WIRE ROUTING BY OPTIMIZING CHANNEL ASSIGNMENT WITHIN LARGE APERTURES

Akihiro Hashimoto and James Stevens

Center for Advanced Computation
University of Illinois, Urbana, Illinois

ABSTRACT

The purpose of this paper is to introduce a new wire routing method for two layer printed circuit boards. This technique has been developed at the University of Illinois Center for Advanced Computation and has been programmed in ALGOL for a B5500 computer. The routing method is based on the newly developed channel assignment algorithm and requires many via holes. The primary goals of the method are short execution time and high wireability. Actual design specifications for ILLIAC IV Control Unit boards have been used to test the feasibility of the routing technique. Tests have shown that this algorithm is very fast and can handle large boards.

INTRODUCTION

In the past few years, several new wire routing algorithms have been introduced. Until that time Lee's [1] algorithm had been the only widely known technique. The primary advantages that these new algorithms have over Lee's method are that they require much less computer time and much less storage space. The most noted of these new algoritms are the line search technique of Hightower [2] or Mikami [3], the cellular routing method of Hitchcock [4] and the stepping aperture technique of Lass [5]. The channel routing technique being introduced in this paper attempts to combine the advantages of each of the above algorithms. The major objectives of this new method are to increase the speed of wiring and to increase the flexibility so that a large percentages of connections can be made with a minimum wire length.

In the channel routing algorithm all connections are represented as collections of wire segments as is done by Hightower and Lass. These line segments however are free to move sideways within the boundaries of a given space which may correspond to a cell used by Hitchcock but is many times larger than the cells he uses. Finally wire segments can be moved from one space to another in a manner similar to the method used by Lass. The final assignment of the wire segments to positions within a space is accomplished in an optimum manner by a simple and efficient procedure. While the channel routing algorithm allows great flexibility in the placing and moving of wire segments, it also is very fast and easy to implement.

The channel routing algorithm was designed to be used in routing two layer printed circuit boards which have the following properties. The type of package employed is a dual inline integrated circuit package arranged on the board in straight rows and columns (Fig. 1). Also there must be a capability for a large number of "floating" via holes. The positions of these vias are determined by the channel routing program. The reason for this is that the program will assign all horizonital wire segments to one side of the board and all vertical wire segments to the other. (A modification of this rule will be discussed in Appendix A.) All connections between horizontal and vertical wires are made by means of plated through via holes. In general, each connection made on the board will cause four vias to be assigned. The dependance of this algorithm on inexpensive and reliable via holes is considered the overriding factor determining its usefulness.

The description of the channel routing technique depends heavily on the definition of a space and a channel. First of all a space is an area on a board which extends from one edge to the other and may be of any desired width. For the purposes of this description the width of a space may be defined to be equal to the distance between rows or columns of packages on the board (Fig. 2). Within a given space it will be possible in general for several wires to run parallel to each other. The maximum number of wires which can run parallel is determined by the minimum spacing between wires and the width of the space. This number will be referred to as the number of channels available in that space. As a subdivision of a space, a channel also runs from one edge of the board to the other.

The algorithm itself is composed of two stages, space assignment and channel assignment. Space assignment is the process of

breaking each connection into a set of horizontal and vertical wire segments and assigning these wire segments to spaces on the board. The details of how each connection is originally made will be discussed in a later section. The assigning of wire segments to spaces takes no account of available channels within the spaces; they are simply assigned as if each space had an infinite capacity. After all wire segments have been assigned to spaces, the channel assignment stage of the algorithm finalizes the positions of the wire segments. More than one wire segment can be assigned to one channel as long as there is no overlap between wire segments in that channel (Fig. 3). The algorithm used to assign channels places all of the wire segments in a given space into the minimum possible number of channels. In this way it can be determined whether all of the wire segments can be placed within the physical dimensions imposed by the board layout. If a given space is overcrowded, i.e. requires too many channels, wire segments can be moved from it to a nearby space which is not overcrowded. If all of the spaces on the board become filled or overcrowded through shifting of wire segments then this algorithm would not be able to wire one hundred percent of the connections.

The remainder of the paper is devoted to explaining the details involved in implementing the channel routing program. Also, a discussion is given of some of the possible extensions and modifications that can be applied to the algorithm. The specific example of wiring an ILLIAC IV Control Unit board is used to demonstrate the application. It should be noted that ILLIAC IV CU boards are large, 165 components, multilayer boards which were layed out and wired with much difficulty due to transmission line effects. For the purposes of this paper it will be assumed that these effects due to extra high speed can be ignored so that all of the signal wires can be connected on two layers using minimum distance wiring wherever possible.

BOARD LAYOUT

The package arrangement on an ILLIAC IV CU boards (Fig. 4) is composed of eleven rows of packages with fifteen packages per row giving a maximum of 165 packages per board. Each component is a dual inline integrated circuit package with sixteen pins. The arrangement of holes for mounting these packages is very orderly with all of the sixteen holes lined up in two rows. The orderly arrangement of packages on the board gives rise to large open spaces which run between rows and columns of packages. It is the wiring of these spaces which is of major interest in this algorithm. The board being considered has two sides available for wiring, one for horizontal wire segments only and one for vertical wire segments only. Since wire segments are to be assigned to spaces on the board, the side used for horizontal wire segments will be divided into horizontal spaces and the other side into vertical spaces.

On the horizontal side of the board there are ten identical horizontal spaces between rows of packages. There is also another type of horizontal space, the space beneath a row of packages. Because of the arrangement of pins the only difference between these two types of horizontal spaces is their width (Fig. 5). On the vertical side of the board there are also two types of spaces, the spaces between columns of packages and the spaces beneath columns of packages. Since the vertical spaces beneath columns of packages contain the package pins they must be handled slightly differently than any other spaces. For the specific case of ILLIAC IV CU boards there are 19 horizontal channels and 16 vertical channels available per package. This can be broken down to the channel capacity for each type of space on the board. Horizontal spaces between packages have a 14 channel capacity while horizontal spaces beneath packages have 5 channels. Vertical spaces between packages have a 9 channel capacity and vertical spaces beneath packages have 7 channels (one wire channel between each of 7 pairs of pins).

Plated through communicating holes, called vias are assigned by the algorithm at the end points of each wire segment. These vias allow wire segments on one side of the board to be connected to wire segments on the other side so that a completed connection can be made. In general many of these vias will be assigned during the course of wiring one board. For this reason the vias must not have fixed positions and at the same time they must be reliable. In the process of performing the routing the information concerning vias can be used to create drilling instructions for producing the necessary holes. At this point it should also be mentioned that great care must be taken to allign the end points of the two wire segments to be connected so that the proper connection will be made and no shorts will occur.

SPACE ASSIGNMENT

Each connection made on the board is specified by a starting pin and an object pin. The connection itself is in general

broken up into five wire segments (Fig. 6). At each of the two pins a short vertical wire segment (A,E) is constructed which initially extends to the center of one of the adjoining horizontal spaces. A horizontal wire segment (B) is next started from the end of the short vertical wire (A) at the starting pin and extended to the center of a vertical space which adjoins the package containing the object pin. From that point a vertical wire (C) is created which extends to the center of the horizontal space adjacent to the object pin. The final wire is a horizontal wire (D) which completes the connection. Many alternatives to the above description exist all of which produce a minimal or close to minimal connections. In an attempt to distribute wire segments evenly over the horizontal and vertical spaces, an arbitrary means of selecting one of the possible minimal paths has been employed.

The space assignment method described above is a simplified technique which may not allow complete wiring of the board. Basically, incomplete wiring will result from one of two conditions. First, it may not be possible to make all of the connections required within the physical restriction imposed by the board dimensions. This problem may be solved by using a different placement of components on the board or by decreasing the number of components assigned to the board. Secondly, incomplete wiring may result from the overcrowding of one area of the board. It is possible that the wire segments assigned to one space may overflow the channel capacity of that space whereas all other spaces may have very few channels used. Some steps can be taken to avoid such a situation or to alleviate it after it occurs. At the time of assigning a wire segment to a space it is possible to determine how many wires have already been assigned to that space and to the other space on the board. This information can be used to place a wire segment in the least crowded of the spaces to which it can be assigned. On the other hand, after all channel assignments have been made it is also possible to reassign wire segments to new spaces so that a more even distribution of required channels can be realized. Since each wire segment can be specified by its end points, it is an easy matter to reassign a wire segment to a new space and to make the necessary adjustments to the end points of wire segments it connects to.

In this initial assignment of wire segments to spaces, all wire segments are assumed to travel down the center of their assigned spaces. This assignment creates three kinds of conflicts which must be resolved at a later time. The conflict of many wires overlapping in the center channel of each space is resolved by the second stage of the algorithm. Via conflicts which arise from having many wire segments end at the same point are also resolved by the second stage. One other type of conflict may result from this method of space assignment. Since the connection to each package pin is made by a short vertical wire segment which does not use up any channel area, (vertical channels go between package pins) it is assumed that these connections can always be made and therefore no information must be kept about them. However two of these short wires may be caused to overlap when channels are assigned to horizontal wires (Fig. 7(a)). This type of conflict can be handled in several ways and is presently resolved by an extra stage in the algorithm which reroutes one of the connections. If space is available in an adjacent channel this is quite easily done (Fig. 7(b)).

CHANNEL ASSIGNMENT

After space assignment is completed each space will contain a set of wire segments. These wire segments are regarded as a set of intervals having an upper bound and a lower bound (Fig. 8). The object of channel assignment is to position all wire segments in the fewest possible number of channels without any wire segments overlapping. The first step in the procedure is to search the list of intervals for the element which has the greatest upper bound. This element is assigned to the first channel and eliminated from the list. The list is then searched for the interval which has the greatest upper bound which is less than the lower bound of the previously chosen interval. This element is also assigned to the first channel and eliminated from the list. The search is repeated until no element fulfills the requirements, at which time the entire process is repeated for the next channel. When all of the intervals have been eliminated from the list the channel assignment is complete. This algorithm always uses the minimum possible number of channels to finalize the positions of all of the wire segments in a given space. The proof of minimality will be given in the next section along with an interesting extension.

When the channel assignments are being made, precautions must be taken to ensure that the wire segments which must be joined to form a connection have end points which coincide. At the conclusion of the space assignment stage many conflicts may exist since all wire segments are assumed to travel down the center of the space. Assume that

two horizontal wire segments 1 and 2 (Fig. 9(a)) have the same end point. If vertical wire segment 3 is to connect to 2 and vertical wire segment 4 is to connect to 1 then the two connections are sharing one via hole. Channel assignment is now performed on all vertical wire segments. Since the upper bound of wire 4 is not less than the lower bound of wire 3, these two wire segments may not be assigned to the same channel. For this reason there can no longer be a via conflict between connections 1-4 and 2-3, but wire 1 may be in conflict with wire 2 (Fig. 9(b)). If such a conflict exists then wire 1 and wire 2 cannot be placed in the same channel when horizontal channel assignment is performed. When all channel assignment is complete there is no conflict between connection 1-4 and connection 2-3 (Fig. 9(c)). Other combinations of conflicting wire segments can be formed, but the channel assignment procedes in such a way as to always eliminate wire overlap and via conflicts.

PROOF OF OPTIMAL CHANNEL ASSIGNMENT

The set of wire segments assigned to a given space can be regarded as a set of intervals as previously stated (Fig. 8). This set of intervals can be represented by a directed graph (Fig. 10(a)). The nodes represent intervals and an arrow is directed from one node to another if and only if the lower bound of the first interval is greater than the upper bound of the other interval. Assigning wire segments to channels corresponds to covering the directed graph which represents the set of wire segments with a set of directed paths. A directed path is composed of a set of nodes which are connected by arrows such that all arrows point in the direction of the path (Fig. 10(a)). Selecting a path corresponds to assigning the wire segments represented by the nodes on the path to one channel without overlap. A path cover of a graph is a collection of paths which includes each node of the graph exactly once. Two nodes in a directed graph are said to be incomparable if no path exists which includes both nodes. It has been shown that the greatest number of mutually incomparable nodes which can be found in the acyclic graph is equal to the minimum number of paths which can be used to cover the graph. This property was first proven by Dilworth and the corresponding theorem bears his name. [6] The largest collection of mutually incomparable nodes is referred to as a maximal incomparable set and in general several distinct incomparable sets from one graph may be maximal.

To show that the channel assignment algorithm is optimal it must be shown that the number of channels used is equal to the minimum number of paths which can cover the corresponding directed graph. First of all a method of constructing paths must be found which corresponds to the method used to assign wire segments to channels. Such a path can be formed by choosing the node with the greatest upper bound then choosing the arrow emanating from that node which leads to the node with the greatest upper bound. The path is complete when a node is reached which has no outgoing arrows. The first step in the proof is to show that such a path which we will call a max chain contains exactly one member from each maximal incomparable set of the graph. By definition no path can contain more than one element of a maximal comparable set of nodes. Assume on the other hand that a max chain can be found which contains no members of a given maximal incomparable set. If this assumption leads to a contradiction then the first step is demonstrated. Since none of the nodes in a path $N = (N_1, \ldots, N_p)$ are in a given maximal incomparable set $M = \{M_1, \ldots, M_q\}$ they must all be comparable with a member of that set (Fig. 11.) There is a node N_j which is the highest node with an incoming arrow which leads from a member M_k of M. The next preceding node N_i must have an outgoing arrow which leads to a member M_ℓ of M. From the comparisons described above it follows that the lower bound of M_k is greater than the upper bound of N_j and the upper bound of M_ℓ is greater than the lower bound of M_k. Therefore the upper bound of M_ℓ is greater than the upper bound of N_j. This proves that N is not a max chain since the arrow from N_i to M_ℓ was not chosen. Therefore every max chain contains exactly one member from each maximal incomparable set. Eliminating the nodes along a max chain from the directed graph reduces the size of each maximal incomparable set by one, reducing the number of paths required to cover the graph by one. Therefore the number of max chains which is equal to the number of channels used is always the minimum. The authors discussed the relation between this channel assignment algorithm and the other graph theoretical topics [7].

This algorithm assumes that each wire segment is an indivisible entity. The question arrises as to whether fewer channels would be required if a wire segment could be split into independent subsections connected by perpendicular wire segments. To show that such a modification would not improve the number of channels required, a proof will now be given that the size of the maximal incomparable sets is not decreased. Assume that a wire segment which is a member of a given maximal incomparable set can be broken

into several nodes so that none are incomparable with the maximal incomparable nodes. The lower bound of any of these nodes would be equal to the upper bound of the following node to preserve continuity. Figure 11 can again be used to show that a contradiction is generated. Node M_k is shown comparable with node M_ℓ ($L(M_k) > U(N_j) = U(N_i) > U(M_\ell)$). Therefore such a division of one wire segment can not reduce the size of a maximal incomparable set and thus cannot reduce the number of channels required to place a set of wire segments.

If channel assignment is performed on all the wire segments on a board without regard for the interconnections they have, a lower limit can be found for the number of horizontal and vertical channels which must be available for wiring the board. The way this is done is by assigning two wire segments, one horizontal and one vertical, for each connection to be made such that all wire segments are of minimum length. All of the horizontal wire segments are assigned to one horizontal space and all vertical wire segments to on vertical space. Finally channels are assigned within these two spaces without any attempt to preserve the connections. The result is a lower limit on the number of horizontal and vertical channels which must be available on the board in order to wire all of the connections. This result can be used as an optimal limit against which the effectiveness of a given wire routing algorithm can be measured. Also such a result can be used in the early stages of a system design to help determine the physical dimensions of a board once its logic has been specified.

PROGRAM AND RESULTS

The channel routing algorithm has been programmed in ALGOL and run on a Burroughs 5500 computer. The program is divided into two stages just as the algorithm is divided. The first stage completes all connections to be made on the board divides them into horizontal and vertical wire segments and assigns all wire segments to spaces. The second stage of the program finalizes the positions of all wire segments and determines whether the board has been successfully wired. The input to the program is a list of connections each represented only by the coordinates of its two end points. The output of the program is a complete specification of the wiring of the board.

Each wire segment created in the first stage of the program must be associated with information giving its position and how it is to be connected. The position of a wire segment is specified by the coordinates of its end points. For each end point a pointer is kept which gives the memory location of the wire segment that connects to that end point. If one end of a wire segment connects to a pin, a special pointer is inserted. The final position of a wire segment is also recorded by storing the number of the channel to which it is assigned. A tag is also associated with each wire segment to indicate whether a final assignment has been made for that wire segment. Since partial word fields can be conveniently manipulated in ALGOL, only one word of storage is needed to specify each wire segment completely (Fig. 12).

During the space assignment stage of the program one word of information is created to represent each wire segment used. These words are then stored in one of two arrays. One array for horizontal wire segments and the other for vertical wire segments. The first dimension of each of these arrays tells which horizontal or vertical space the wire segment is assigned to. All of the wire segments assigned to a given space are stored in the array row corresponding to that space. During the second stage of the program the assignment of channels within spaces is done for each space independently. For this reason only one array row must reside in core at a time. In a multiprogramming environment this is quite convenient, especially since all array rows are the same size. In the present program the array rows have 256 elements so several of them can fit easily on a small core machine even during multiprogramming. (B5500 has 32K words of memory.)

Several test runs have been performed using wirelists for ILLIAC IV CU boards. The maximum number of components on a board is 165 and for each component there are 16 vertical channels and 19 horizontal channels available. Some boards with up to 103 components and 500 connections were wired completely in less than 30 seconds processor time. The program was tested using 40 boards with 135 or fewer components. Of these, 23 were wired with less than 13 channels total overflow (i.e. the sum of extra channels required from all spaces which overflowed). The wiring of these 23 boards should be easily completed by reassigning a few wire segments to new spaces. The remaining 17 boards could probably be completely wired on two layers after considerable shifting of the wire segments which have overlapped. All of the CU boards with more than 135 components have at least 151 components and most of these are not considered wireable on two layers. Channel assignments were attempted for these boards without regard for space capacity and up to 1200 connections were assigned

in less than one minute processor time.

POSSIBLE IMPROVEMENTS

The short run time and high wireability of the developed routing program have proven the usefulness of the channel routing algorithm. Because of the simple nature of the algorithm a channel routing program is very inexpensive to implement. In order to improve the applicability of the program, some new features listed below are being considered for use in future versions of the routing program.

(a) Parallel running wires

In a high speed logic circuit, any two signal wires are not allowed to run side by side for more than a certain predetermined distance. Because of the data structure employed the detection of wires which run parallel for too long a distance is a simple matter. Also, such a problem can be easily resolved by changing channel numbers. Suppose two wires in channels 1 and 2 of a given space run parallel for too long a distance. The problem can be eliminated by exchanging channel 2 with channel 3.

(b) Better space assignment

The current version of the channel routing program assigns a unique vertical or horizontal space to each wire segment even though there are a few alternatives. The selection of a particular space is made in a fixed way. This assignment scheme may cause some local congestion which could have been avoided if the wire segments were distributed carefully. A more flexible space assignment algorithm is now being developed in order to eliminate the unnecessary local congestion which may decrease the wireability.

(c) Better package placement

The channel routing program generally divides a wire into five wire segments with four vias. If a wire connects the two pins which are either on the same horizontal row of pins or two adjacent horizontal rows of pins, the wire will be divided into three wire segments with two vias. Accordingly, a better routing will be obtained if the channel routing program is combined with a placement algorithm which considers the above fact instead of or in conjunction with the total wire length. Taking advantage of the short run time of the channel routing program, an iterative operation between routing and placement programs seems possible.

(d) Reduction of vias

The approach described above will reduce the number of vias before the routing is made. It is also possible to reduce the number of vias after routing is made. The authors have developed an algorithm which, when applied on a routed board, reassigns the wire segments to sides of the board in such a way that a minimum number of vias are used. The details of the algorithm will be described in Appendix A.

ACKNOWLEDGEMENT

The authors wish to thank Prof. D. L. Slotnick for his valuable advice and encouragement. Thanks are also due to Mr. David Pearson of Automation Technology Inc., for his discussions.

REFERENCES

1. C. Y. Lee, "An algorithm for path connections and its applications", IRE Transactions on Electronic Computers, (September 1961), 346-365.

2. D. W. Hightower, "A solution to line-routing problems on the continuous plane", Proc.1969 Design Automation Workshop,1-24.

3. K. Mikami and K. Tabuchi, "A Computer Program for Optimal Routing of Printed Circuit Conductors." Proceedings of the IFIP Congress, 1968 Vol. 2 pp. 1475.

4. R. B. Hitchcock, "Cellular wiring and the cellular modeling technique", Proc. 1969 Design Automation Workshop, 25-41.

5. S. E. Lass, "Automated printed circuit routing with a stepping aperture", Comm. of ACM, 12, No. 5, (May 1969), 262-265.

6. R P. Dilworth, "A Decomposition Theorem for Partially Ordered Sets," Ann. Math., (2) 51 (1950), 161-166.

7. A. Hashimoto and J.E. Stevens, "Path Cover of Acyclic Graphs", ILLIAC IV Document No. 239, December 24, 1970.

8. U. R. Kodres, "Formulation and solution of circuit card design problems through use of graph methods", in Advances in Electronics Circuit Packaging, Vol. 2, G. A. Walker, Ed., New York: Plenum 1962, 121-142.

APPENDIX A

ALGORITHM FOR REDUCTION OF VIAS

After accomplishing the wire routing by the channel routing technique described in this paper, we may be able to reduce the number of vias by moving some wire segments from one side of the board to the other. For example, the routing shown in Figure A-1 requires five vias, providing that all the vertical wire segments have been assigned to one side of the board and all the horizontal ones to the other. Small circles in the figure denote the vias. Only two vias remain necessary if the wire segments are assigned as shown in Figure A-2.

Now we will show that the problem of minimizing the number of vias can be solved by finding a maximum bipartite subgraph of the graph derived from the routing. The two sides of a board will be identified as side A and side B for convenience.

Let $V^h = \{v_i\}$ and $V^v = \{v_j\}$ denote the sets of horizontal and vertical wire segments, respectively, and define $V = V^h \cup V^v$.

Let B^c be the set of all the unordered pairs (v_i, v_j) such that v_i and v_j intersect if placed on the same side of the board. Similarly, B^v is defined to be the set of all the unordered pairs (v_i, v_j) such that v_i and v_j must be electrically connected. Let $B = B^c \cup B^v$. Note that one component of a pair in B belongs to V^h and the other to V^v.

Regarding V and B as the sets of nodes and branches, respectively, we obtain a bipartite graph $G = (V, B)$. A bipartite graph is a graph without a circuit of odd length. Graph G of Figure A-3 is derived from Figure A-1. The solid lines and the dotted lines correspond to the branches in B^c and B^v, respectively.

If $V = V^A \cup V^B$ and $\phi = V^A \cap V^B$, then the pair $\{V^A, V^B\}$ is called a partition of V. A partition $\{V^A, V^B\}$ is feasible if every branch of B^c connects a node in V^A and a node in V^B. A feasible partition $\{V^A, V^B\}$ defines an assignment of the wire segments to the sides of the board, that is, the wire segments in V^A and V^B can be placed on the sides A and B, respectively, without a conflict. For a given feasible partition $\{V^A, V^B\}$, a branch in B^v requires a via if, and only if, it connects a node in V^A and a node in V^B. Thus, our problem is reduced to that of finding a feasible partition $\{V^A, V^B\}$ which has a minimum number of branches of B^v connecting a node in V^A and a node in V^B.

A subgraph of G, defined by $G^c = (V, B^c)$, consists, in general, of several connected components $G_1^c, G_2^c, \ldots, G_n^c$, $n \geq 1$. Let $G_i^c = (V_i, B_i^c)$ and define $V_i^h = V_i \cap V^h$ and $V_i^v = V_i \cap V^v$ (Fig. A-4).

Proposition 1. No partition other than $\{V_i^h, V_i^v\}$ is feasible for each V_i, $i = 1, 2, \ldots, n$.

This proposition is derived from the definition of the feasibility and the fact that G_i^c is a connected bipartite graph.

If a branch in B^v connects a pair of nodes in V_i for some i, then the branch (or the corresponding via) is called essential. Let B^e be the set of such branches. For example, in Figure A-4 $B^e = \{(5,6)\}$.

Proposition 2. An essential via can not be eliminated.

A branch in B^e must connect a node in V_i^h and a node in V_i^v for some i, but the two nodes can not belong to the same subset of a feasible partition due to Proposition 1.

Accordingly, only those vias which correspond to the branches in $B^v - B^e$ are the candidates for elimination. In our example, $V^v - B^e = \{(1,2), (3,4), (7,8), (9,10)\}$.

Proposition 3. Let $\{V^A, V^B\}$ be a feasible partition. If $V_i^h \subset V^A$ and $V_j^v \subset V^A$ (consequently, $V_i^v \subset V^B$ and $V_j^h \subset V^B$) then all the vias corresponding to the branches connecting V_i and V_j are eliminated. Otherwise, all of the vias must remain.

This proposition follows from the fact that there is no branch between V_i^h and V_j^h, or between V_i^v and V_j^v. For example, in Figure A-4 if V_1^h, $V_2^v \subset V^A$ and V_1^v, $V_2^h \subset V^B$ then the two vias corresponding to the branches (1,2) and (9,10) are removed. On the other hand, if V_1^h, $V_2^h \subset V^A$ and V_1^v, $V_2^v \subset V^B$ then both vias are not removed.

Now, replace each V_i of G by a node v_i^* and eliminate all self-loops, then a graph $G^* = (V^*, B^v - B^e)$, $V^* = \{v_1^*, v_2^*, \ldots, v_n^*\}$ will be obtained (Fig. A-5).

Assign 0 or 1 to each of the node of G^* arbitrarily, and let $V_i^h \subset V^A$, $V_i^v \subset V^B$ if v_i^* is assigned 0 and $V_i^h \subset V^B$, $V_i^v \subset V^A$ if v_i^* is assigned 1, then $\{V^A, V^B\}$ is a feasible partition of V. Conversely, any feasible partition of V has a 0-1 assignment of G^* due to Proposition 1. For a given partition

derived from a 0-1 assignment, the number of remaining non-essential vias is equal to the number of branches in G^* which connect two nodes having the same value. The following proposition will be easily derived from the definition of the bipartite graph.

Proposition 4. All the non-essential vias can be eliminated if, and only if, the graph G^* is bipartite.

Thus the via minimization problem is reduced to the following one. "Given a graph G^*, find a bipartite subgraph of G^* with a maximum number of branches."

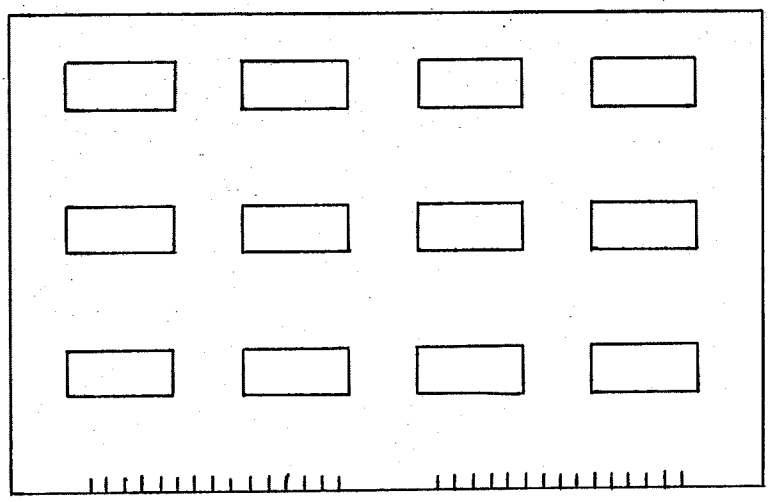
The graph G^* of Figure A-5 is not bipartite and either the branch (v_1^*, v_3^*) or (v_2^*, v_3^*) must be removed in order to obtain a bipartite subgraph. If (v_1^*, v_3^*) is selected, then the corresponding partition will be as follows:

$$V^A = V_1^h \cup V_2^v \cup V_3^h = \{1, 2, 3, 5\}$$

$$V^B = V_1^v \cup V_2^h \cup V_3^v = \{4, 6, 7, 8, 9, 10\}$$

The layout of the wire segments, specified by this partition, is shown in Figure A-2. Two vias remained; one for the essential branch (5,6) and the other for the non-essential branch (3,4) which was removed in the bipartization process.

The problem of finding a maximum bipartite subgraph of a given graph may be solved by either the minimum cover algorithm or the integer linear program [8]; however, no efficient algorithm is known. But the authors expect rather large connected components, G_i^c's, and a small graph G^*, and hence simplicity in solving the problem.



PRINTED CIRCUIT BOARD LAYOUT

Figure 1

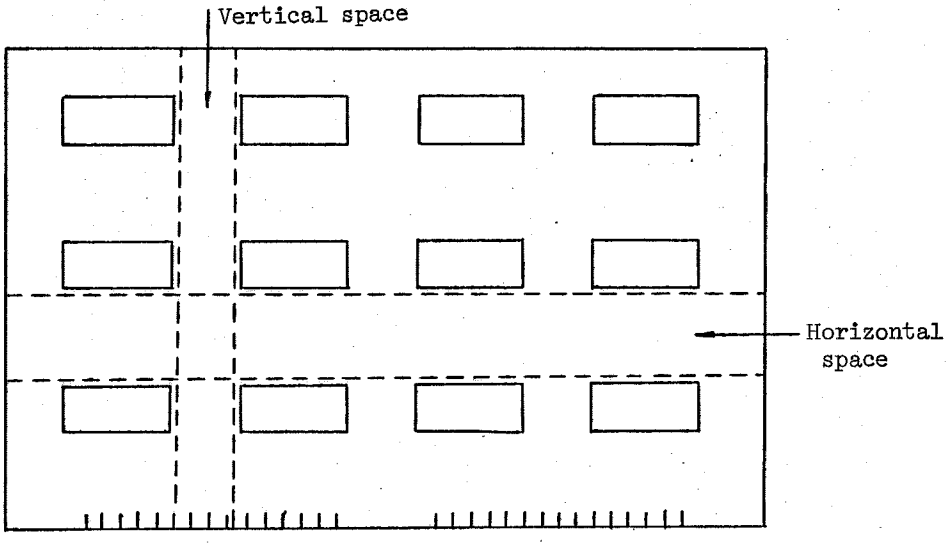


Figure 2

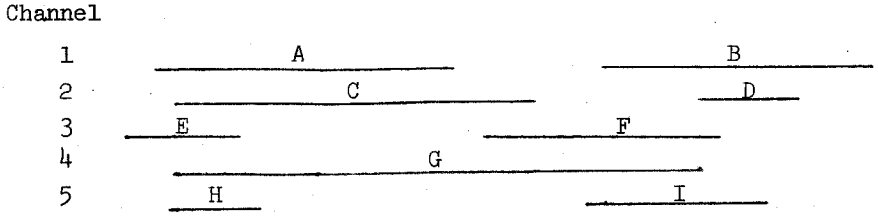
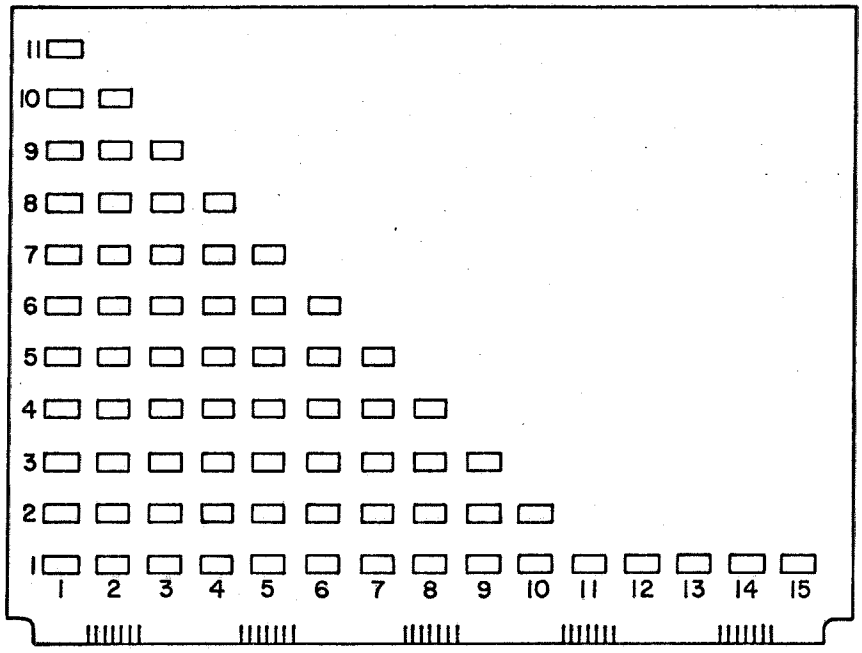


Figure 3



ILLIAC IV CU BOARD

Figure 4

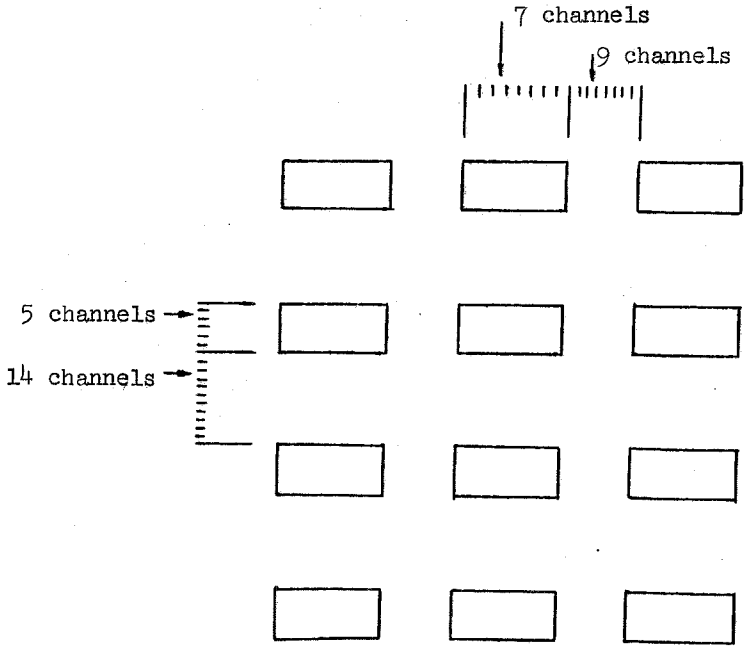


Figure 5

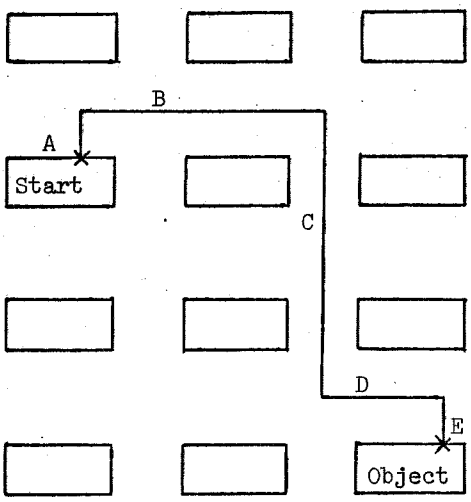


Figure 6

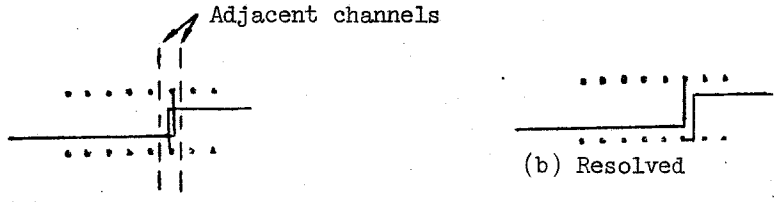


Figure 7

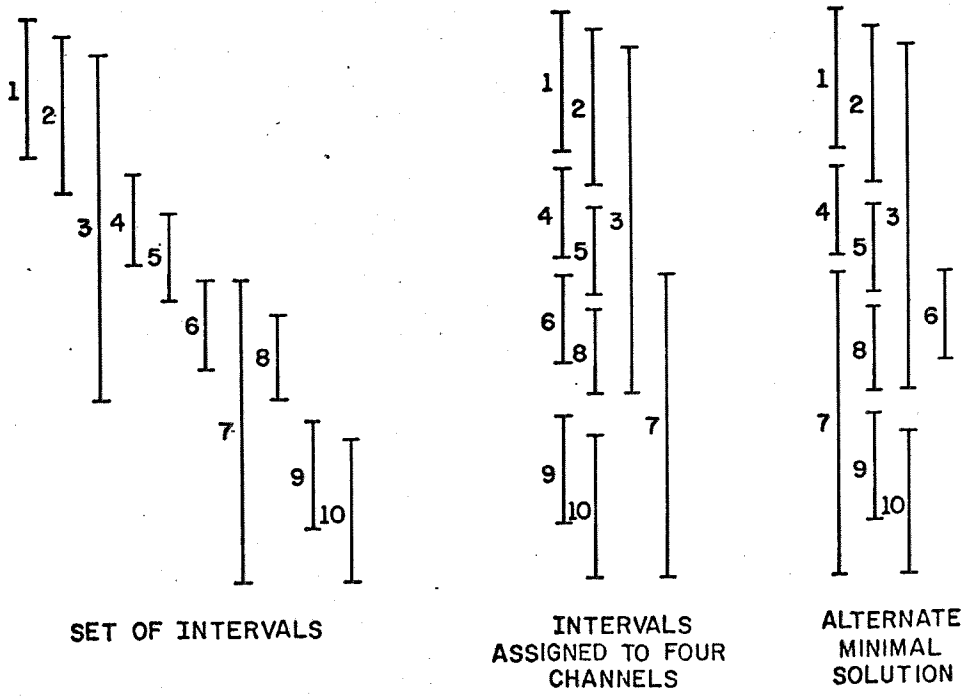
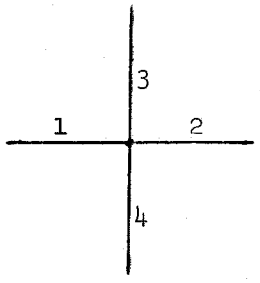
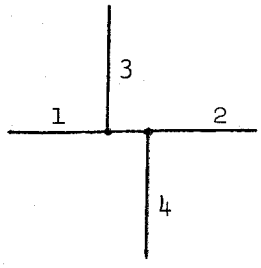


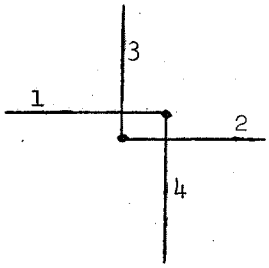
Figure 8



(a) After space assignment

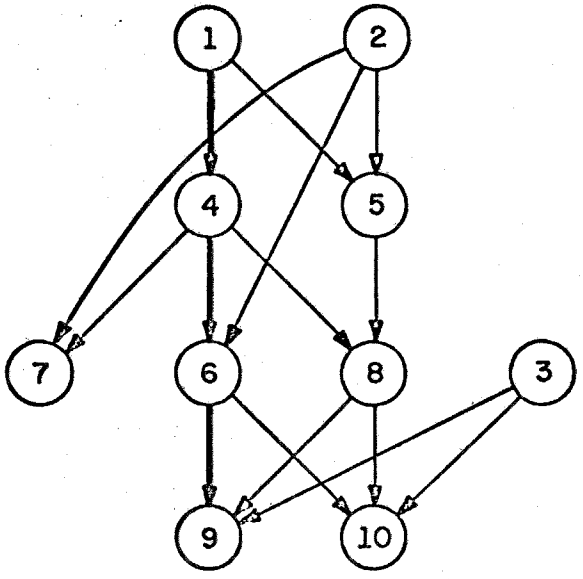


(b) After vertical assignment

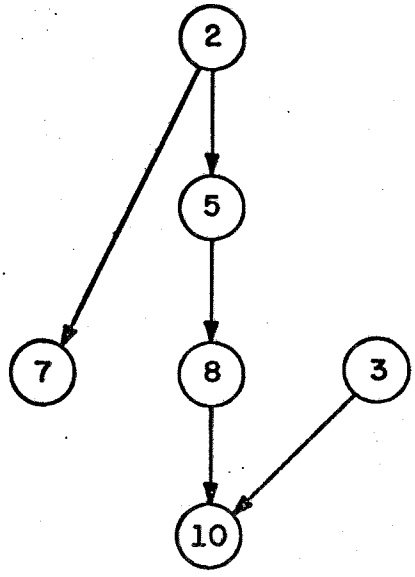


(c) After horizontal assignment

Figure 9

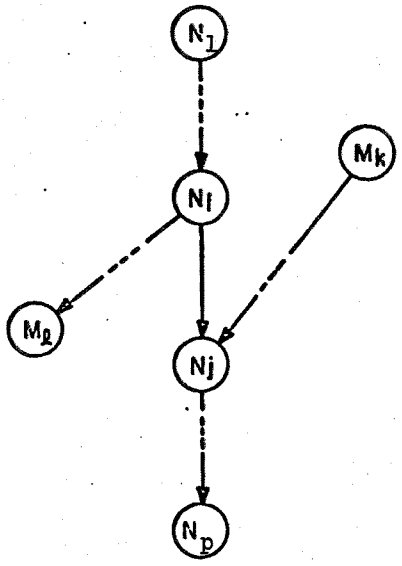


(a) DIRECTED GRAPH



(b) REDUCED GRAPH

Figure 10



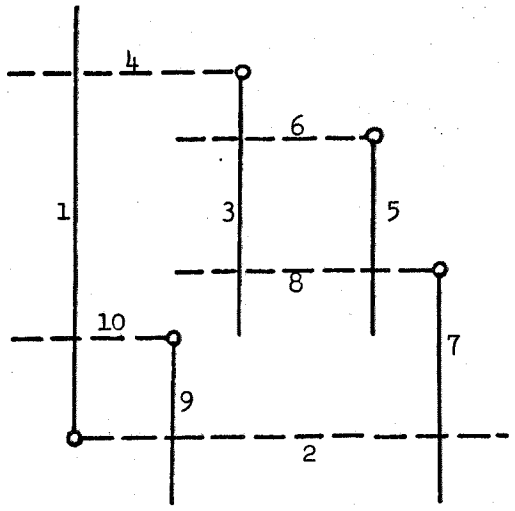
GENERALIZED PATH

Figure 11

Channel	T	Left Pointer	Right Pointer	Left end	Right end

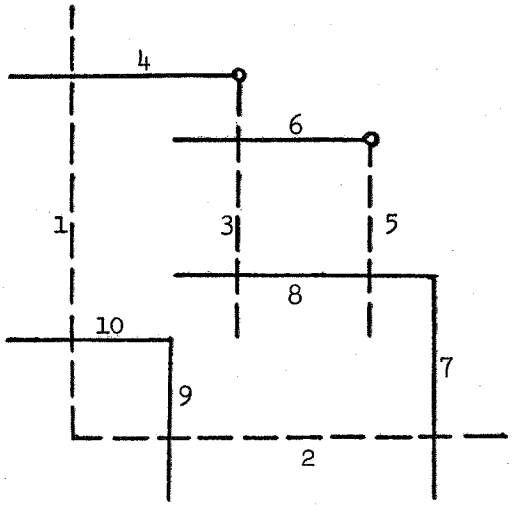
HORIZONTAL WIRE SEGMENT

Figure 12



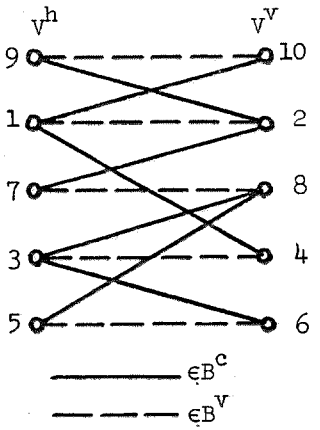
ORIGINAL ASSIGNMENT

Figure A-1



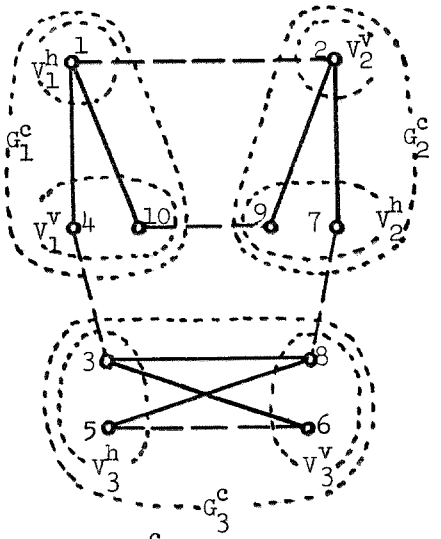
IMPROVED ASSIGNMENT

Figure A-2



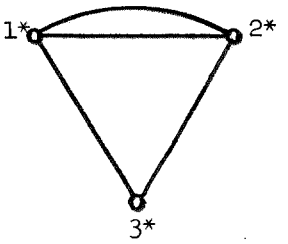
GRAPH G DERIVED FROM Fig. A-1

Figure A-3



GRAPH G^c DERIVED FROM G

Figure A-4



GRAPH G* DERIVED FROM G

Figure A-5