other layers. Via sites are found as via site rectangles which are large enough to contain legal vias plus their associated spacings. Target via site rectangles are constrained such that a via placed anywhere within them will contact the target track segment. The method used is loosely based on a method for finding "maximum empty rectangles" in a given area given by Ref.4. During smoothing passes or for single-pass routing blocked edges are treated as fixed, and hence no routing with conflicts is allowed.

4. Any target edges are considered for addition to the activity array. If the length of the edge is too short the adjacent areas are examined to determine if it is possible to reach the edge legally. The cost from the previous cost point is computed and the cumulative cost found. If this cost is less than or equal to the current lowest route cost, the item is added and the current lowest route cost updated.

5. Any free edges are next considered for addition to the activity array. For addition the cumulative cost of reaching them plus the notional cost of reaching the nearest target must be less than or equal to the current lowest route cost, and the length of the edge must be sufficient for the current track width and associated spacing.

6. Blocked edges are treated in a similar manner to free edges except that the computed cost includes cost penalties for using them. These penalties are increased if the blocked edge or part of a merged edge is on a net already routed this pass rather than on a net remaining from the previous pass.

7. Via site rectangles are added to the activity array as edges on all the layers that they can reach if the cost of reaching them, using them, plus the notional cost of reaching the nearest target, is less than or equal to the current lowest route cost.

8. If there are any edges in the activity array that have a cost less than the current lowest route cost that have not yet been expanded, route searching continues from 2 above. Otherwise, if the current lowest route cost is equal to its initial impossibly high value, no route has been found. This could happen if it is impossible to find a path because of intervening fixed obstacles. For the normal case the current lowest route cost is lower than its initial high value and a route has been found in terms of abutting or intersecting expansion rectangles, and associated escape edges.

9. A backtrack routine is then used to define the actual path from the lowest cost target reached through the expansion areas and edges. During the routing passes this backtrack process "contour follows" the perimeters of the expansion areas where possible in order to maintain the largest free spaces for future route paths. The contour following keeps the track segments and vias formed one space away from the perimeter of the expansion rectangles. During smoothing passes the backtracking is altered to attempt to continue in one direction for as long as possible to reduce the number of segments, and to aim for the centre of the expansion areas to even out the spacing between tracks. The cost parameters are also adjusted to reduce the number of vias.

COST FUNCTIONS

As the router is "cost-driven" the definition of, and the balance between, the various costs used affects the routing success very considerably. The costs currently used by the router at any given point are a complex function of:

Length in the preferred direction for each layer from the source to the current point.

Length in the non-preferred direction for each layer from the source to the current point.

Via usage.

Conflicts with other nets not yet routed this pass.

Conflict with other nets already laid down this pass.

Notional cost of reaching the nearest target.

Completion rate on the previous pass.

Pass number.

Some of the above items are relatively straightforward linear functions, others are non-linear graphical functions.

GRIDDING

In practice there are several reasons for constraining the route segments and vias produced to be on grids.

1. NC equipment, e.g. photoplotters and drilling machines, used to produce the printed circuit board may be less accurate than the printed circuit design coordinate resolution. Typically these have accuracies in the order of 0.1 to 1.0 mil.

2. Manual designers prefer a coarser grid for manipulating route segments on the graphics screen. A value of 5 mil is considered reasonable.

3. To be compatible with earlier products that have a coarse coordinate resolution, typically 25 mil.

4. With multi-layer board designs in particular a via grid is often a manufacturing requirement.

To satisfy these requirements both a route segment grid and a via grid have been incorporated into the router. These are user-definable, so that for earlier products a coarse grid can be used, for new products a much finer grid can be chosen, to meet the needs of the photoplotter only, for example. Using a coarse grid has the effect of increasing the run times and/or reducing the completion rates because it is imposing restrictions upon an essentially free system and does not have the normal advantage of reducing the data structure size requirement.

IMPLEMENTATION

A prototype progam was implemented on a VAX 11/780 computer running under UNIX, and written in "C". This version was used for early development of the method and to do initial evaluation of cost functions. Subsequent development of the production version is largely independent of the hardware and is currently available on VAX and APOLLO computers.

RESULTS

Figure 4 shows the routing of a small conventional 2-layer digital board which allows two signal tracks between D.I.L. pads. The main characteristics are:

Board area 17.25 square inches

285 connections
96 nets

20 mil grid

Component density 0.47 square inches/ equivalent 14 pin D.I.L.

Running on a VAX 11/750 the single-pass mode routed 96.8% of the connections in 13 minutes, which compared well with our own VAX-based routers.

For the multi-pass mode the results were:

Pass	% Valid Routing	No of Vias	No of Segments	Length	Time (Mins)
1	95.1	80	1050	259.8	15.0
2	100.0	93	1151	265.1	13.6
Smoothing					
1	100.0	91	925	268.8	12.8
2	100.0	83	908	268.3	11.8

Figure 5 shows part of a larger design with mainly surface-mounted components, routed on a 12.5 mil grid to 100% completion.

CONCLUSIONS

The data structure developed for this router is amenable for other purposes than the routing method outlined above. It has been used for other routers, in particular a pattern router for memory area routing. Without much alteration it is also very suitable for dimensional checking of a completed layout.

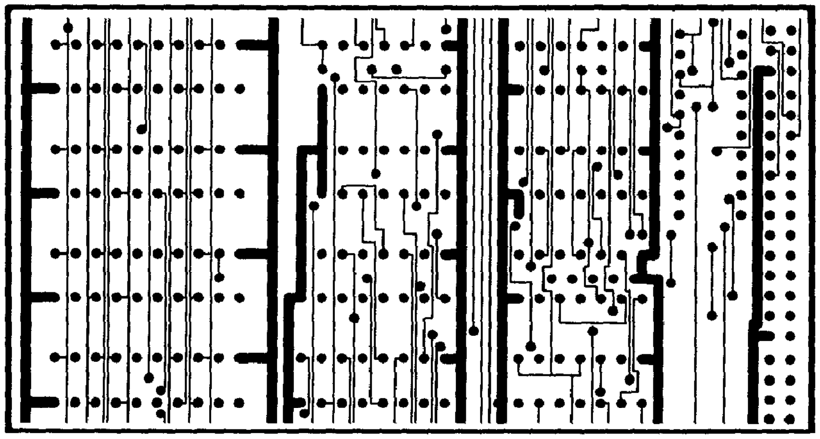
The cost functions for the current router contain many variables and further development of them is expected to pay dividends in completion rates and run times.

Some special purpose hardware is under consideration to speed up the operation of the router. This involves the use of parallel processing of the obstacle data structure and computation of target costs.

REFERENCES

1. Lee, C. "An Algorithm for Path Connections and its Applications," IRE Transactions on Electronic Computers, September 1961, pp. 346-365.
2. Damm, E. and Gethoffer, H. "Hardware Support For Automatic Routing", Proceedings of the 19th Design Automation Conference, 1982. pp. 219-223.
3. Lauther, U. "A Data Structure For Gridless Routing", Proceedings of the 17th Design Automation Conference, 1980. pp. 603-609.
4. Jayakumar, V. "A Data Structure For Interactive Placement of Rectangular Objects", Proceedings of the 17th Design Automation Conference, 1980. pp. 237-242.

October 1984



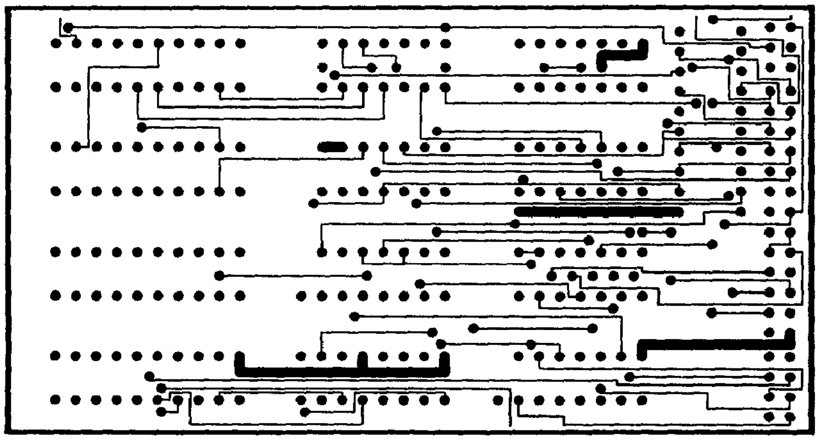


FIG. 4

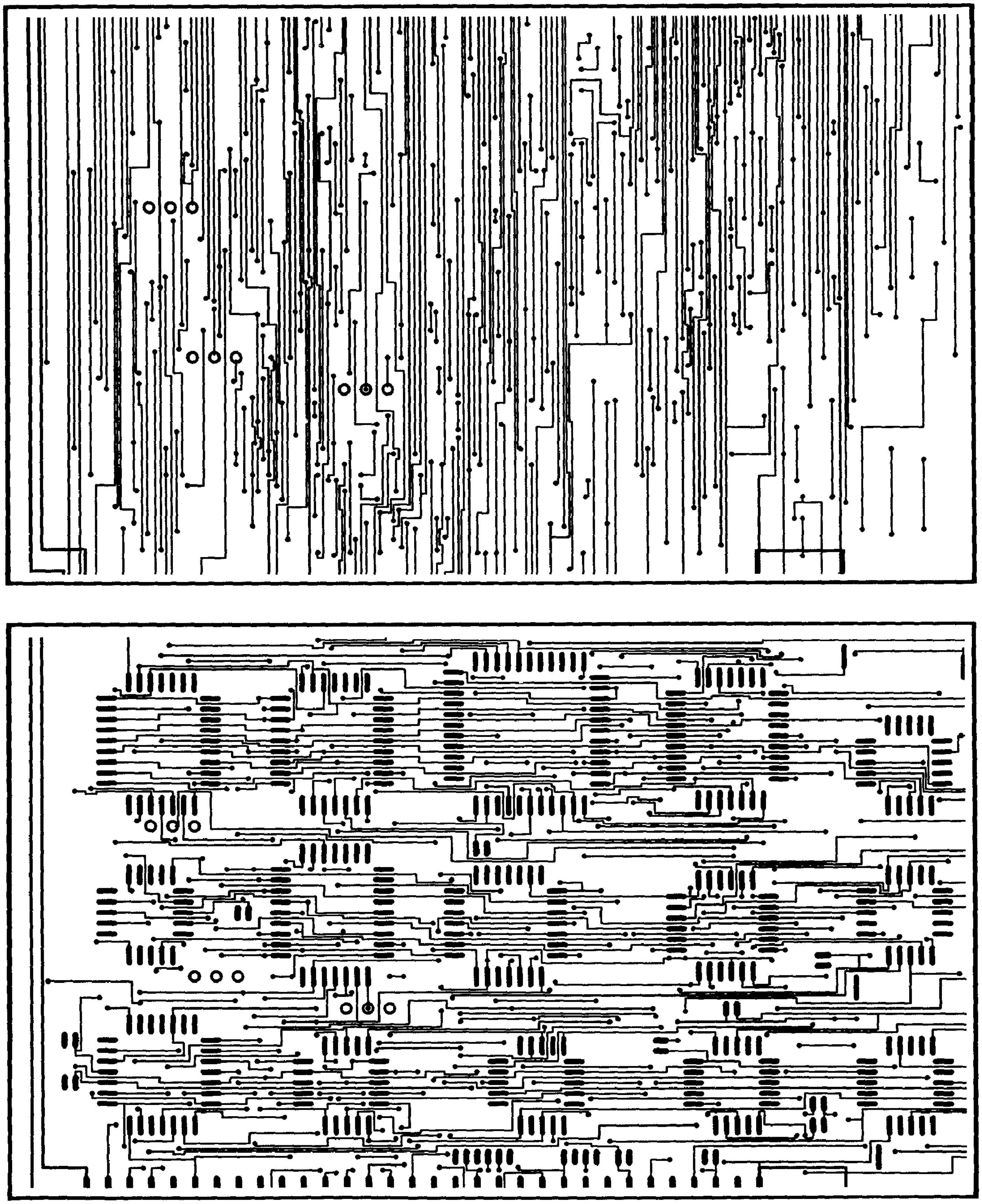


FIG. 5