GATE ASSIGNMENT AND PACK PLACEMENT: TWO APPROACHES COMPARED

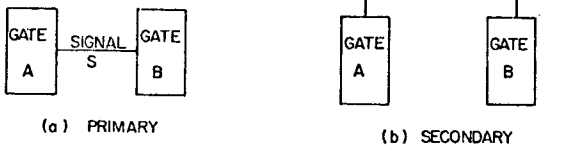
Frank Luebbert and Mike Ulrey
NCR Corporation
Wichita, Kansas 67226

ABSTRACT

In this paper we discuss mathematical models for the gate assignment and pack placement problems. Relative to objective functions to be described, heuristic methods of solution are discussed, one for gate assignment and two for pack placement. Performance figures for two actual boards are presented, together with a comparison to manual layout for one of the boards.

I. GATE ASSIGNMENT: THE MODEL

For the purposes of this paper, we ignore discrete components. We assume there is a given collection G of gates, each associated with a unique part number. Also given are a set E of edge connectors and a set of signals. A signal is a subset of $G \cup E$ with at least two elements. Two gates A and B can be thought of as being connected by a signal S if $A \in S$ and $B \in S$, as shown in Figure 1(a) below. This is called a primary connection between A and B.



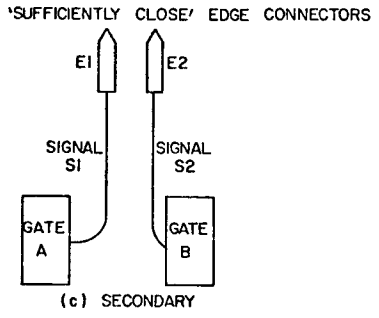


FIG. 1 GATE CONNECTIONS

Gates A and B have a secondary connection if there is a third gate C such that A and B are primarily connected to C by distinct signals S1 and S2, respectively. (Figure 1(b)). In Figure 1(c) we see a second situation in which A and B are said to be secondarily connected, namely that A and B are connected to "sufficiently close" edge connectors E1 and E2 by distinct signals S1 and S2, respectively. This presumes that there is some notion of distance between edge connectors. In the usual interpretation of the term "edge connectors" on an actual board, these objects are physically fixed from the start, and distances between them easily computed. The definition of "sufficiently close" is controlled by an adjustable parameter in practice.

Also given is a set P of packs, each associated with a unique part number, as in the case of gates. In fact the two sets of part numbers are actually the same set. Consider all the mappings $p: G \to P$. Every such p is called an assignment. It is further assumed that there is a non-empty subset of assignments called admissible assignments. In practice, important factors in determining admissibility are part numbers and control signals.

Now let PC(A,B) and SC(A,B) denote the numbers of primary and secondary connections, respectively, between gates A and B. If C and D are two packs, define $d(C,D) = 0$ if $C = D$ and $d(C,D) = 1$ if $C \neq D$. Let $p: G \to P$ be an admissible assignment and a and b be positive real numbers. Then define the total gate connection cost TC(p) by

$$TC(p) = \Sigma(a \cdot PC(A,B) + b \cdot SC(A,B)) \cdot d(p(A),p(B))$$

where the sum is over all the pairs of gates A and B. Then the object is to find the admissible assignment p which minimizes TC(p).

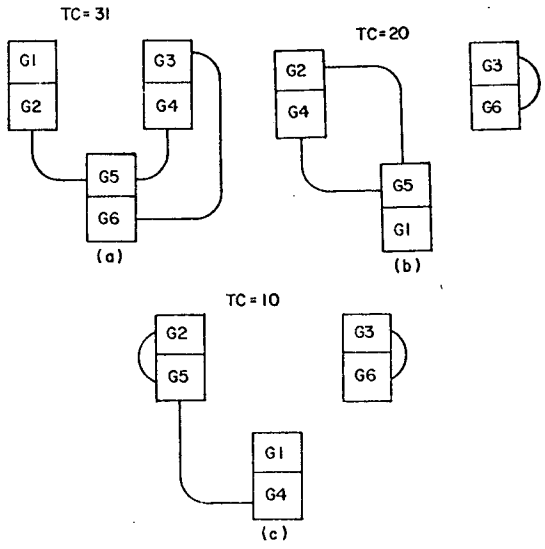


FIG. 2 GATE CONNECTION COSTS (a=10, b=1)

See Figure 2 above for some examples of gate connection costs. All connecting lines represent primary connections.

II. PACK PLACEMENT: THE MODEL

The givens for pack placement begin with a set P of packs, a set of signals, and a set L of locations. A signal S is a subset of P with at least two elements, and two packs A and B belonging to the same signal S can be visualized as being connected by S, similar to the gate situation as pictured in Figure 1(a). This is called a primary connection between A and B. Secondary connections are not considered for pack placements (though they could be).

One of the assumptions we shall make is that the packs will eventually be laid out on the board in a very regular matrix-like array. Of course this is not always true and there are even minor exceptions on boards which are predominantly this way. Almost all of the boards we have seen that are produced using auto-insertion are this way, however.

In order to motivate our discussion of locations, therefore, picture a board oriented and laid out in a manner similar to Figure 3 below. The pack locations form a regular array with a known number of rows and columns. For our purposes, we assume every pack "occupies" an integral number of locations, although in reality this is not quite true. For example, a 14- or 16-pin pack would occupy one location, while a 24-pin pack would occupy two such locations, say. Hopefully, all of these parameters can be adjusted to provide a good approximation to the board in question. The specific pack locations can all be determined more precisely after their "optimum" relative locations in this array are fixed. We also assume there is a single row of edge connectors across the bottom for purposes of this discussion.

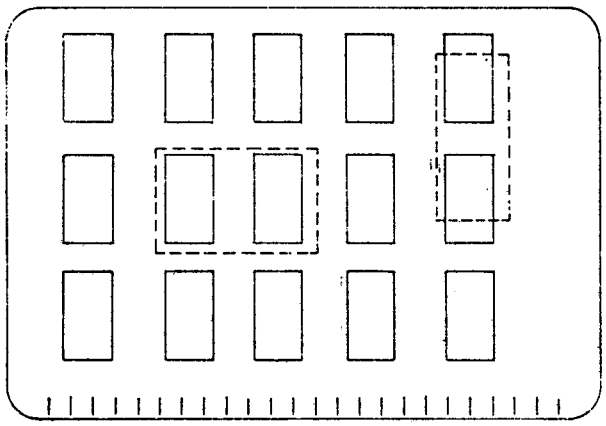


FIG. 3 3X5 LOCATION ARRAY

Now let l and m denote two locations in this array which are found at row I, column J and row K, column L respectively. Then we define the distance between them by

$$d(l,m) = |I-K| + |J-L|$$

A function f:L → P (onto but not necessarily one to one) is called a placement. As in the case of gate assignments, it is assumed there is a known non-empty subset of these placements which are called admissible. For example, a pack occupying more than one location must be assigned to adjacent locations which as a whole correspond to the shape and size of the pack. Another possible constraint is that certain packs are fixed.

Among these are what we call edge connector "packs", which are not packs at all but groups of edge connectors so chosen as to add a bottom row of pack locations to the array, one location per column.

Signal connections from such a "pack" to an ordinary pack are comprised of all the connections from the edge connectors in the edge connector pack to the ordinary pack. See Figure 4 below.

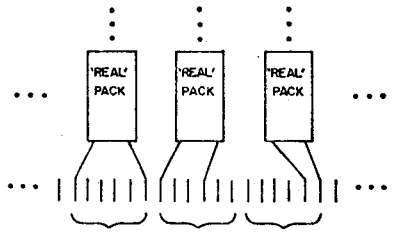


FIG. 4 EDGE CONNECTOR 'PACKS'

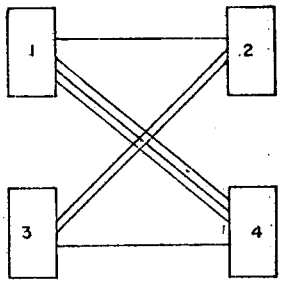
For a given pack A, let n(A) denote the number of locations which it must be assigned in any admissible placement. Then define the connection cost between packs A and B to be

$$c(A,B) = PC(A,B)/n(A)\cdot n(B)$$

where PC(A,B) denotes the number of primary connections between A and B. Then for a given admissible placement f:L → P, define the total pack connection cost by

$$TPC(f) = \Sigma\, c(f(l),f(m)) \cdot d(l,m)$$

where the sum is over all pairs of locations l and m. Then the objective is to find the admissible placement f which minimizes TPC(f). See Figure 5 below for an example.



TPC = 1·1 + 1·1 + 2·2 + 2·3 = 12

FIG. 5 PACK CONNECTION COST

III. THE HEURISTIC METHODS OF SOLUTION

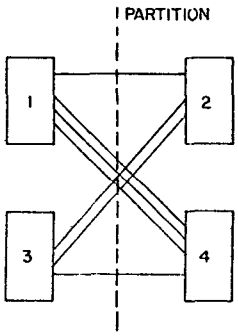
In the beginning literature-search stage of this project, we gradually settled on a Kernighan-Lin (KL) algorithm-type approach to the problem of pack placement. Simultaneously, a few brute force attacks on gate assignment were tried and rejected. We briefly

considered using KL on gate assignment but then recalled the obvious fact that two gates with different part numbers cannot be "exchanged". Thus instead of considering pairwise exchanges among the entire set of gates, one need only consider all gates with a given part number at any one time. A big problem is reduced to a series of small problems. Since KL is designed for the big problems, we settled on a more naive approach to gate assignment. For lack of a better name, we will call this a Random Pair Search (RPS) algorithm.

This development led to a modification in our approach to pack placement. Why not use KL to partition the packs into clusters relatively unconnected to each other, and then proceed to use RPS on each of the clusters, one at a time? This we tried, but then found that by skipping KL and going straight to RPS applied to the whole board, we could get better results in a shorter time.

Thus the whole process has evolved in a way we had not foreseen, and is in fact still evolving as experiments continue.

The Kernighan-Lin algorithm has been well discussed in the literature [1,2,4], so we will give only a brief description here of our particular version. First the packs are partitioned into two (non-empty) groups. Then a partition cost is computed, which is simply the total number of primary connections between packs on opposite sides of the partition (in short, which are cut by the partition). See Figure 6 below.



PARTITION COST = 7

TPC = 1·1+1·1+2·2+2·3 = 12

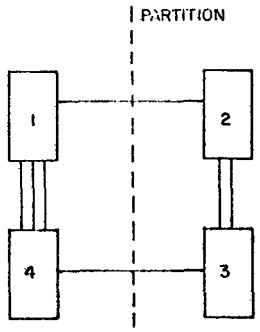
FIG. 6 PACK CONNECTION COSTS

Then in some predetermined order, all pairs of packs separated by the partition are considered for interchange. Any pair whose exchange would result in an inadmissible placement is rejected.

If a pair of packs is exchangeable, the gain in partition cost ((old partition cost) - (new partition cost)) which would result from the exchange is noted. The pair yielding the greatest gain is marked for interchange. Then a search begins for a new exchangeable pair of packs whose exchange would result in the greatest gain given that the previous pair has been interchanged. This pair is then also marked for interchange and the gain noted. This process continues until all packs are either fixed or marked for interchange. Suppose there are N pairs (P(I), Q(I)) of packs marked for interchange and let G(I) denote the gain which would result from P(I) and Q(I) being exchanged. Call this Phase I.

The second phase of KL starts by determining the integer K, $1 \le K \le N$, such that $\Sigma G(I)$ is a maximum, where the sum is over all I from 1 to K.

Then packs P(I) and Q(I) are exchanged for I=1,....K (our particular version makes no exchanges unless the partial sum of gains is positive for some K=1,.....N). See Figure 7 for an example of partition cost reduction by pairwise interchange. Note that TPC is reduced also.



PARTITION COST = 2

TPC COST = 1·1+2·1+1·1+3·1 = 7

FIG. 7 PACK CONNECTION COST REDUCTION

Now a new partition is chosen and the process begins anew. Each partition, however, adds new criteria to admissibility for later placements, namely, no two packs separated by a previous partition can be exchanged (with each other). In this way the clusters are formed. For example, in figure 8 below, the clusters are 1,4 and 2,3.

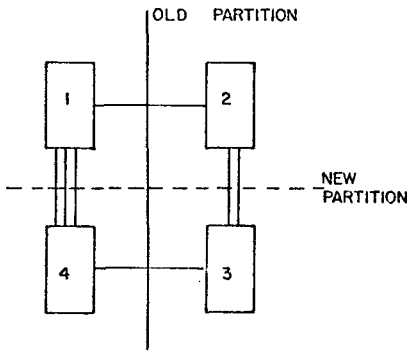


FIG. 8 CLUSTER FORMATION

We note that a partition cost reduction is not always accompanied by a corresponding decrease in TPC. See Figures 9 and 10.

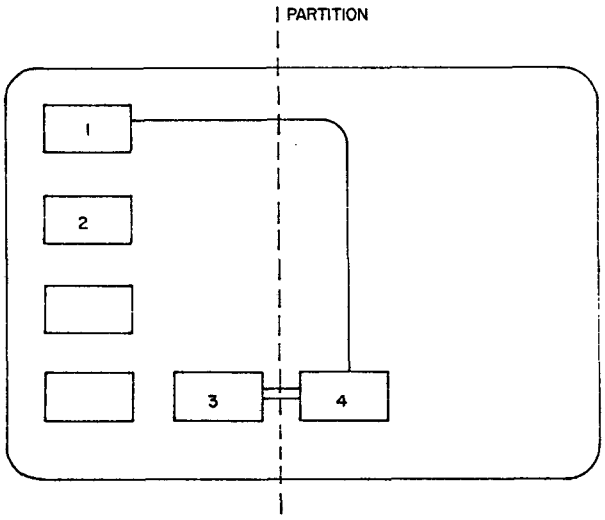


FIG. 9 CONNECTION COST EXAMPLE

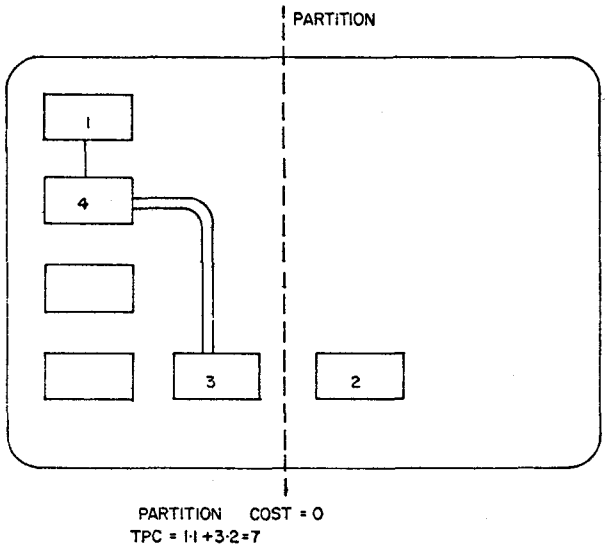


FIG. 10 CONNECTION COST EXAMPLE

See Figures 11 and 12 at the end of the paper for flow charts of the two phases of KL.

We now describe the operation of the Random Pair Search algorithm. As mentioned before, this algorithm is applied to both problems. In the gate assignment problem, the input to the algorithm is a set of gates with the same part number. In the pack placement problem, the input is a cluster of packs. This cluster may be a proper subset of the set of packs produced by a pass of KL, or may be the entire set of packs. In the description to follow, terminology appropriate to both problems will be used. That applying to gate assignment will appear in parentheses.

As its name implies, the RPS algorithm starts by randomly choosing a sequence of pairs of packs (gates) from the current cluster (set of gates with current part number). This is done as follows. If there are $N \geq 2$ packs (gates) in the collection under consideration, label the packs (gates) with the integers 1,..N- at random. (This can be done by using a (pseudo) random number generator to simulate the uniform distribution on the set of all ordered N-sequences of distinct integers from 1,....,N). This labelling is independent of any other labelling the packs (gates) may have for different purposes. With respect to this labelling, all pairs are considered one by one in the following order: (1,2), (1,3), (2,3), (1,4), (2,4), (3,4), (1,5),... . To be more precise (1,2) is the first pair, and if (I,J) is currently under consideration the next is

$$(I + 1, J) \quad \text{IF } I < J - 1,$$

$$(1, J + 1) \quad \text{IF } I = J - 1 \text{ and } J < N.$$

and the process stops otherwise.

A pairwise exchange is performed if and only if it results in a reduction in the total pack connection cost (total gate connection cost) for the current cluster (set of gates with the current part number). After an exchange is performed, the connection costs are updated and the determination of any further gains is based on the new placement (assignment) and connection costs.

Refer to Figure 13 at the end for a flow chart of the algorithm. The phase "select a pair of packs at random" in the first box is to be understood in the sense described above. Not all possible sequences of pairs of packs (gates) are equally likely under our scheme. Most, in fact, have probability zero of being chosen.

IV. IMPLEMENTATION

The whole process begins with a logical schematic, from which the relevant data is manually extracted. This includes a numbering of all the signals, edge connectors and gates, and the part numbers of the latter. Of course all the interconnect information must be entered in order to compute costs and any functionally dependent gates must be identified in order to determine admissible assignments.

This information is then pre-processed to produce three files which contain, respectively, an initial arbitrary admissible assignment, the "class" numbers of the gates, and all the gate connection costs. Since all our algorithms are based on pairwise interchanges, the class number file is simply a convenient device for guaranteeing that no inadmissible assignments are produced. This is done by allowing a gate exchange if and only if the gates in question have the same class number.

This information is then input to the main gate assignment program ASSIGN. This program is intended to be used interactively by a PCB designer, and therefore has the following features. The program proceeds automatically through the part numbers one by one. The gates with the current part number are displayed to the user, together with their class numbers. If no exchanges are possible the user is so informed and the program proceeds automatically to the next part number.

If exchanges are possible, the user can choose either a manual or automatic mode. In the manual mode, the exchange pairs are chosen by the user. He is automatically informed of any impossible exchanges and of the gains to be realized from permitted ones. Any or all of the permitted exchanges are performed at the direction of the user, whether the gains are positive or not (unlike the automatic mode). See Figure 14 at the end of this paper for a flow chart of the routine for user-chosen exchanges.

After any exchange(s), the user can request a display of the updated assignment and connection cost information for the current part number. He can then request more exchanges of his own choosing or proceed to the automatic mode, which is of course based on the RPS algorithm previously described. Again, after a pass through computer-chosen exchanges, an updated display of gate assignment and costs can be requested. The user can proceed through a sequence of user-chosen and computer-chosen exchanges in any combination for as long as he wants. When he is satisfied with the connection cost reduction, he simply requests to go to the next part number and the process begins anew.

The output of ASSIGN is a file giving the updated gate assignment. This assignment plus the original interconnection and edge connector "pack" information is then pre-processed to produce an initial arbitrary admissible pack placement together with class numbers and connection costs for the packs. This is input to the program PARK, which is based on the Kernighan-Lin algorithm. The user chooses the partitions for this routine, but once one is chosen, its operation is automatic.

After a series of exchanges is completed, the user can request a display of the updated placement, partition cost and total pack connection cost. The options then are to request more exchanges or to proceed to a new partition. When the costs are reduced to user satisfaction, the program is halted.

The output of PARK is a revised pack placement together with a collection of clusters of packs. This information, together with connection costs and class numbers is input to the program PLACE, which is based on the RPS algorithm.

The operation of PLACE is very similar to ASSIGN. The user is provided immediately with the current placement and total pack connection cost for the board. The routine then proceeds to output information about packs in the first cluster, such as class numbers and total connection cost for packs in the cluster. Again, either a manual or automatic mode can be selected.

The operation and interplay of these is exactly as described for gate assignment. In addition to being constantly updated on the placement and cluster connection cost, the user is also informed of the total pack connection cost for the entire board upon request.

After proceeding through all the clusters, the program halts. At this point one can take the updated placement, input it to PARK, and begin the whole process anew. This can proceed as long as resources allow or until the user is satisfied.

As mentioned previously, it is also possible to skip PARK entirely and proceed directly to PLACE with only one cluster, namely, the whole board.

V. RESULTS

The programs to implement the KL and RPS algorithms were coded in Fortran and run on a CDC Cyber 176. The program for gate assignment is called ASSIGN, while the two for pack placement are named PARK and PLACE, based on the KL and RPS algorithms, respectively. They have been tested on two boards, one of which is in production (board #1) and another which is concurrently being designed manually. See Table 1 for the relevant data. Note that the manual layout for board #1 has one more 'real' pack (and therefore one less 'dummy') than the starting assignment provided by our pre-processor routines. The human designer decided to use three packs to contain a certain collection of gates which would have fit into two; our pre-processor routine did not. Note also that the second board is significantly more dense than the first.

Results from three runs of ASSIGN on board #1 can be seen in Table 2. These results were obtained by using ASSIGN as follows. No user-chosen exchanges were made; the program was run entirely in the 'random' mode. Therefore, the following (arbitrary) rule was adopted in the running of the routine for the gates of a given part number: as soon as a sequence of pairs of gates is chosen such that no pair exchange would yield a gain in total connection cost, the program is instructed to proceed to the next part number; otherwise attempt another sequence of exchanges. It should be noted also that assignments #1 and #2 resulted from using an arbitrary initial assignment, while #3 started with #2 as the initial assignment.

There are similar remarks to be made about the conditions under which the results for pack placement were achieved. See Table 3 for the outcome for board #1. In every run of PARK, two partitions were always chosen, one horizontally through the middle of the board (between row 5 and 6) followed by one vertically through the middle of the board (between columns 9 and 10). In every run of PLACE, as soon as a sequence of pairs of packs (in a given cluster) is chosen so that no pair exchange would result in a gain in total intracluster connection cost, the program is instructed to proceed to the next cluster; otherwise choose another sequence of exchanges.

Several remarks should be made about table 3. First, a run of 'PARK, PLACE' means that PARK and PLACE are run in that order, with PARK operating on the 'starting placement' and PLACE subsequently using the resulting placement as its input. Any run of PLACE following a run of PARK operates on 4 clusters. Isolated runs of PLACE operate on 1 cluster, namely the whole board at once. A similar remark holds for any run with more than one pass. It is to be understood that the output placement of any pass is the input to the next. Thus are the placements successively improved and the connection cost reduced. Note also that 4 pairs or runs are grouped together: 1 and 2, 3 and 4, 10 and 11, and 12 and 13. The input placement to the second run in each pair is exactly the output of the first.

Also there are three different gate assignments involved: #1 and #3 from Table 2 and the manual assignment. Recall from Table 2 that assignment #3 has a higher gate connection cost than #1. It is therefore reassuring to note that, given the same arbitrary initial placement, the total pack connection costs for the two assignments are similarily related (18093 for #2 vs. 17664 for #1).

We now make some observations about these results.

(1) In one run of PARK (#12), the total pack connection cost actually increased, even though

partition cut costs were reduced considerably. This phenomenon occurred two other times (runs #2 and #4) although the loss on the first partition was gained back on the second. This seems to occur only when the connection cost is already fairly low. Note the considerable gain provided by the single run #5 of PARK.

(2) Any single run of PLACE seems to reduce connection cost more than any single PARK-PLACE combination, and in not much more time.

(3) The very best result (run #14) was a run starting with the manual, rather than arbitrary, initial layout. In fact, run #15, which was really the first pass of run #14, resulted in a connection cost almost as good as run #9, which required 4 passes to achieve (and considerably more time).

(4) The total pack connection cost for the manual layout is 11239 (see runs #12, 14, and 15), which is of course much better than the cost of 17664 from the arbitrary initial placement but not as good as that provided by even a single 10 minute run of PLACE on that initial placement. On the other hand, as noted in (3) above, the best outcome resulted from starting with the manual layout. Without more experimentation, it is impossible to say whether results as good as this 'best' can be achieved in a reasonable amount of time by several iterations of PLACE, starting with an arbitrary initial assignment.

The results for board #2 may be seen in Tables 4 and 5. We note the following about the outcome.

(1) The greater density of board #2 is reflected in much greater total gate and pack connection costs than for board #1. Also, the run times for the pack placement routines were much greater, yet those for gate assignment were less. This latter fact was due to the fact that there were very few admissible gate assignments for board #2. Note the almost insignificant gains in gate connection cost.

(2) Run #1 of PARK (Table 5) used two partitions, one horizontal and one vertical, both through the center of the board. Run #2 of PLACE began with the placement produced by run #1 of PARK. PLACE therefore operated once on each of the four clusters produced by PARK. On the other hand run #3 of PLACE began with the initial arbitrary placement and operated on one cluster only (the whole board). Note that the (combined) gain produced by the PARK-PLACE duo was greater than that produced by a single run of PLACE, but the computation time was significantly higher.

We repeat our earlier statement that more experiments need to be done to draw any firm conclusions here. For example, it would be interesting to see what another run of PLACE based on the output of run #3 would produce. Also, it might improve the results for PARK-PLACE combinations in different partitions were tried in the PARK segment. Unfortunately, time has not permitted these and other experiments for inclusion in this paper. Tentatively, we would say that single applications of RPS seem to be more cost effective (in terms of CPU time) than combined treatments of KL followed by RPS.

BIBLIOGRAPHY

1) Breuer, M.A., "A Class of Min-Cut Placement Algorithms", Proceedings of 14th. Design Automation Conference, 1977.

2) Corrigan, L.I., "A Placement Capability Based On Partitioning", Proceedings of 16th. Design Automation Conference, 1979, pp. 406-413.

3) Hanan, Wolff and Agule, "Some Experimental Results On Placement Techniques", Proceedings of 13th. Design Automation Conference, 1976.

4) Kernighan, B.W. and Lin, S., "An Efficient Heuristic Procedure For Partitioning Graphs", Bell System Technical Journal, Vol. 49, February 1970, pp. 291-307.

5) Nishioka, Jurimota, Yamamota, Ozaki, Shirakaw, "An Approach to Gate Assignment and Module Placement for Printed Wiring Boards", Proceedings of 15th. Design Automation Conference, 1978, pp. 60-69.

PROGRAM PARK (Phase I)
BASED ON KERNIGHAN-LIN ALGORITHM

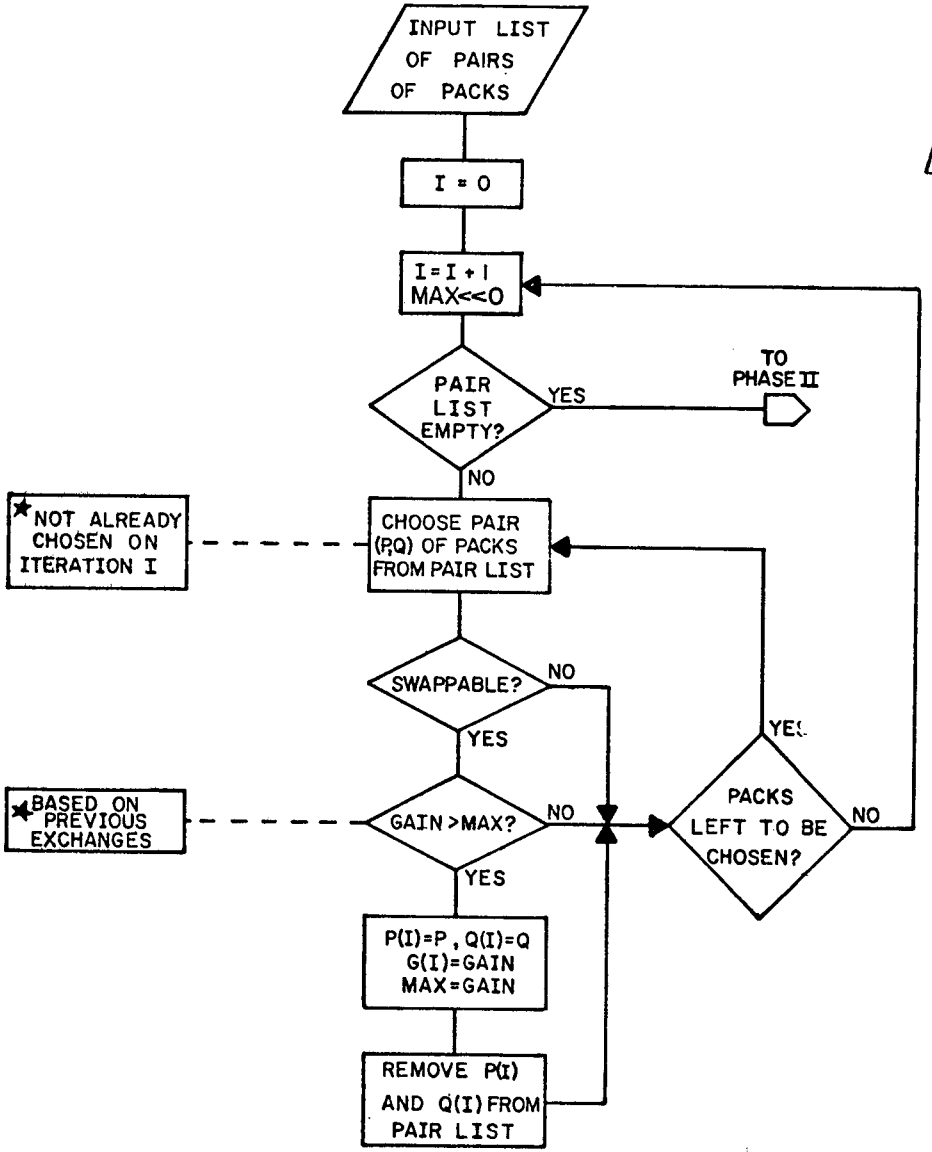


FIG. 11

PROGRAM PARK (PHASE II)
BASED ON KERNIGHAN-LIN ALGORITHM

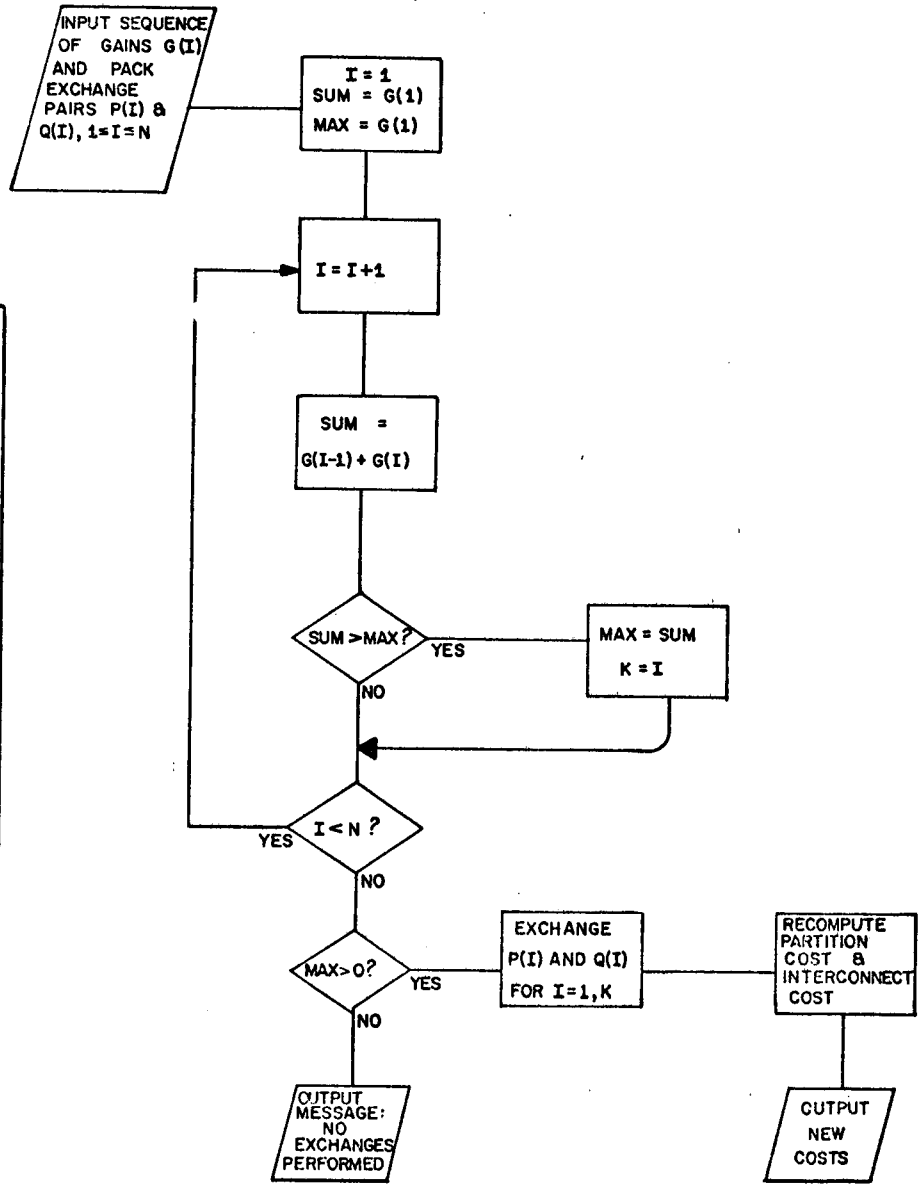


FIG. 12

PROGRAM PLACE

RANDOM EXCHANGES

(Cluster is fixed throughout)

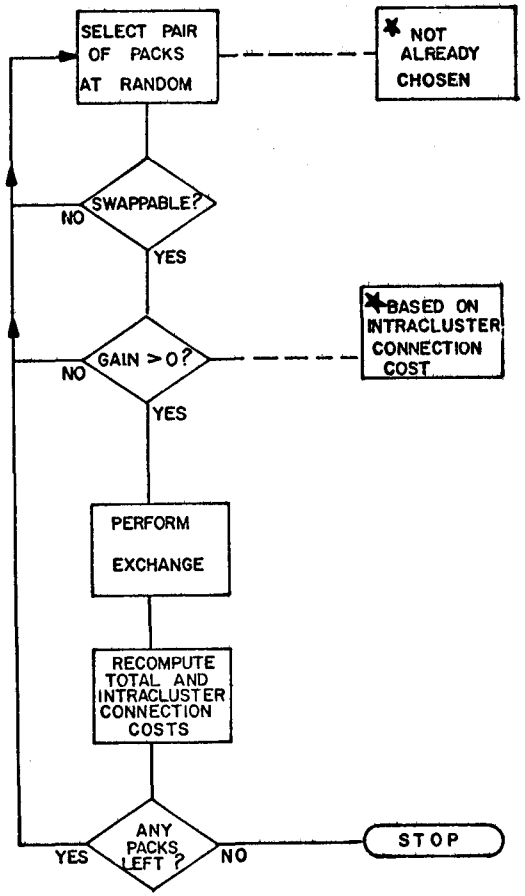


FIG. 13

PROGRAM ASSIGN

USER CHOSEN EXCHANGES

(PART NUMBER CONSTANT)

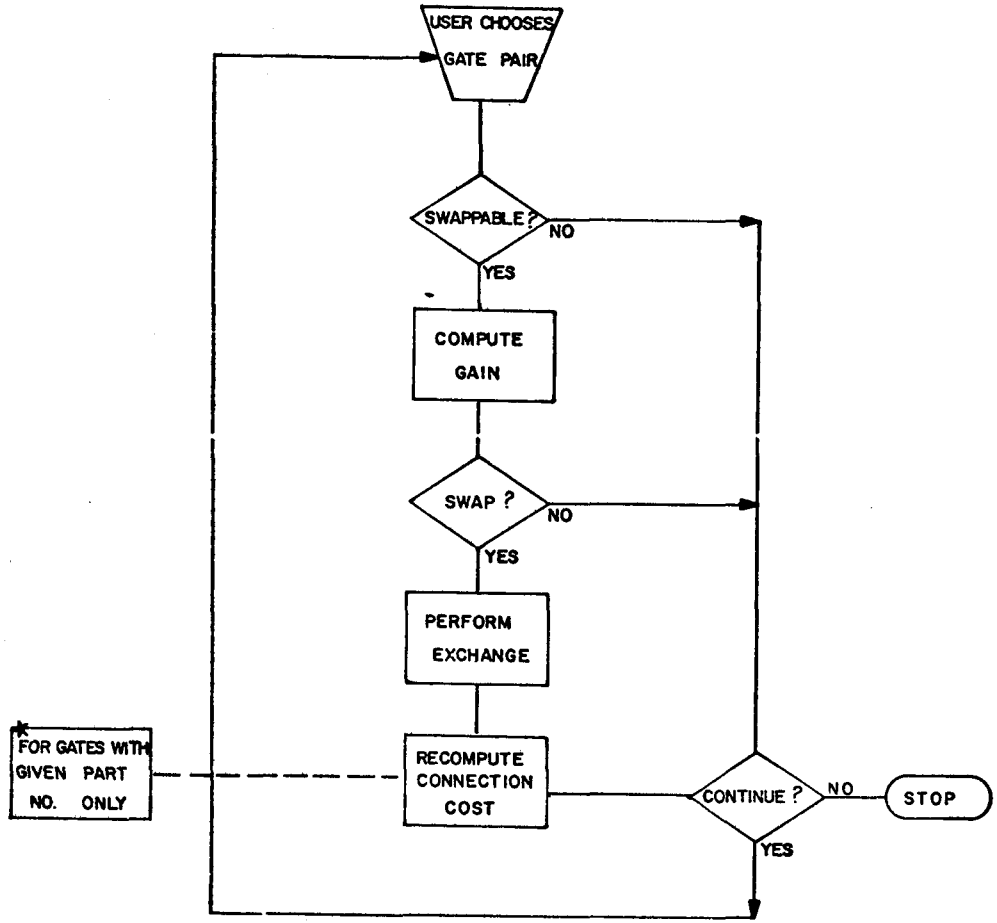


FIG. 14

	BOARD #1	BOARD #1 (MANUAL LAYOUT)	BOARD #2
SIZE	11" X 14"	11" X 14"	11" X 14"
GATES	692	692	878
SIGNALS	604	604	780
EDGE CONNECTORS	204	204	204
'REAL' PACKS	138	139	167
'DUMMY' PACKS	6	5	20
'EDGE CONNECTOR' PACKS	18	18	24
TOTAL PACKS	162	162	211
LOCATION ARRAY	9 ROWS X 18 COLUMNS	9 ROWS X 18 COLUMNS	10 ROWS X 24 COLUMNS

TABLE I BOARD DATA

	INITIAL CONNECTION COST	FINAL CONNECTION COST	% IMPROVEMENT	CPU TIME
ASSIGNMENT # 1	6288	5700	9.4%	N.A.
ASSIGNMENT #2	6288	5761	8.4%	2 min. 3 sec.
ASSIGNMENT #3	5761	5711	.87%	2 min. 1 sec.

TABLE 2 GATE ASSIGNMENT: BOARD NO.1

	RUN	NUMBER OF PASSES	GATE ASSIGNMENT	STARTING PLACEMENT	INITIAL CONNECTION COST	FINAL CONNECTION COST	GAIN	CPU TIME
1	PARK, PLACE	1	#1	ARBITRARY	17664	10674	6990	9 min. 3 sec.
2	PARK, PLACE	1	#1	FINAL FROM ABOVE	10674	10119	555	9 min. 0 sec.
3	PARK, PLACE	1	#1	ARBITRARY	17664	10355	7309	12 min. 11 sec.
4	PARK, PLACE	1	#1	FINAL FROM ABOVE	10355	10080	275	9 min. 11 sec.
5	PARK	1	#1	ARBITRARY	17664	12995	4669	7 min. 56 sec.
6	PLACE	1	#1	ARBITRARY	17664	10207	7457	9 min. 54 sec.
7	PLACE	1	#1	ARBITRARY	17664	10015	7649	10 min. 5 sec.
8	PLACE	1	#1	ARBITRARY	17664	10116	7548	9 min. 52 sec.
9	PLACE	4	#1	ARBITRARY	17664	9108	8556	39 min. 0 sec.
10	PLACE	1	#3	ARBITRARY	18093	10285	7808	9 min. 47 sec.
11	PLACE	1	#3	FINAL FROM ABOVE	10285	9142	1143	9 min. 47 sec.
12	PARK	1	MANUAL	MANUAL	11239	12072	-833	7 min. 58 sec.
13	PLACE	1	MANUAL	FINAL FROM ABOVE	12072	10251	1821	4 min. 8 sec.
14	PLACE	4	MANUAL	MANUAL	11239	8444	2795	38 min. 37 sec.
15	PLACE	1	MANUAL	MANUAL	11239	9187	2052	N.A.

TABLE 3 PACK PLACEMENT: BOARD NO.1

	INITIAL CONNECTION COST	FINAL CONNECTION COST	% IMPROVEMENT	CPU TIME
ASSIGNMENT #1	68899	68641	.37%	N.A.
ASSIGNMENT #2	68641	68641	0%	1 min. 2 sec.
ASSIGNMENT #3	68899	68682	.31%	1 min. 6 sec.

TABLE 4 GATE ASSIGNMENT: BOARD NO. 2

RUN	NUMBER OF PASSES	GATE ASSIGNMENT	STARTING PLACEMENT	INITIAL CONNECTION COST	FINAL CONNECTION COST	GAIN	CPU TIME
1 PARK	1	#1	ARBITRARY	39108	28971	10137	24 min. 38 sec.
2 PLACE	1	#1	FINAL FROM ABOVE	28971	20600	8371	19 min. 22 sec.
3 PLACE	1	#1	ARBITRARY	39108	21405	17703	20 min. 39 sec.

TABLE 5 PACK PLACEMENT: BOARD NO. 2