A "DOGLEG" CHANNEL ROUTER

David N. Deutsch

Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

This paper presents an algorithm for interconnecting two sets of terminals across an intervening channel. It is assumed that the routing is done on two distinct levels with all horizontal paths being assigned to one level and all vertical paths to the other. Connections between the levels are made through contact windows. A single net may result in many horizontal and vertical segments. Experimental results indicate that this algorithm is very successful in routing channels that contain severe constraints. Usually, the routing is accomplished within one track of the mathematical lower bound. The routing algorithm presented here was developed as part of LTX, a computer-aided design system for integrated circuit layout and was implemented on an HP-2100 minicomputer. A typical channel (300 terminals, 100 nets) can be routed in less than 5 seconds. Routing results are presented both for polycell chips under development at Bell Laboratories and for examples that exist in the published literature. For the latter, reductions of 10% in the wiring area were typical.

1. INTRODUCTION

A basic problem in the design of large-scale integrated (LSI) circuits is to complete the necessary interconnections in as small an area as possible. Many routing algorithms have been used with varying degrees of success. Some employ general purpose two-dimensional routing strategies such as those described by Lee [1], Hightower [2] and Mattison [3]. These methods, however, often yield poor results because a net routed early may be arbitrarily placed in a location that blocks some subsequent net. Various net ordering schemes have been proposed in an attempt to minimize this problem but unfortunately, the blockages persist.

If the original routing problem is amenable, another approach is often used. Simply decompose the original routing requirements into a series of routing subproblems, each of which is a channel routing problem. Mattison [3] actually performs this decomposition even though he does not use a channel router. In descriptive terms, a channel typically consists of the space between two parallel rows of terminals. For this situation, the interconnection problem can be transformed into a one-dimensional routing problem with constraints. If the constraints are non-cyclic and the channel height is adjustable, 100% completion of the routing is always guaranteed.

Channel routers are usually easier to implement and execute much faster than the more general routers mentioned previously because the problem under consideration is inherently simpler. In addition, channel routers operate on one routing track at a time, thus minimizing the amount of data that must be core resident. Since the data storage requirements are small, channel routers can be implemented on minicomputers. Channel routing algorithms have been described by Hashimoto and Stevens [4] ("unconstrained left-edge") and by Kernighan, Schweikert and Persky [5] ("constrained left-edge with zoning" and "optimal" via branch and bound). As LSI circuit layouts become more modular, the channel routing approach tends to yield better results. In a typical design (see Figure 9), there may be rows of polycells with perhaps some additional special logic functional groups such as programmable logic arrays (PLA's) or shift registers [6]. In this paper we will present a new channel routing algorithm that generally gives better results than those mentioned previously.

Before describing the new router in detail, we will define some of the terms and describe some of the concepts that apply to channel routing in general. In this part of the discussion we will use simple examples which are not necessarily realistic in order to clarify some of the definitions and descriptions. In general, our usage will correspond to that of Reference [5].

Consider a rectangular channel with terminals along its top and bottom edges. The terminal abscissas usually lie on a grid that has a uniform spacing equal to that of adjacent terminal positions. A net consists of a set of terminals that must be interconnected via some routing path. Some nets may also have a connection point at one or both ends (left and/or right) of the channel. Such a point corresponds to a terminal that has a known abscissa (that of the specified channel end) but whose ordinate will be determined by the channel router. Let us assume that all of the horizontal routing occurs on one level while the vertical segments occur on another level. For integrated circuits, the horizontal segments are typically metal while the verticals are polysilicon and/or diffusion. In order to interconnect a horizontal and vertical segment, a contact must be placed at the intersection point. The task of a channel router is to route all of the nets successfully in the minimum possible area. Since the channel length is fixed, the area goal is equivalent to minimizing the channel height; this height is equal to the spacing between adjacent horizontal tracks (a constant imposed by the technology) times the number of such tracks required to accomplish the necessary connections plus appropriate clearances for the top and bottom tracks.

Let us now consider the constraints under which channel routers usually operate. The requirement that all horizontal segments occur on one level with all verticals on another may theoretically cost a track or more as is shown

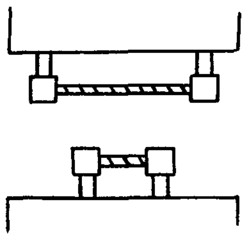
Figure 1A

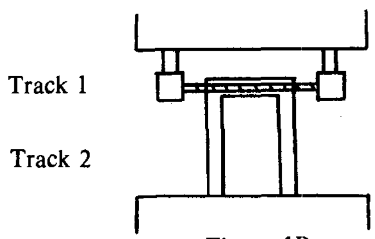

Figure 1B

in Figures 1A and 1B. Such a saving can seldom be achieved in real problems. The overlapping of nets in this manner is generally undesirable because of increased capacitive coupling between the nets. The imposition of this constraint, however, has some benefits. In order to ensure that two distinct nets are not shorted, it is only necessary to ensure that no vertical segments overlap and no horizontal segments overlap. If a top terminal and a bottom terminal have the same abscissa and they are to be connected to distinct nets, the horizontal segment of the net connected to the top terminal must be placed above the horizontal segment of the net connected to the bottom terminal; otherwise, the vertical segments would partially overlap. Since the abscissas of these vertical segments are given (the terminal positions are known and fixed), many such vertical constraints must be satisfied. It is of course possible that constraint chains of arbitrary length (A above B, B above C, etc.) as well as constraint loops (A above B at one terminal position, B above A at another is a loop of size 2) can occur.

If there is no constraint loop, all channel routing algorithms will be able to complete the required routing. The "Left-Edge" algorithm [4] always yields optimum results if there are no vertical constraints (e.g., a channel which has terminals on one side only). As was shown in Kernighan, et al. [5], it often yields suboptimal results if vertical constraints are present. Figure 2A, an example of

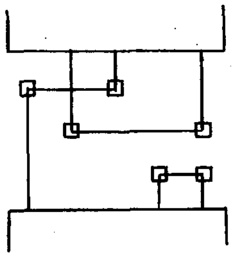
Figure 2A

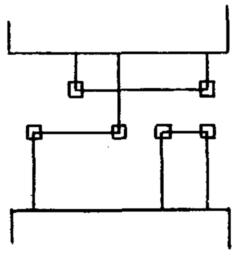
Figure 2B

such suboptimality, is taken from this reference. It is quite obvious that the routing in Figure 2B is optimal. In the "Optimal Channel Router" [5] a branch and bound algorithm is presented which guarantees an optimum result. Unfortunately, that algorithm contains the restriction that each net consists of a single horizontal segment, thus

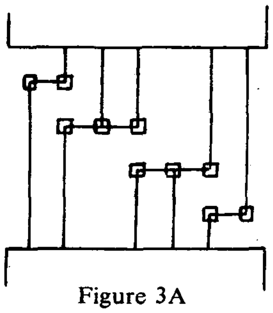
Figure 3A

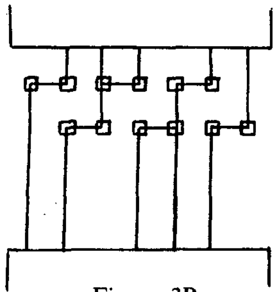
Figure 3B

Figure 3A is considered optimal. Figure 3B is much better, however. What is needed, therefore, is a suitable mathematical lower bound on the number of tracks needed to route the channel. Fortunately, such a bound exists and it is very easy to calculate. Determine the extent of each net by finding its leftmost and rightmost terminal abscissa (ignore any net for which these are equal). Then, for each terminal position in the channel, count the number of nets whose extent includes that terminal's abscissa. This is the "local density." We define the "channel density" as the maximum local density. The channel density is this mathematical lower bound. We use the term "span" to indicate the number of terminal positions where this maximum occurs. Note that both density and span are properties of the terminal positions and the connectivity data and are independent of the routing.

To illustrate the above, consider Figure 4A. The

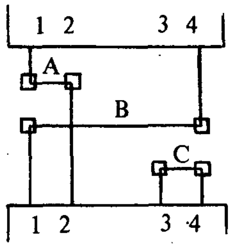

Figure 4A

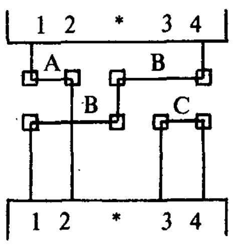

Figure 4B

constraint in column 1 requires net A to be above net B while that in column 4 requires net B to be above net C. As shown, the routing is accomplished in three horizontal tracks. Note, however, that the channel density is only two. Figure 4B shows a possible procedure whereby the nets are routed in two tracks. The key feature is the "dogleg" in net B which transfers part of the horizontal section from track 2 to track 1. Whether or not such doglegs can be introduced beneficially is very much data-dependent. In this particular example, routing in two tracks is possible if and only if there is sufficient space between columns 2 and 3 (i.e., column * exists). Otherwise, there would be no available location for the dogleg.

Now consider the problem posed in Figure 5A. There

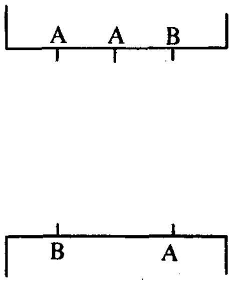

Figure 5A

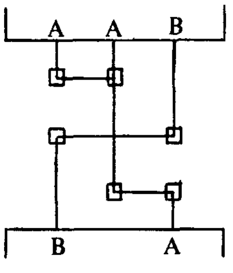

Figure 5B

are two distinct nets to be routed, but there is a constraint loop. At the left edge, net A must be above net B while at the right edge the reverse is true. This channel cannot be routed if only one horizontal segment per net is permitted. The obvious solution is a dogleg in net A as shown in Figure 5B. Note that in this case, however, the channel density is two but the routing requires three tracks.

The motivation for introducing doglegs is thus twofold. First, doglegs enable us to break long constraint chains which otherwise would prevent us from routing at or near density (e.g., Figure 3A versus 3B). Secondly, they may permit us to route a channel that otherwise would be unroutable. The former is much more important, however, since routability can almost always be obtained by some minor adjustment in the polycell ordering prior to routing. The addition of doglegs has certain disadvantages, however. From an electrical standpoint, each dogleg adds one or two additional contacts and this increases the capacitance. From a mathematical standpoint, if a dogleg occurs at a position where the local density is already equal to the channel density, the required number of horizontal segments increases by one and the subsequent routing cannot be completed in channel density (Figure 5B is an example of this situation). This is one reason why it is worthwhile to minimize the span (number of terminal columns with local density equal to the channel density) since the likelihood of this event is very dependent on the span. It is thus desirable to add as few doglegs as possible. Any routing style that extends a net beyond its original endpoints or permits its horizontal sections on different tracks to overlap [6] can only increase the apparent local density. While sometimes beneficial, the incidence of such situations did not seem to justify the coding effort required to exclude detrimental cases.

In Section 2, we will describe the new dogleg routing algorithm for a rectangular channel with all terminals located on a grid (the typical case). Section 3 will show how the algorithm described in Section 2 can be trivially modified to handle the situations where terminals are not on a grid, where there are tolerance problems for adjacent contacts on the same track, etc. These modifications are included in our implementation of the dogleg router. The situation where some tracks are partially blocked (e.g., some cells extend into the channel) is also easily handled. Section 3 concludes with a formal description of the algorithm. Section 4 presents some experimental results and compares them with the results of other routing strategies.

2. *THE DOGLEG ALGORITHM - SIMPLIFIED*

In this section we will describe the new dogleg channel routing algorithm as simply as possible. We shall purposely defer some details that, necessary as they are, might confuse or complicate this description. For example, we assume here that the channel is a rectangle and that all terminals are on grid, although neither of these assumptions is actually necessary. Consideration of these details, along with a formal algorithm description, are given in Section 3. We will also present the motivation for various decisions that were crucial during the development of this algorithm. The primary goal is to be able to route the typical channel using as few tracks as possible, and hopefully equal to the channel density. Since doglegs increase the apparent local density and the corresponding added contacts increase the capacitance (generally an undesirable result), the number of doglegs should be kept as small as possible, subject to the goal of track minimization.

By studying the output of non-dogleged routers, it was found that when they required many tracks in excess of the channel density, the usual cause was quite evident. There was a long constraint chain with a few crucial nets (typically clock lines) that were heavily connected to both sides of the channel. If these multi-terminal nets could be dogleged efficiently, a far more compact routing would be achieved. Based upon this observation, it was decided that doglegs would only be introduced at a terminal position for the net. This decision severely constrains the potential number of doglegs in that no 2-terminal net is ever dogleged, 3-terminal nets have only one potential site for doglegs, etc. Note that this rule is independent of net length (distance between leftmost and rightmost terminals). The number of added contacts is also minimized by introducing doglegs only at a position where the net already has a terminal (one added contact per dogleg as shown in Figure 3B instead of two as shown in Figure 4B).

An obvious procedure, therefore, is to order the terminals within a net based upon their abscissas, and decompose the original net into a series of 2-terminal subnets such that the n'th subnet consists of terminals n and n+1. In this decomposition, each interior terminal becomes the end of two subnets. Apply this procedure to all nets (a 2-terminal net yields a single subnet) and then use the constrained left-edge algorithm on the subnets with the following modification. When a subnet ends, the next subnet of the same net (if any) can be placed in the same track (i.e., they can overlap at the common terminal). Subnets from different nets are not permitted to overlap. This procedure works reasonably well but it often adds many more doglegs than are necessary. In order to minimize this undesirable result, a parameter called the "range" was introduced. It represents the minimum number of consecutive subnets that must be assigned to the current track. The above description thus corresponds to a range of one. As the range gets larger, fewer doglegs tend to be introduced. The only additional rule is that a sequence of subnets less than the range will also be accepted if and only if there is no unplaced subnet for this net that begins where the sequence ends (e.g., the actual net end is reached). Without such a rule, 2-terminal nets would never get placed. Our implementation of this routing algorithm permits a range from 1 to 9 for doglegs as well as the value N to indicate no doglegs.

Another feature, not related to doglegs, was included in this router. Rather than route all tracks starting from the top-left, a more symmetric choice of tracks was made. This is possible because each track is equivalent to the first track for some reduced problem. Therefore, we alternate between top and bottom tracks except for the situation where all terminals are on one side of the channel. This alternation between top and bottom usually produces routing with a smaller total vertical length (summed over all nets) than the traditional one-sided approach. This procedure tends to reduce the average capacitance per net. In addition, for two adjacent tracks (both on top or both on bottom), if one starts from the left end of the channel, the other starts from the right. If we consider the example of Figure 2, starting with top-left we arrive at 2A. Any other start (bottom-left, top-right, or bottom-right) yields 2B. Since the first track can be any one of four possibilities and the second any one of two (opposite side of channel but can start at either end), there are eight possible sequences. The routing sequence is thus another parameter available to the user.

Since we have a two-parameter procedure with ten ranges and eight routing sequences, it would be quite natural for the program to try various combinations automatically. If the user specifies a particular range and sequence, the algorithm is employed for those values only. The user has, however, the option to try all ten ranges for a particular routing sequence, all eight routing sequences for a particular range, or all eighty possible combinations of range and routing sequence automatically. The program, of course, saves the routing results and the parameter values that were best, based upon the smallest number of tracks first and the smallest number of doglegs (added contacts) second. A more formal description of the dogleg algorithm is given at the end of the next section.

3. *THE GENERAL DOGLEG ALGORITHM*

Sometimes the terminals in a channel may not all be on grid. This situation can occur, for example, if polycells with one terminal spacing share a channel with PLA's or pads with a different terminal spacing. For certain technologies, there is often the restriction that adjacent terminals cannot have contacts on the same routing track unless they are connected to the same net. A constraint similar to this must always be satisfied when terminals on opposite sides of the channel are not on grid. We will, therefore, introduce a few additional definitions and examples before concluding with a formal description of the general algorithm. Since all terminals and paths have nonzero widths, the following discussion assumes the use of centerlines to represent the terminal and path positions.

Let us first define the minimal terminal spacing to be the minimum distance between two adjacent terminals on the same side of the channel. Consider the two vertical paths emanating from such a pair of terminals. In general, the routing would be some variant of that shown in Figure

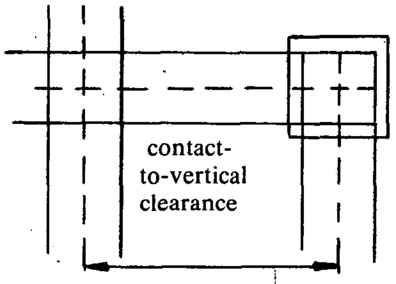

Figure 6

6. Obviously, there is some minimum required distance between the contact and the adjacent vertical segment based upon the technology being used. We call this distance the contact-to-vertical clearance. (If this minimum clearance was greater than the minimal terminal spacing, routing in the style considered here would be impossible.)

Another important clearance parameter is the minimum spacing between adjacent contacts on the same track as

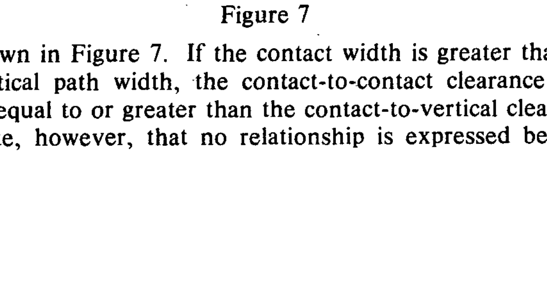

Figure 7

shown in Figure 7. If the contact width is greater than the vertical path width, the contact-to-contact clearance must be equal to or greater than the contact-to-vertical clearance. Note, however, that no relationship is expressed between the terminal spacing and the contact-to-contact clearance. The former can, in fact, be greater than, equal to, or less than the latter.

If the terminals are all on grid, the vertical constraint relationship as described previously is quite simple. At any terminal position, a vertical constraint exists if and only if both top and bottom terminals are connected to different nets. In the more general case where terminals are not on grid, a terminal on one side of the channel can generate a constraint relationship with two adjacent terminals on the

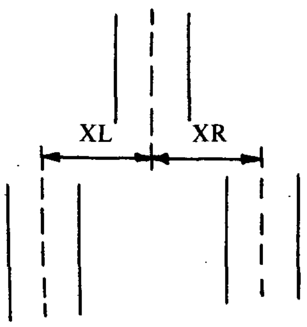

Figure 8

opposite side. As illustrated in Figure 8, this situation arises whenever both XL and XR are less than the contact-to-vertical clearance. If we consider the hypothetical but extreme case where the terminal spacing is uniform on both sides of the channel but the terminals are misaligned by half this spacing, the number of vertical constraints would, on the average, be double that of the normally aligned case. These additional constraints would be expected to adversely affect the routing, making it highly desirable to have the terminals "on grid." The routing algorithm described here, however, does not require it.

A minor adjustment is necessary in order to compute the channel density precisely for the case of off-grid terminals and for the case where the contact-to-contact clearance is greater than the terminal spacing. As before, simply apply the left-edge algorithm ignoring the vertical constraints except that each net must first be extended by the contact-to-contact clearance. It does not matter how this extension is accomplished so long as it is done consistently for all nets (e.g., add this amount to the right end of all nets). A vertical net (left end abscissa equal to right end abscissa) should be ignored entirely since it has no horizontal section and no contacts.

In the following algorithm description, only a single track is actually being routed. When that track is finished, the nets or net segments that are still unplaced constitute a reduced problem. The algorithm terminates when the track being routed remains empty. If there are no unplaced nets, the routing is complete. If, however, some unplaced nets remain, this is proof that one or more constraint loops exist. It is usually a simple matter to break such loops by means of cell reflections or exchanges.

We will assume that the netlist data has been formatted such that for each net the terminals are ordered by ascending abscissas with the channel side (top or bottom) indicated. There may be many nets whose left and/or right end is the respective channel edge. By sorting and cross-referencing various data lists before starting the routing calculations, many of the searches described below become unnecessary or trivial.

THE GENERAL DOGLEG ALGORITHM

1. Initialize the channel side (top or bottom), the starting abscissa (left or right edge of channel) and the routing direction (right, if we start from the left edge, and vice versa) based upon the type of track being routed.

2. Search for a net terminal equal to the starting abscissa or closest to it when going in the routing direction. This is the starting terminal. If none, the track is done. If this terminal is the actual start of a net (not an interior terminal), set the start flag. Otherwise, clear the start flag.

3. If the terminal is on the opposite side of the channel from the routing track, a search must be made to see if it is constrained by some other as yet unplaced net (i.e., is there some terminal on the routing side within the contact-to-vertical clearance that is not yet routed?). If so, the net is blocked. Go to step "2" ignoring this starting terminal. If the terminal is not blocked, clear the end flag.

4. Determine the number of consecutive additional terminals that are not blocked (as in step "3") for this net. The last of these is the ending terminal. If the end of the net in the direction of routing is the ending terminal, set the end flag.

5. If non-dogleg routing is requested, test if both start flag and end flag are set. If not, go to step "2" ignoring this starting terminal.

6. If the number of consecutive additional terminals is less than the range and the end flag is not set, go to step "2" ignoring this starting terminal.

7. Assign the net from the starting terminal to the ending terminal to this track. Set the starting abscissa equal to the ending terminal abscissa modified by the contact-to-contact clearance in the given routing direction.

8. If the start flag is set, delete the starting terminal from the unplaced net list. If the end flag is set, delete the ending terminal from the unplaced net list. Delete all interior terminals (if any) in this net between the starting and ending terminals. If some middle part of a net is thus deleted, the remaining end pieces constitute two distinct nets for the remainder of the algorithm. Go to step "2".

The case of a non-rectangular channel is relatively easy to handle. Typically, this situation occurs when some cell(s) protrudes into the routing channel. Instead of having a routing track which extends for the entire channel length, the track consists of two or more distinct segments. For each such segment, employ the above algorithm except that in step "1" the initialization should be to the appropriate segment end and in steps "2-4" the terminals must be tested to ensure that they lie within the length of the track segment. A terminal that does not lie within this range is automatically blocked.

4. RESULTS

4.1 BTL Usage.

The dogleg channel routing algorithm has been programmed in assembly language for the HP-2100 minicomputer as part of LTX [7], a computer-aided design system for integrated circuit layout. This program has been used to route polycell chips under development at Bell Laboratories as well as examples that exist in the published literature. The number of tracks needed to route a typical channel with no constraint loops is within three of the channel density. For new layouts, the density and span are first minimized as much as possible via cell reflections, exchanges and insertions. Constraint loops are then eliminated using the same type of cell moves while attempting to maintain the minimal density and span. Finally, a few different cell orderings, all of which have the same density and span (the generating program makes neutral moves which are random with respect to the router) are routed using a variety of routing parameters. The goal, of course, is to route the channel in as few tracks above density as possible. Using this procedure, the number of routing tracks required is usually only 5% above the channel density.

Example A: BTL Layout.

A complete polycell layout for an integrated circuit is shown in Figure 9. Table 1 gives a summary of the routing results for this chip. Three channels required one track above density while a fourth required two tracks over density. This is the only channel we have encountered which could not be routed within one track of density using the procedure described above. The entire chip was routed in approximately 35 seconds.

Example B: A 'Difficult' Channel.

Figure 10 shows a channel with a density of 19 that is routed in 21 tracks using the dogleg router. An attempt was made to find the optimal routing without doglegs using the program of Reference [5]. The program was aborted after four hours (the branch and bound algorithm did not finish) with the best routing obtained during these computations requiring 26 tracks.

4.2 Comparison with Published Results.

To compare the dogleg algorithm results with those of other routers, we have attempted to reroute published routing examples. There may be some minor discrepencies between our results and the given examples, but these are due entirely to approximations made during the data conversion and should not affect the dogleg router's comparative performance in any significant way.

Example C: GTE Layout.

Consider Example B from Reference [3] (their Figure 10). For this layout with a total density of 52, the dogleg router required 53 tracks. Note also that two of the channels actually had constraint loops which were eliminated by doglegs. Our results are shown in Figure 11 and are summarized in Table 2. The previously published routing [3] required 64 tracks, but, because of a constraint prohibiting vertically adjacent contacts, the required track spacing is reduced by about 10% for typical contact and path sizes. The dogleg router thus decreased the routing area by approximately 8%.

Example D: Hitachi Layout.

Preliminary work on a layout produced at Hitachi, Ltd. (Figure 6 of Reference [8]) also indicates a reduction in wiring area using the dogleg channel router. Their routing strategy tends to minimize the amount of vertical routing at the expense of horizontal routing while ours does the opposite. Hitachi's layout required 132 horizontal tracks and 40 vertical tracks while the dogleg router required 107 and 45 tracks respectively. The routing area was thus reduced by approximately 15% while the total chip area decreased by about 11%.

REFERENCES

1. Lee, C.Y., "An Algorithm for Path Connections and Its Applications," IRE Transactions on Electronic Computers, pp. 346-365, September 1961.

2. Hightower, D.W., "A Solution to Line-Routing Problems on the Continuous Plane," Proc. 6th Design Automation Workshop, pp. 1-24 (1969).

3. Mattison, R.L., "A High Quality, Low Cost Router for MOS/LSI," Proc. 9th Design Automation Workshop, pp. 94-103 (1972).

4. Hashimoto, A. and Stevens, J., "Wire Routing by Optimizing Channel Assignment within Large Apertures," Proc. 8th Design Automation Workshop, pp. 155-169 (1971).

5. Kernighan, B.W., Schweikert, D.G. and Persky, G., "An Optimum Channel-Routing Algorithm for Polycell Layouts of Integrated Circuits," Proc. 10th Design Automation Workshop, pp. 50-59 (1973).

6. Kozawa, T., Horino, H., Ishiga, T., Sakemi, J. and Sato, S., "Advanced LILAC - An Automated Layout Generation system for MOS/LSIs," Proc. 11th Design Automation Workshop, pp. 26-46 (1974).

7. Persky, G., Deutsch, D.N. and Schweikert, D.G., "LTX - A System for the Directed Automatic Design of LSI Circuits," Proc. 13th Design Automation Conference (1976).

8. Kozawa, T., Horino, H., Watanabe, K., Nagata, M. and Hukuda, H., "Block and Track Method for Automated Layout Generation of MOS-LSI Arrays," IEEE International Solid-State Circuits Conference, Digest of Technical Papers, Vol. XV, IEEE Cat. No. 72C3-ISSCC, pp. 62-3 and pp. 214-215 (1972).

Table 1. Example A.

Channel	Density	Tracks
D1	1	1
D2	16	17
D3	4	4
D4	14	16
D5	3	3
D6	16	17
D7	3	3
D8	14	15
D9	2	2
D10	17	17
D11	1	1
Total	91	96

Total number of doglegs: 195

Total number of contacts: 1756

Table 2. Example C.

Channel	Density	Tracks Ref. [3]	Tracks Dogleg
D1	18	23	19
D2	1	1	1
D3	15	19	15
D4	1	1	1
D5	17	20	17
Total	52	64	53

Total number of doglegs: 41

Total number of contacts: 595

Note that both channel D1 and channel D5 have a "constraint loop" of size 3.

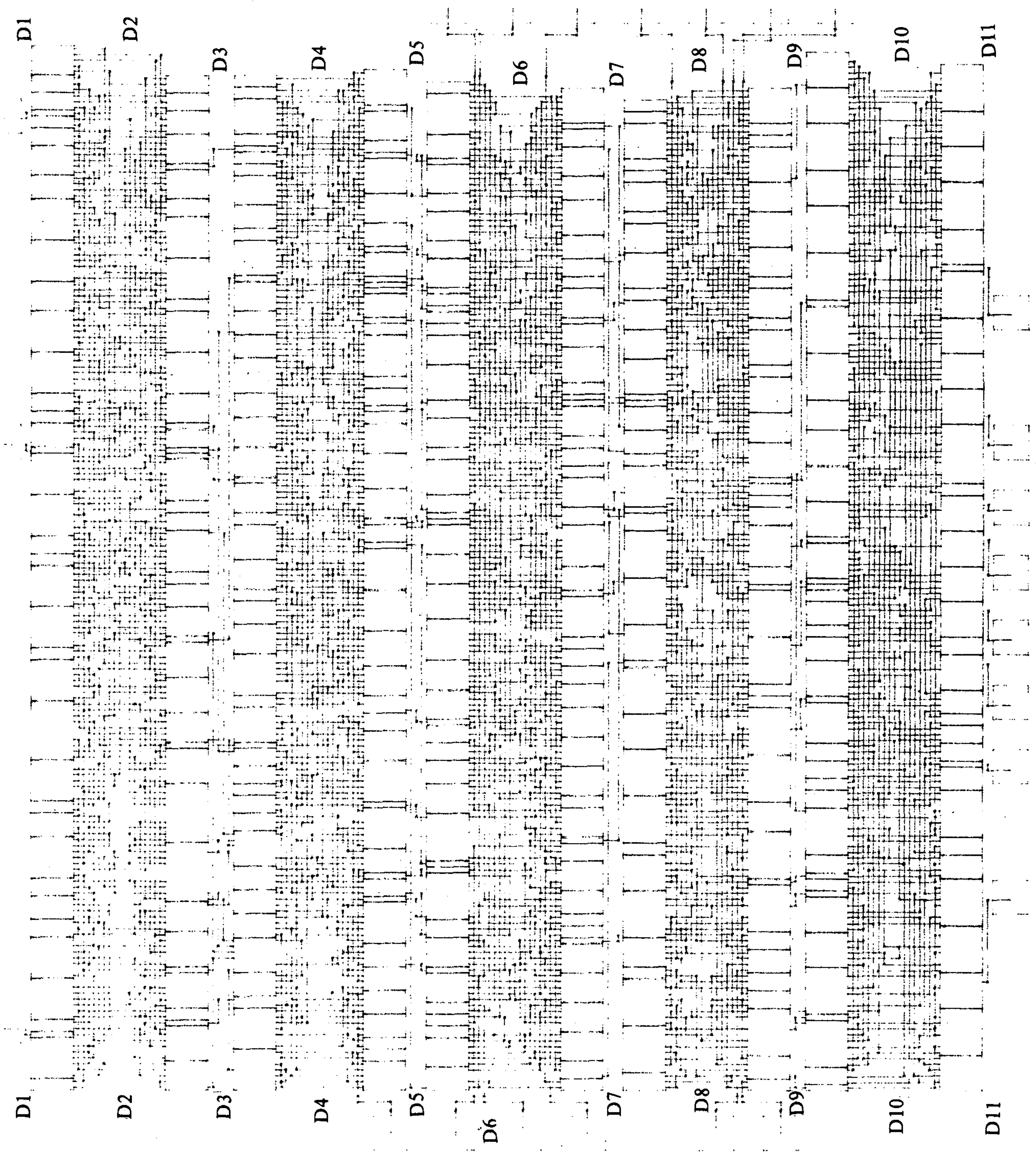

Figure 9. Example A.

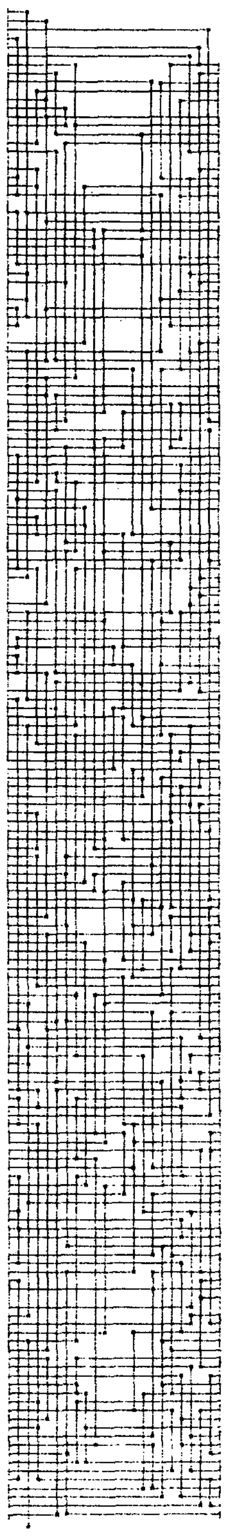

Figure 10. Example B.

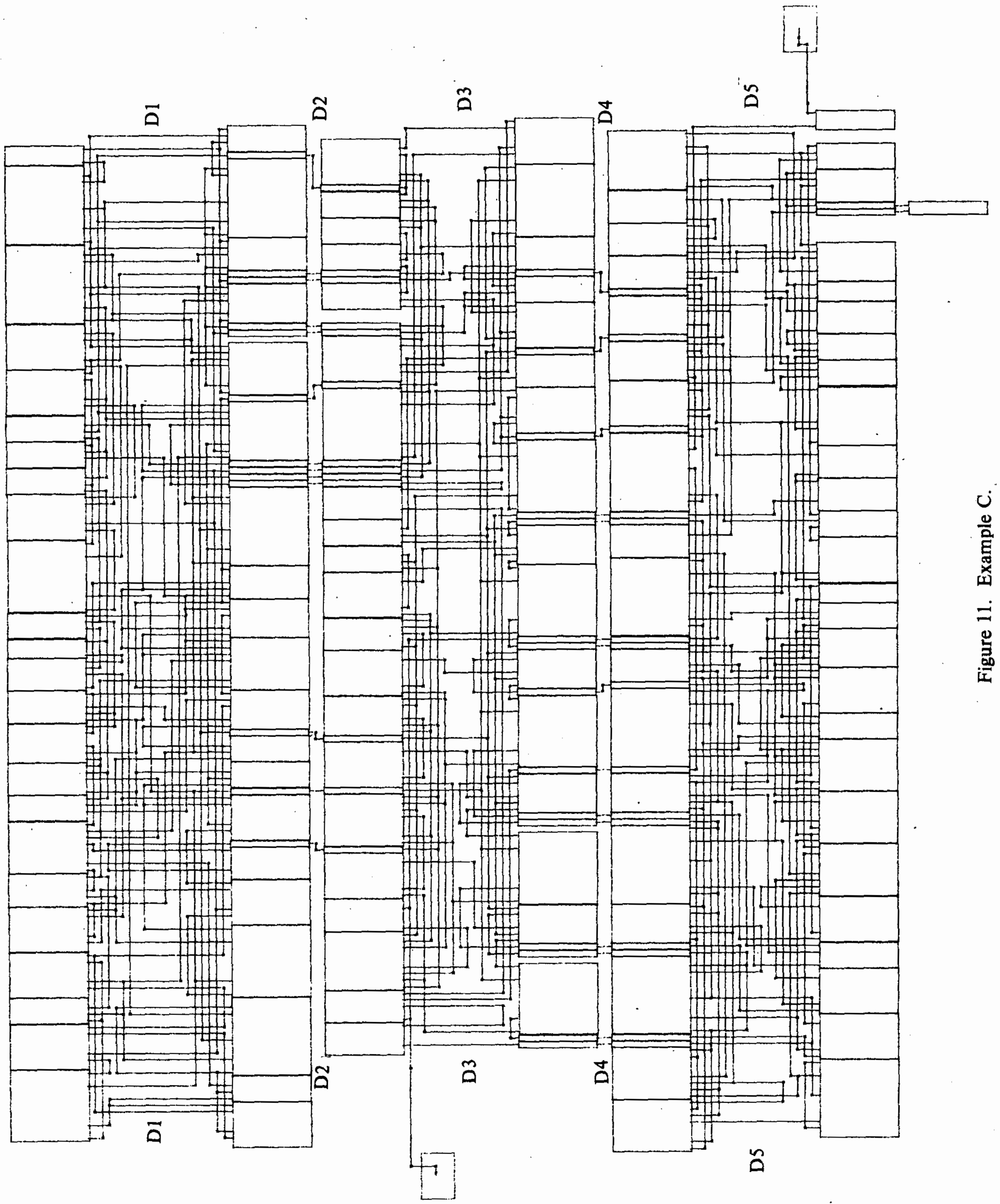

Figure 11. Example C.