NASA CONTRACTOR
REPORT

NASA CR-149966

(NASA-CR-149966) PRINTED WIRING BOARD SYSTEM PROGRAMMER'S MANUAL (M&S Computing, Inc., Huntsville, Ala.) 80 p HC $5.00 CSCL 09A G3/33

N76-33405

Unclas 05316

PRINTED WIRING BOARD SYSTEM PROGRAMMER'S MANUAL

By C. D. Brinkerhoff
M&S Computing, Inc.
Post Office Box 5183
Huntsville, Alabama 35805

April 25, 1973

Prepared for
NASA - GEORGE C. MARSHALL SPACE FLIGHT CENTER
Marshall Space Flight Center, Alabama 35812

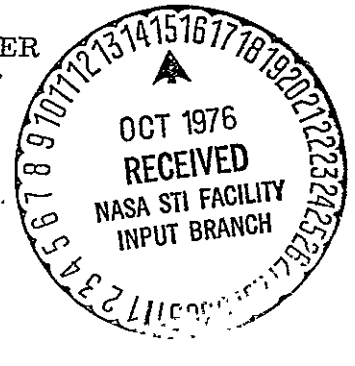


TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

1. PRINTED WIRING BOARD SYSTEM

The Printed Wiring Board System (PWB) provides automated techniques for the design of printed circuit boards and hybrid circuit boards. The four design automation programs which comprise the PWB System are:

- o Preprocessor
- o Placement
- o Organizer
- o Router

The programs will normally be executed in the order indicated with each program requiring the successful execution of the previous program. Figure 1-1 illustrates the program execution sequence, input sources, and data outputs.

The Preprocessor Program combines user supplied data and predefined library data to produce detailed input for the Placement Program. The user supplied data consists of a list of the components to be placed on the board, a list of the desired connections between components, and a board identification. The library data provides physical size and pin placement data as well as a description of the board area and pin logic information. The Preprocessor output file is the Placement Input File (PIF).

The Placement Program is designed to automatically assign components to specific areas of the board. The program attempts to optimize the placement solution using the interconnection characteristics between components. A Pin Connection File (PCF) is generated and passed to the Organizer Program.

The Organizer Program converts the PCF logic string data into a list of pin pairs and determines the board layer on which each pin pair should be placed. The program also determines the optimal order in which the Router Program should attempt to layout the paths connecting the pin pairs.

The Router Program attempts to determine the wire paths which are to be used to connect each input pin pair on the circuit board. The available wire path area for each board layer is described on the Input Cover File which was generated by the Organizer Program. The pin pairs to be connected by the program are input from the Router Batch Input File (RBI) which was also generated by the Organizer Program. A printed image of the wire paths on each board layer and an artwork (RBO) file are generated as part of the program output. An optional

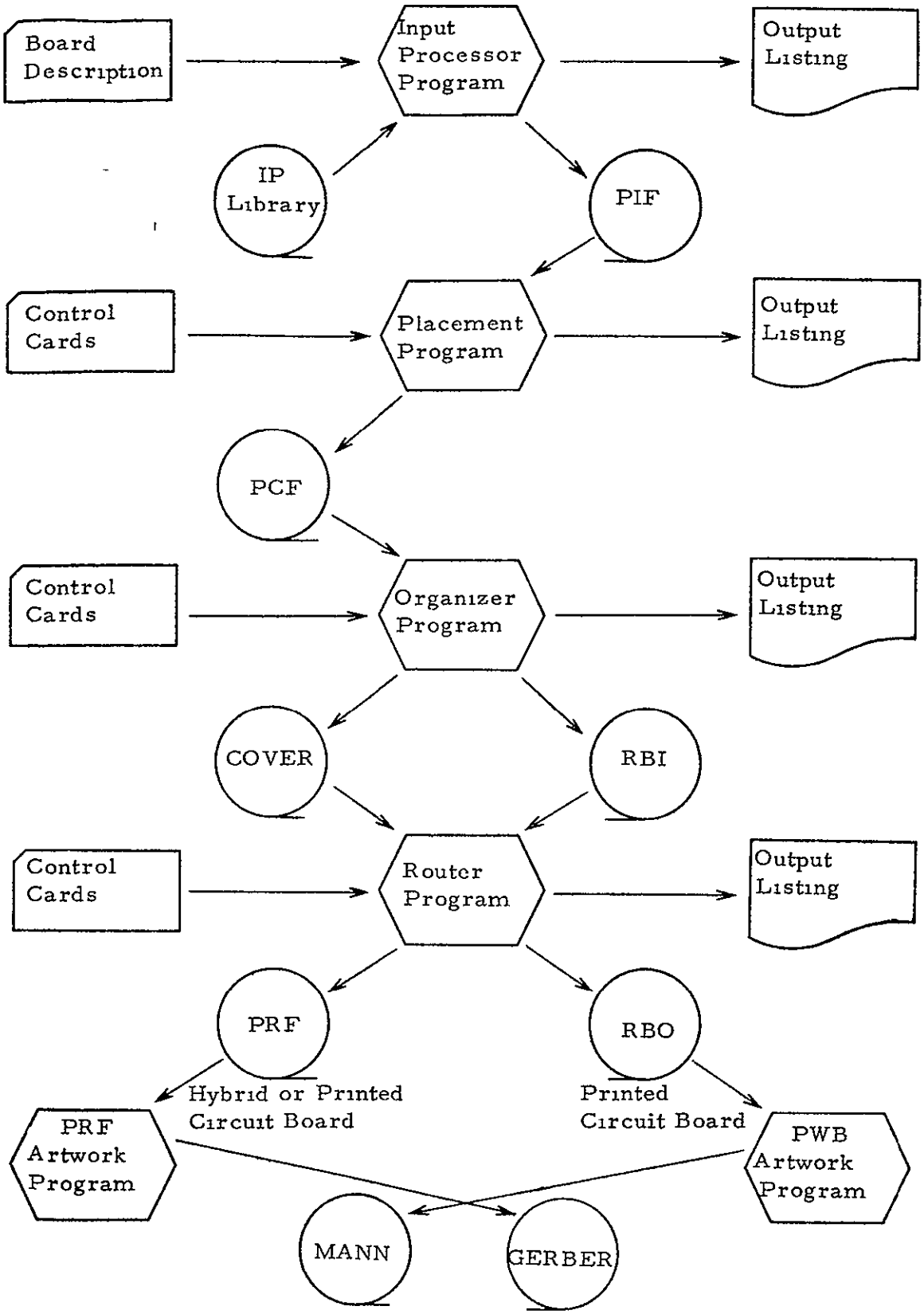

Board Description
Input Processor Program
Output Listing
IP Library
PIF
Control Cards
Placement Program
Output Listing
PCF
Control Cards
Organizer Program
Output Listing
COVER
RBI
Control Cards
Router Program
Output Listing
PRF
RBO
Hybrid or Printed Circuit Board
Printed Circuit Board
PRF Artwork Program
PWB Artwork Program
MANN
GERBER

Figure 1-1

artwork tape (PRF) for hybrid dielectric isolation boards or printed circuit boards may also be generated.

In the terminology of the PWB System the circuit board is a rectangular board onto which specific components are to be mounted. The programs are capable of processing boards that have from two to sixteen layers. The term module is used in a generic sense in referring to any component which is to be mounted on the board. Package is used in referring to the case or physical housing of a component. Package type refers to a particular type of packaging, i.e. 14 pin dual-in-line, 16 pin flat pack, TO-5, etc., of specific dimensions and pin arrangement. Only the dimensions which reflect how much surface area of the board, in the form of a rectangle, a package will occupy are required. The arrangement of the pins on a package is defined as a pin array and is described in Cartesian coordinates relative to the lower left corner of the package area. In substrate boards the term package is extended to mean the pad area or cell from which intermodule connections are made.

A specific module type defines the pin logic of a module as well as the package type. A logical element of a module is defined as a pin or group of pins which perform an independent function. The elements of the module in Figure 1-2 are as follows:

Element	Pins
A	1, 2, 3
B	11, 12, 13
C	4, 5, 6
D	8, 9, 10

The Placement Program will optionally permute elements of the same module or different modules of the same type in attempting to find an optimal placement solution. Module pins which are logically equivalent may be assigned the same swap tag so that the Organizer Program may interchange connections to optimize results. The pins in Figure 1-2 which have the same swap tags are·

Swap Tag	Pins
F	1, 2
G	4, 5

PRINTED CIRCUIT BOARD MODULE

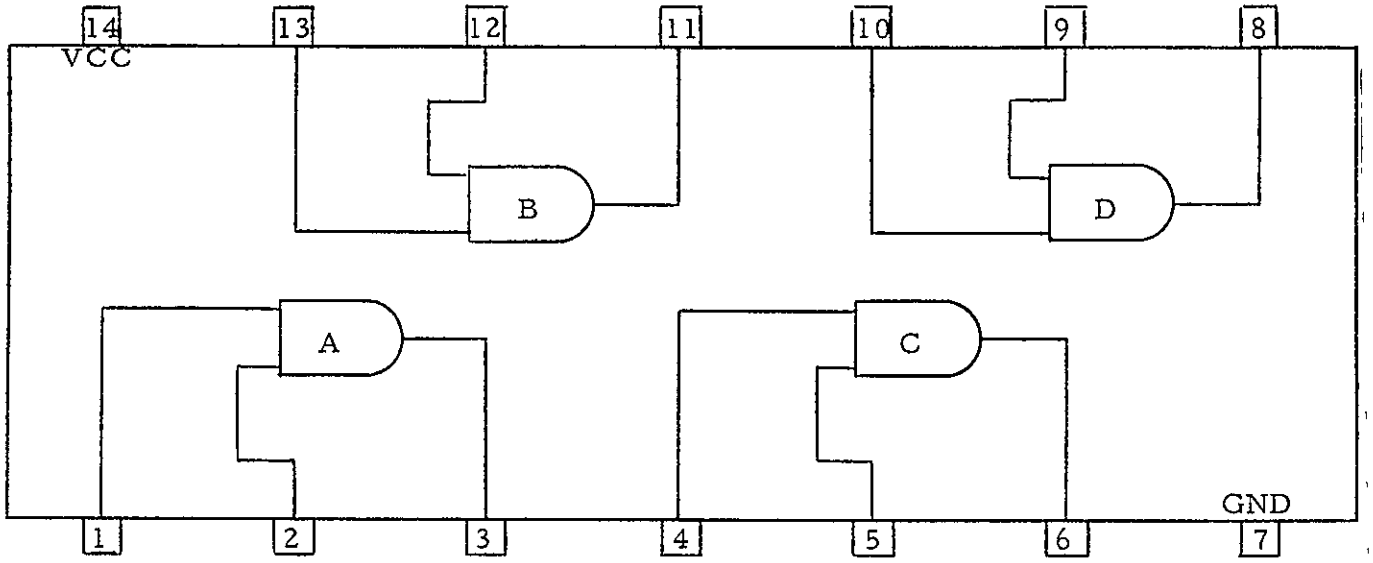


Figure 1-2

Swap Tag	Pins
H	9,10
I	12,13

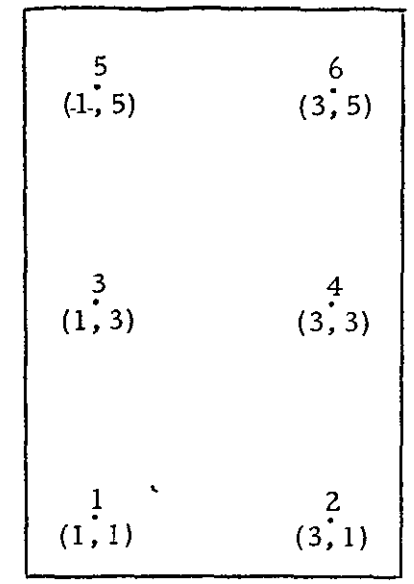
The source tag associated with a pin indicates the pin type and is used by the Placement Program in pairing pins for connection.

A logic string is a list of pins that represent a contiguous path between modules. Each pin of a logic string has the same signal name associated with it.

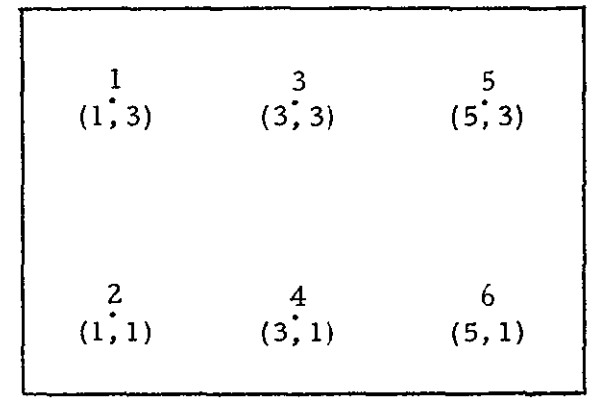
An area of the board in which a module is mounted is called a socket, and a particular socket may accept only one type of package. The positioning of sockets on the board and the distance between sockets are very important considerations for the user. Sockets may be pre-placed or the user may optionally request the Preprocessor Program to generate the necessary socket positions. The socket allocation option has some restrictions however, and at least one socket must be pre-placed.

The user may interchange and/or rotate the pin positions on a package by use of the orientation parameters. The eight orientations that a package may be assigned are shown in Figure 1-3.

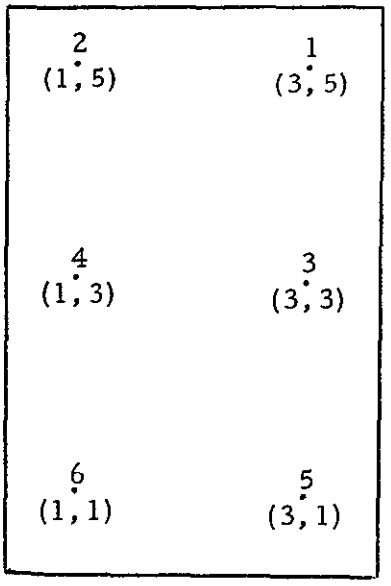
ORIENTATION PARAMETERS



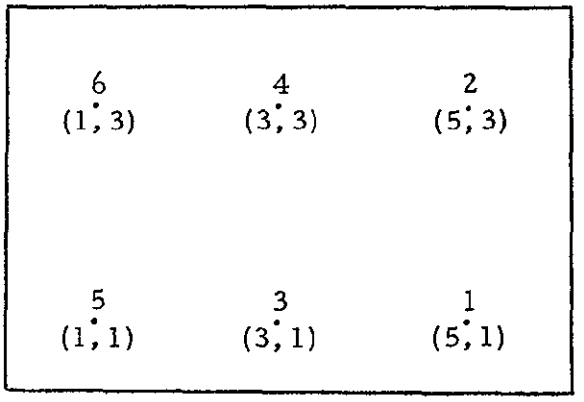
0



1

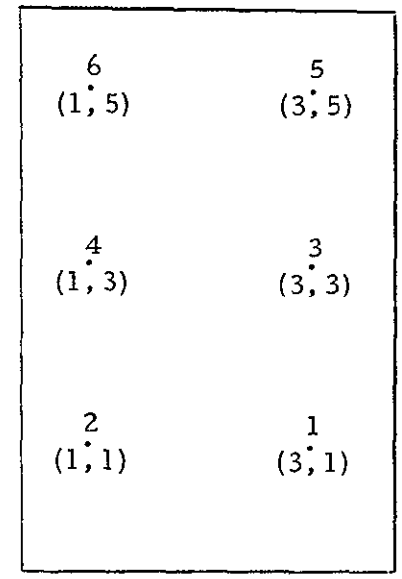


2

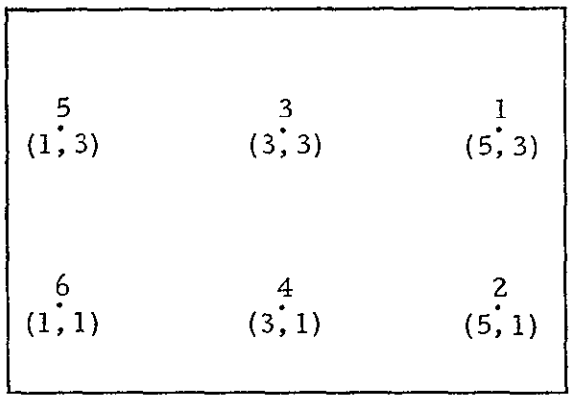


3

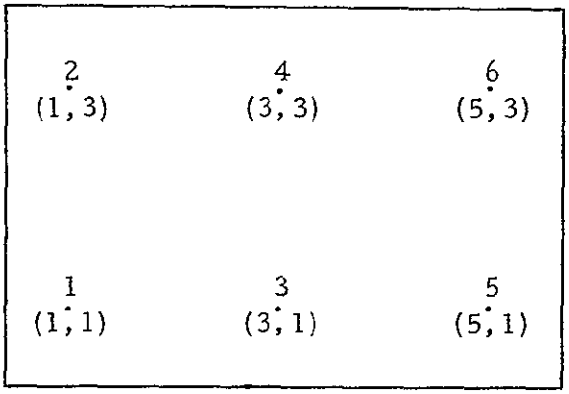
Figure 1-3



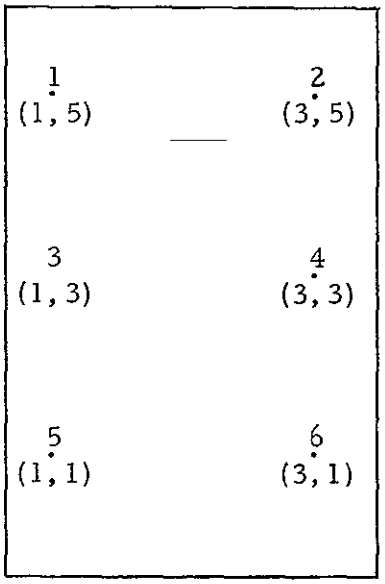
4



5



6



7

Figure 1-3
(continued)

BLANK

2. PREPROCESSOR PROGRAM

In the Placement, Organizer and Router Programs, the board, sockets on the board, connectors, and resistors are all described in grid units. A grid unit is the distance between centers of an imaginary grid placed over the board. Since the programs are unaware of distances, except through grid units, all modules to be placed on the board must be scaled to grid units. One function of the Preprocessor is to convert the mil specifications of the user to grid unit values.

All socket locations, as well as the pin locations with a socket, must also be specified in grid units. The position at which a socket is placed on the board and the distance between sockets are automatically calculated by the Preprocessor for all modules that are not pre-placed.

To describe each socket, the Preprocessor Program generates a POS card as part of the Placement Input File. The POS card data includes a unique name for the socket which will be used later to match the socket to the component placed in it, a pin array name which links the socket to a group of ARRY cards which indicate the pin pattern relative to the socket, the location of the socket (X, Y) on the board in grid units, the size of the socket in grid units, the board layer on which the socket is available, and an orientation parameter that allows the user to rotate and exchange pin locations in order to reduce redundancy.

To describe the various pin patterns present on the board, the Preprocessor generates a group of ARRY cards for each pattern. Each ARRY card contains a pin array name which indicates the pattern associated with the pin, a pin number, and the pin location (X, Y) in grid units relative to a reference point.

Once the board is described the Preprocessor Program generates a group of MLF cards which associates each component with a pin array name and a component type. One MLF card is generated for each component on the board and contains a name which uniquely identifies the component.

The final data cards generated by the Preprocessor Program are the PCF cards. These cards describe the connections which are to be made between the various components on the board.

2.1 Preprocessor Operation

Figures 2-1 through 2-14 depict the flow of operation of the Preprocessor Program. Figure 2-1 gives an overall view of operation while Figures 2-2 through 2-14 show a more detailed flow of the specific Preprocessor functions. As indicated by Figure 2-1 the Preprocessor

operation consists of the following eleven functions:

- CTLCRD - The Preprocessor input control cards are read and checked for errors. If a library update is requested the library processing routine is called.
- MODTYP - The library file is scanned and data for each module type used on the circuit board is read into memory.
- PACDAT - The library file is scanned and data for each package type used on the circuit board is read into memory.
- EPER - The data on the module types is scanned and element permute cards (EPER) are generated.
- PRESOC - The board dimension and pre-placed socket data is extracted from the library LBRD entry and stored in memory.
- SOCGEN - An algorithm is used to generate a number of sockets for each package type of the board modules which have not been assigned to pre-placed sockets by the user. An explanation of the algorithm used in socket generation is given in Section 2.2.
- POSGEN - POS cards are generated for each of the user defined and program generated sockets.
- LBARGEN - Any LBAR cards required to indicate barrier areas and MPOS cards required to restrict modules to pre-placed sockets are generated in this phase.
- MLFGEN - The module name, module type, and package type data are used to generate the Module Location File.
- PCFGEN - Logic strings are built from the user specified pin signal data and the Pin Connection File is generated.
- LIBPRT - If a listing of the Preprocessor Library is requested, it is generated at this time.

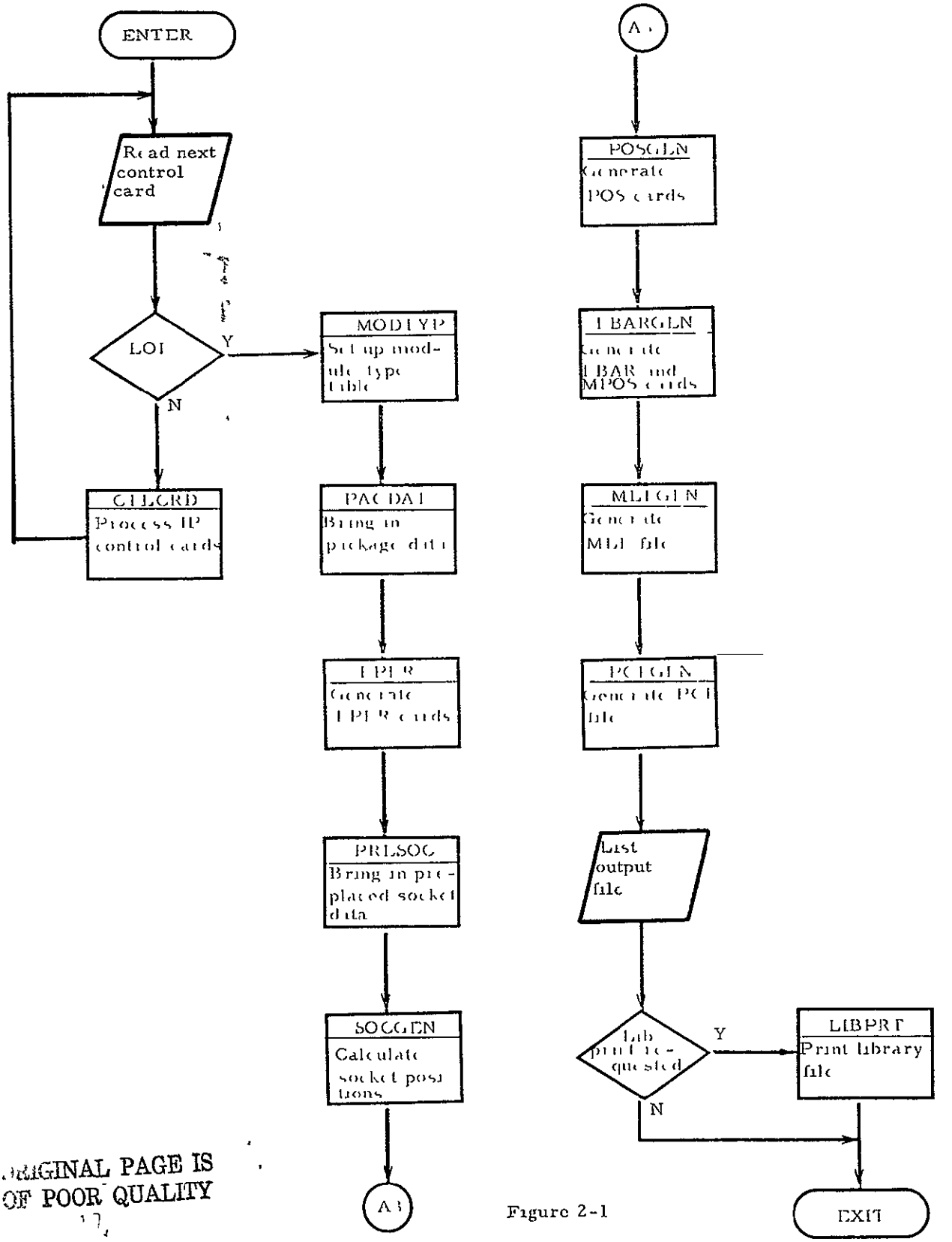


Figure 2-1

CONTROL CARD PROCESSING

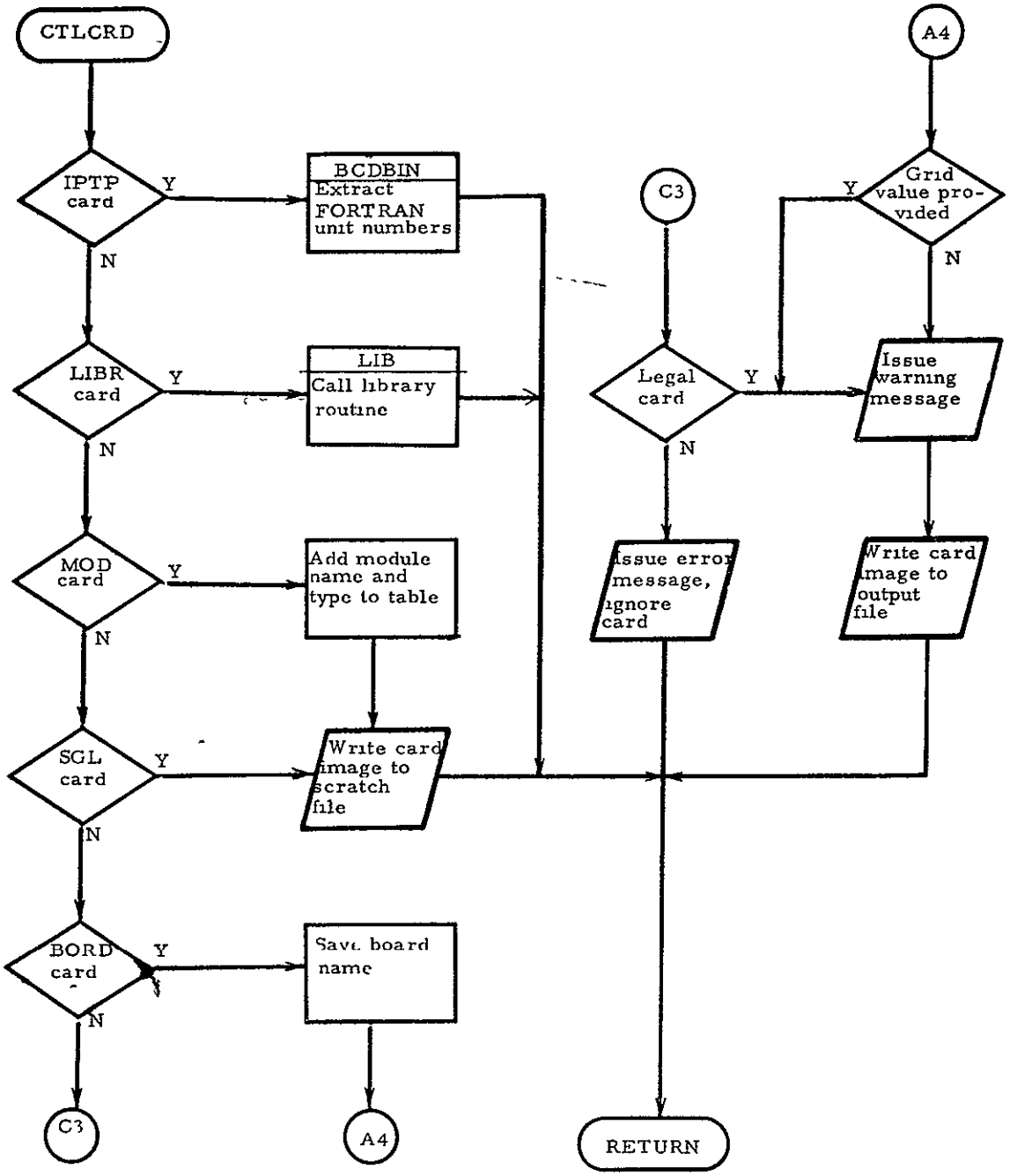


Figure 2-2

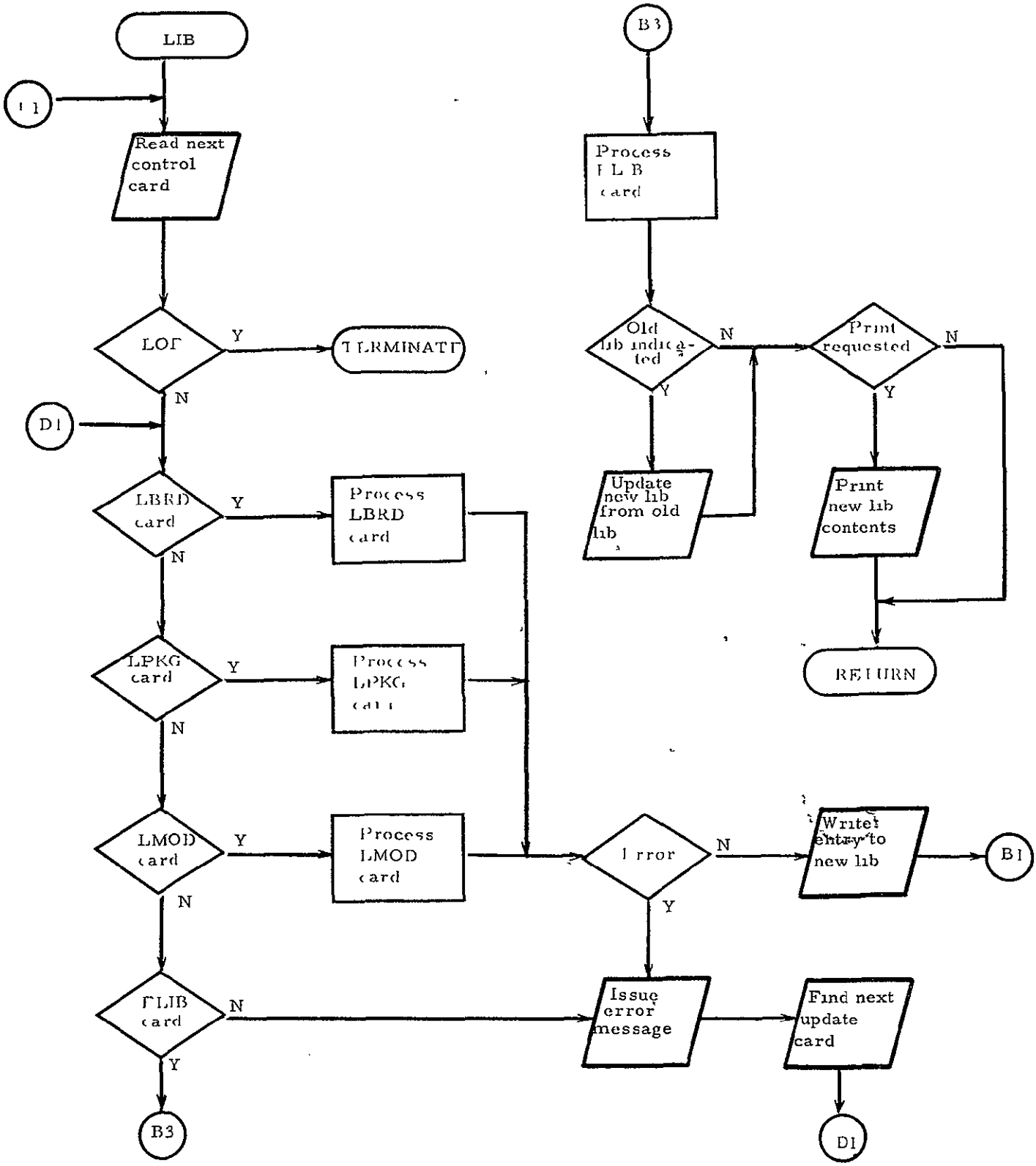


Figure 2-3

BCD TO BINARY PROCESSING

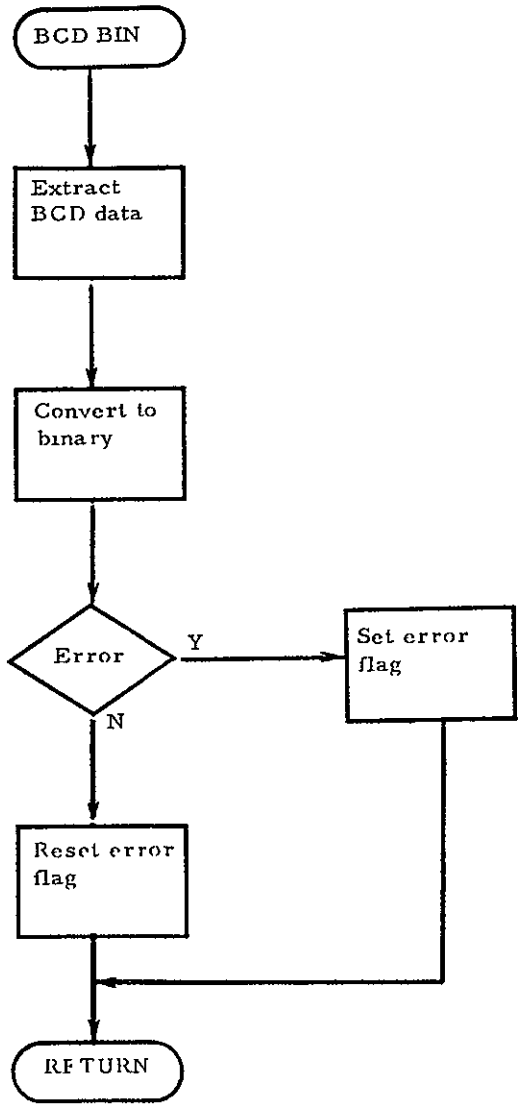


Figure 2-4

MODULE TYPE PROCESSING

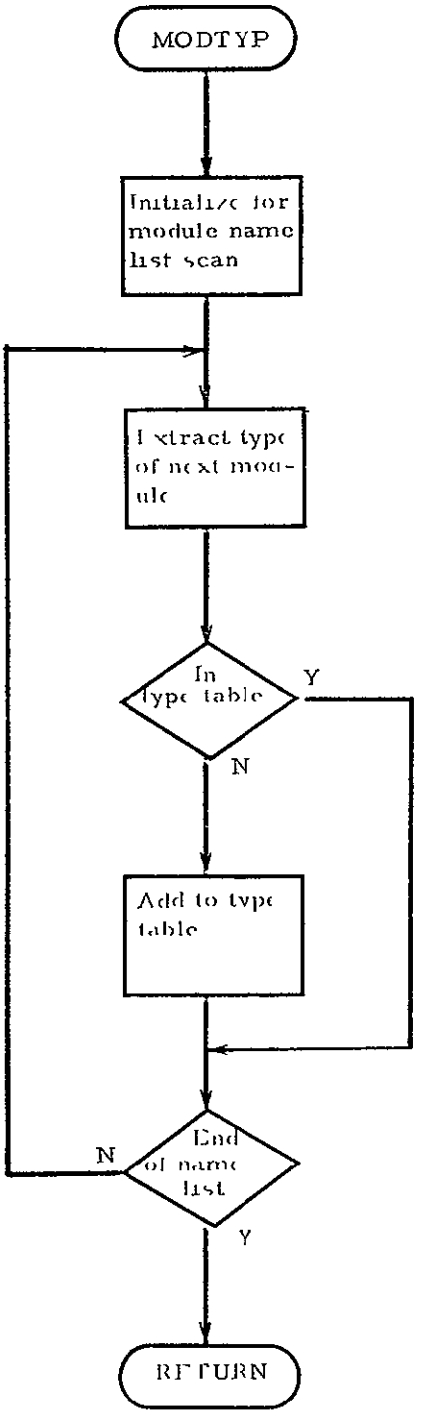


Figure 2-5

PACKAGE TYPE PROCESSING

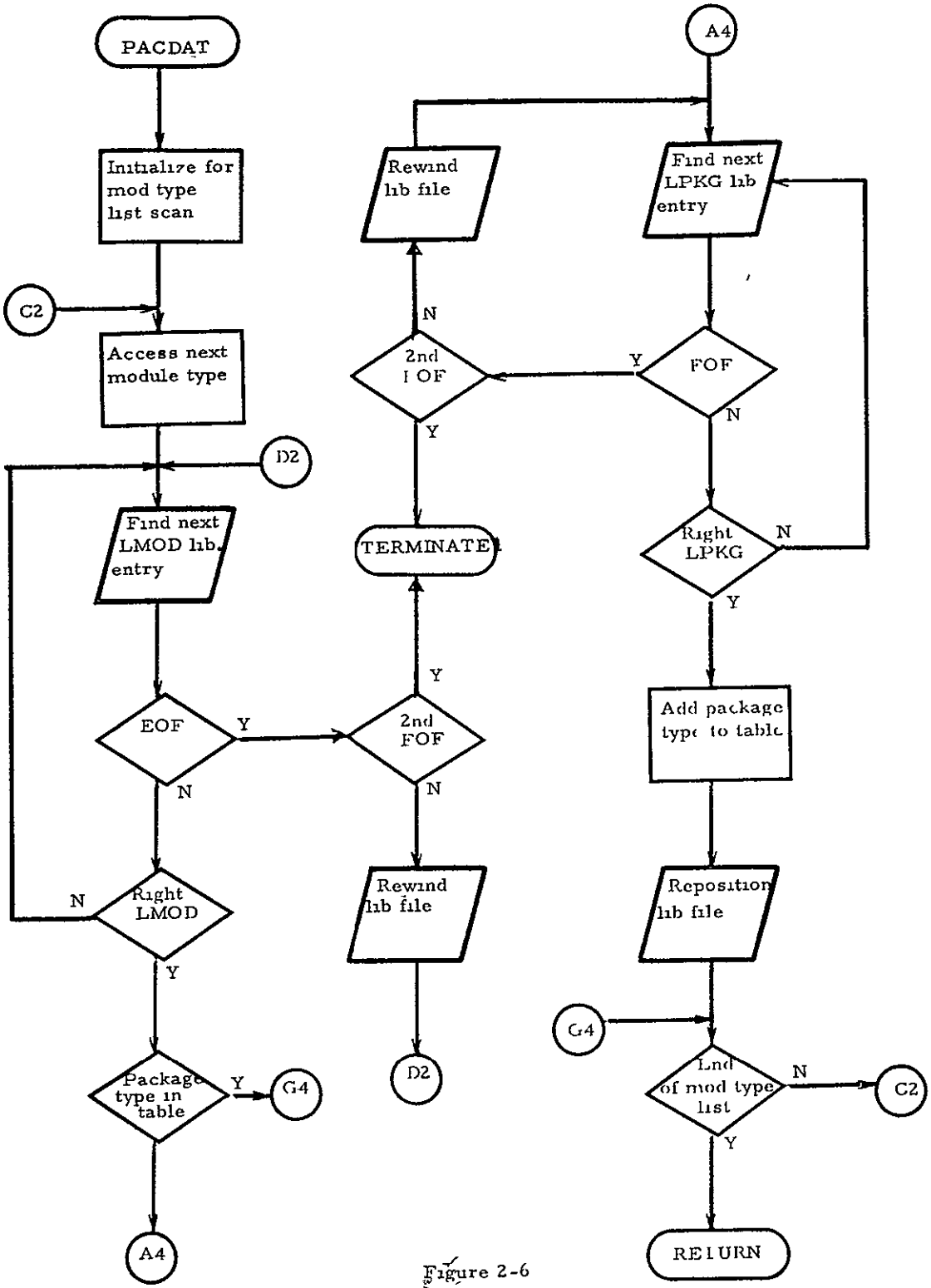


Figure 2-6

ELEMENT PERMUTE CARD GENERATION

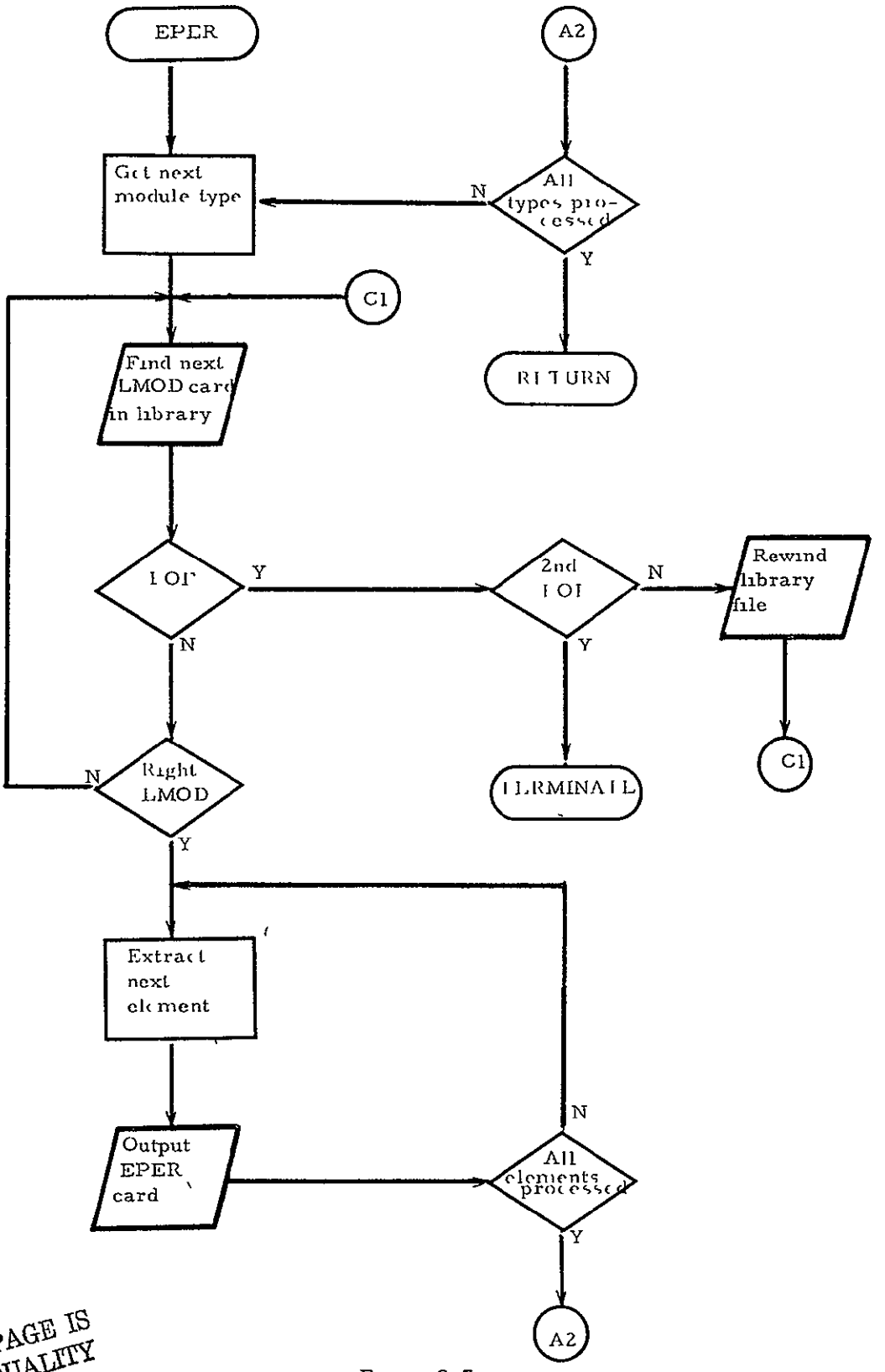


Figure 2-7

PRE-PLACED SOCKET PROCESSING

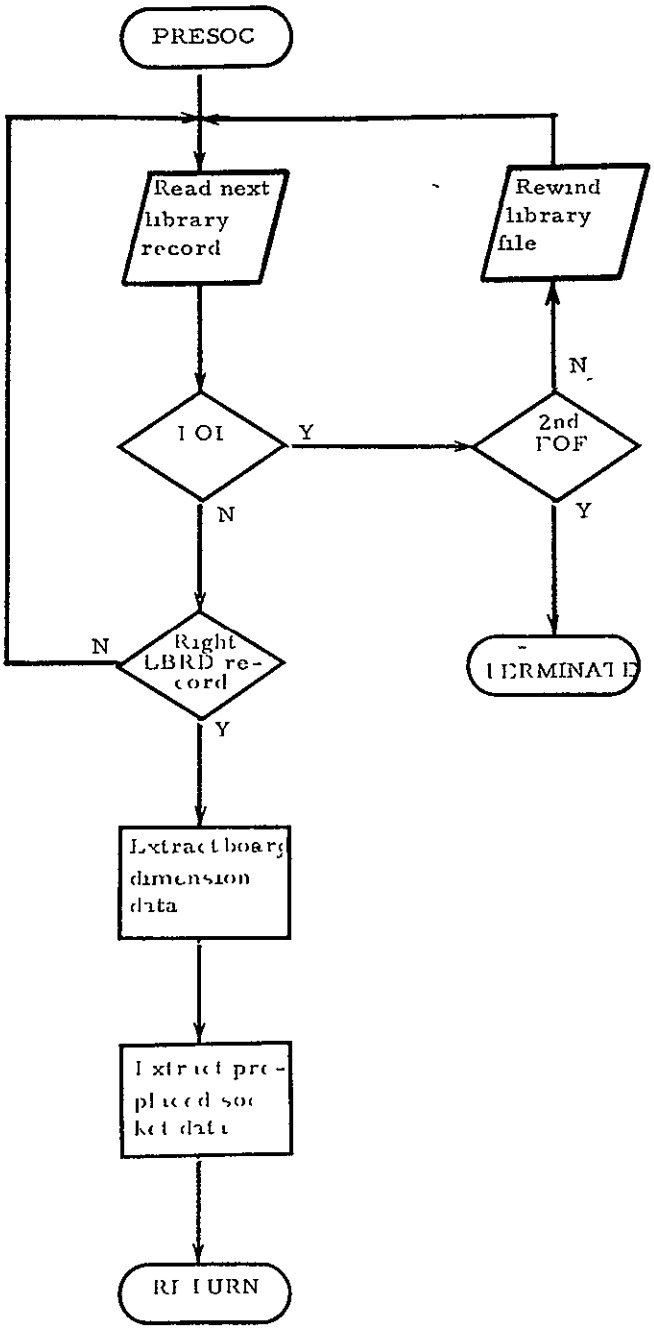


Figure 2-8

SOCKET POSITION CALCULATION

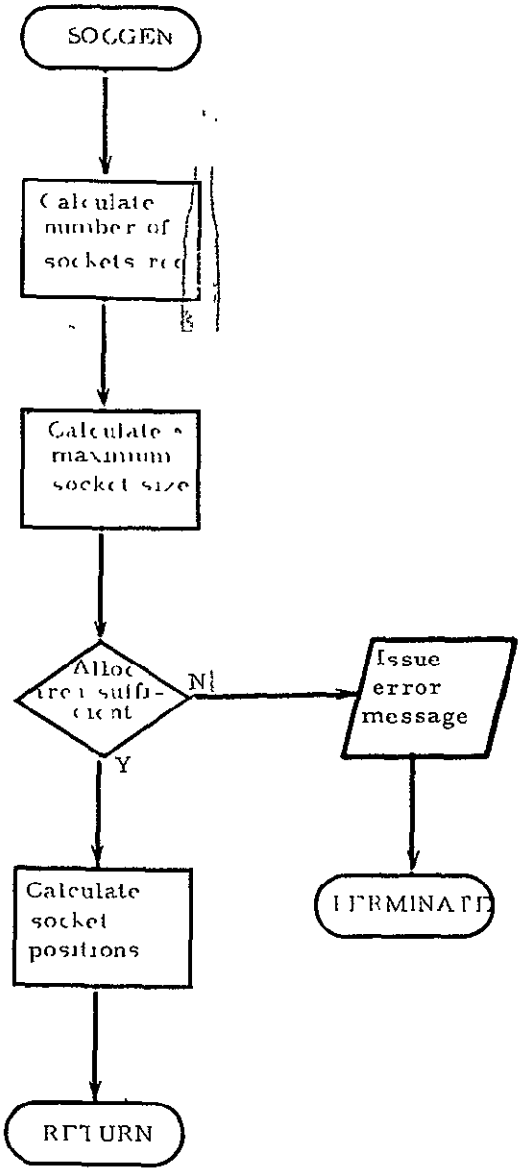


Figure 2-9

SOCKET POSITION CARD GENERATION

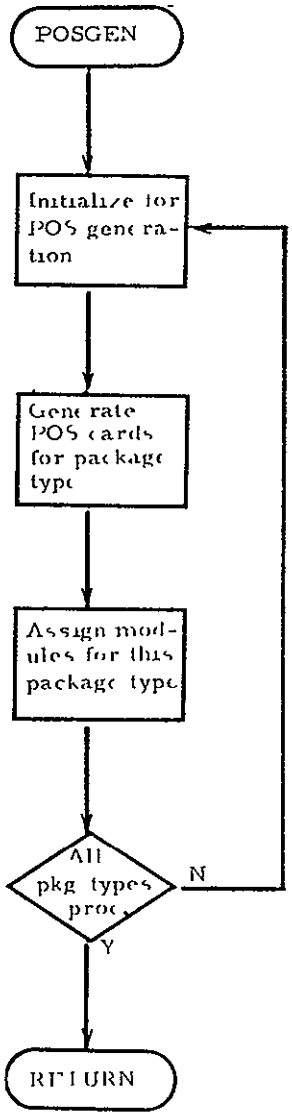


Figure 2-10

BARRIER AND MODULE POSITION
RESTRICTION CARD GENERATION

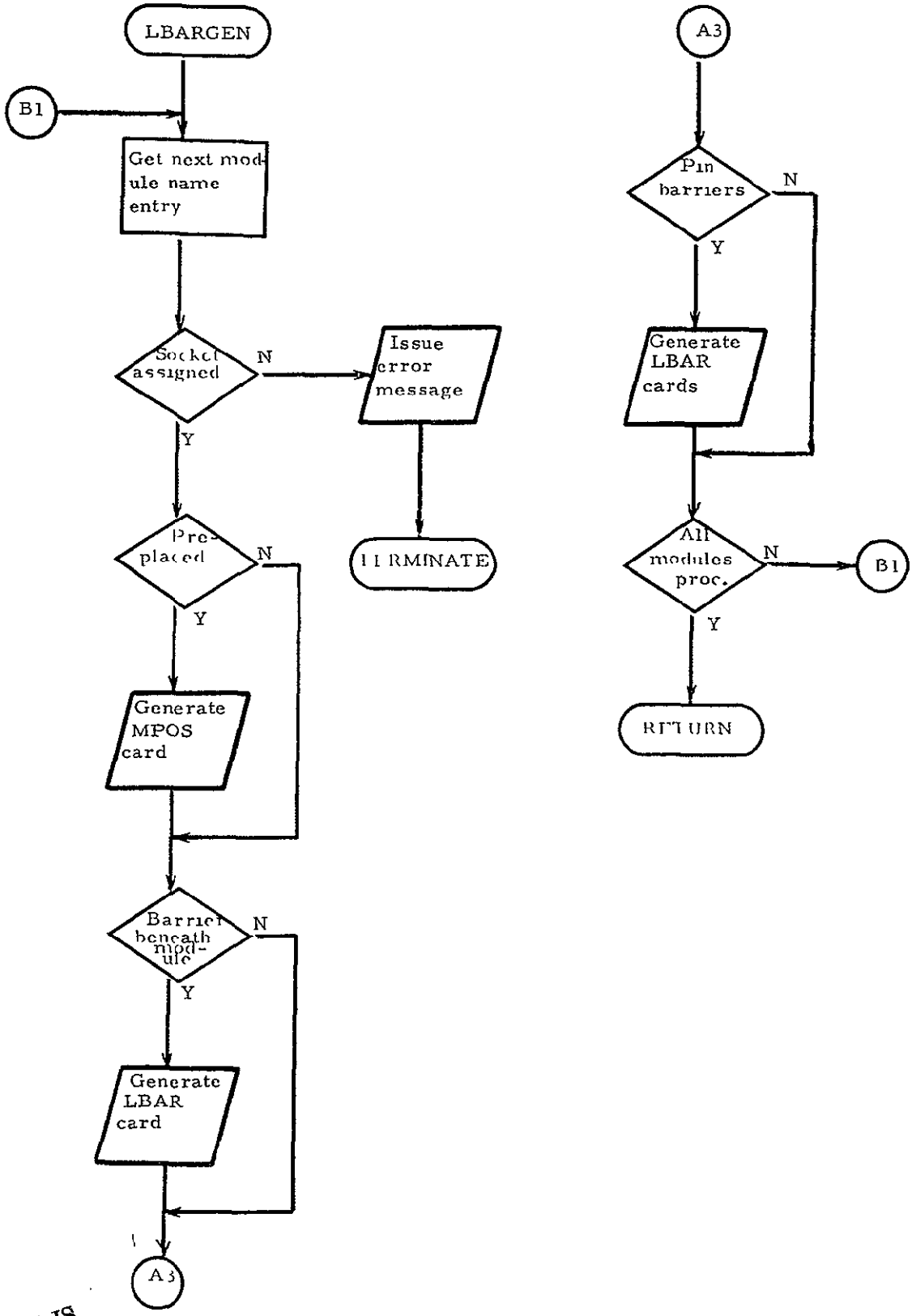


Figure 2-11

MODULE LOCATION FILE GENERATION

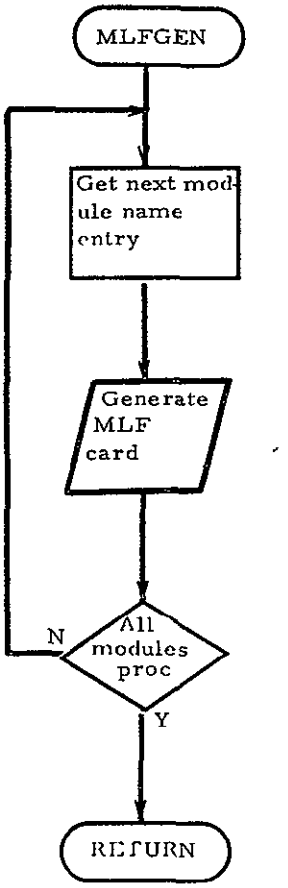


Figure 2-12

PIN CONNECTION FILE GENERATION

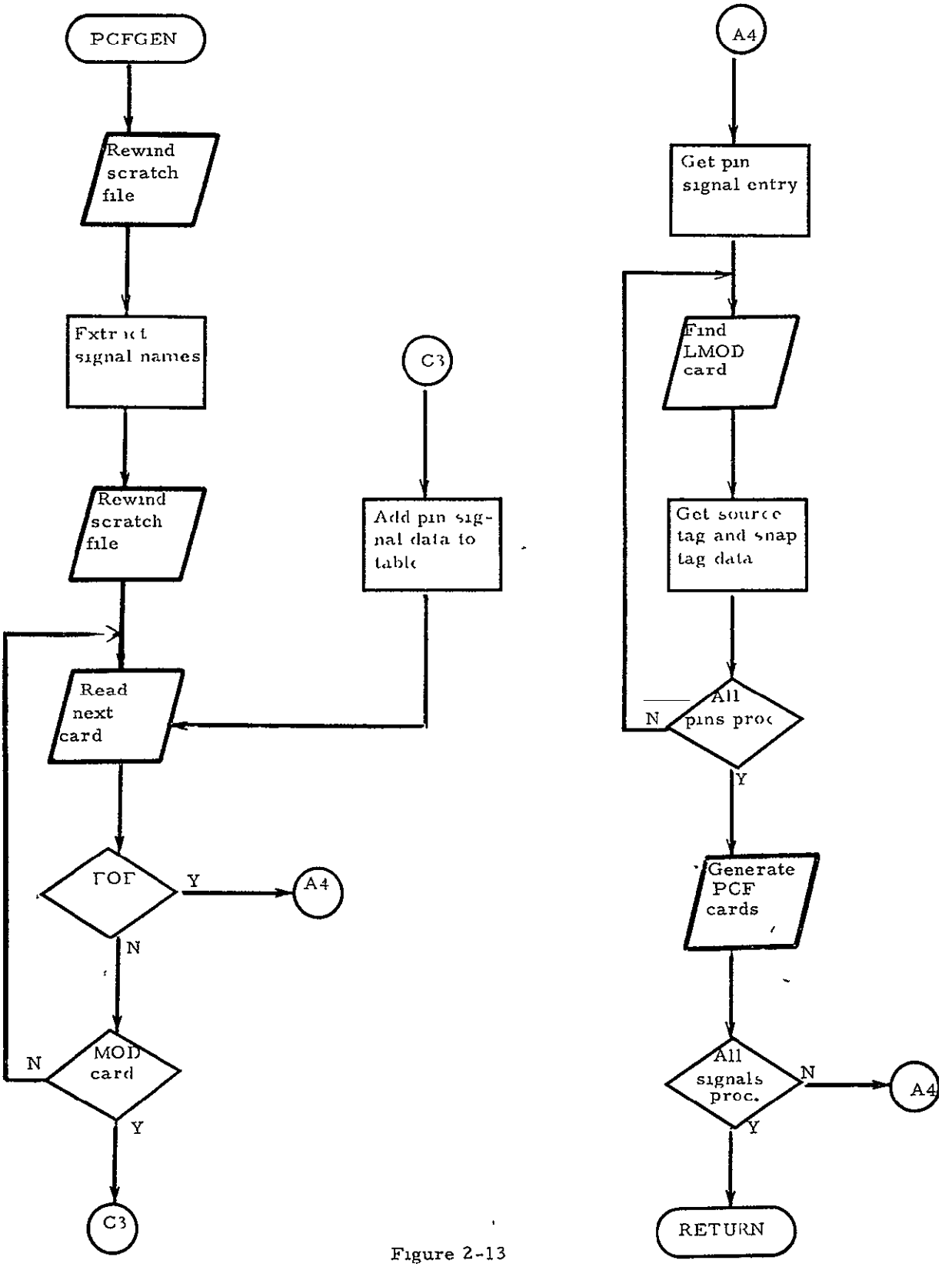


Figure 2-13

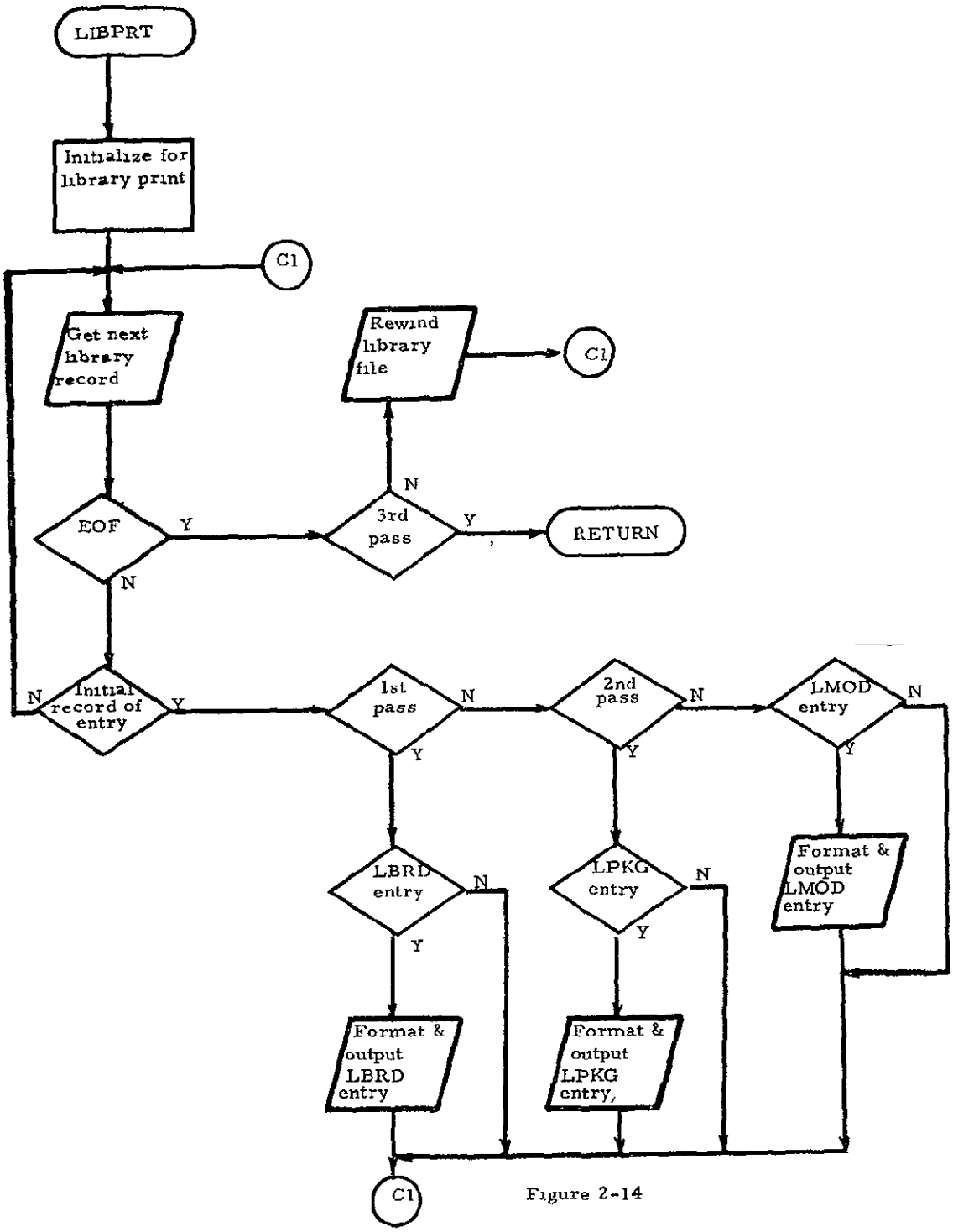

LIBPRT
Initialize for library print
C1
Get next library record
Rewind library file
C1
EOF
Y
3rd pass
Y
RETURN
N
N
N
Initial record of entry
Y
1st pass
N
2nd pass
N
LMOD entry
N
Y
Y
Y
LBRD entry
N
LPKG entry
N
Format & output LMOD entry
Y
Y
Format & output LBRD entry
Format & output LPKG entry
C1

Figure 2-14

2.2 Socket Generation Algorithm

In the Preprocessor socket generation algorithm two assumptions are made

- o Best routing results are obtained when the sockets are evenly distributed over the board surface.
- o All of the modules for which sockets are to be generated are of approximately the same dimensions.

The algorithm consists of 14 steps as follows

1. Calculate the dimensions of the sockets to be generated by determining the maximum X, IMAXX, and maximum Y, IMAXY, of the modules for which sockets are to be allocated.
2. Initialize the row counter, IR_1, to 1.
3. If $IR_1 * IMAXY + (IR_1 + 1) * 3 \geq IALY$, where IALY is the y-dimension of the allocation area, go to step 8.
4. Set the column counter, IC_J, to 1 and increment until $IR_1 : IC_J \geq NUMSOC$, where NUMSOC is the number of modules for which sockets are to be generated.
5. If $IC_J : IMAXX + (IC_J + 1) * 3 > IALX$, where IALX is the x-dimension of the socket allocation area, go to step 1.
6. Calculate $ISEP_{IJ}$ where $ISEP_{IJ} = [IALY - IR_I (IMAXY + 3) - 3] - [IALX - IC_J (IMAXX + 3) - 3]$.
7. Increment IR by 1 and go to step 3.
8. Using the values of IR_1 and IC_J which produce the minimum value of $ISEP_{IJ}$ set $IMX = IR_1$ and $JMX = IC_J$.
9. Set Ix = 1 and Iy = 1.
10. Calculate ISEPX and ISEPY where:

$$ISEPX = IALX - \frac{IMX\ (IMAXX + 3)}{IC + 1}$$

$$ISEPY = IALY - \frac{JMX\ (IMAXY + 3)}{IC + 1}$$

11. Using the socket allocation area origin coordinates IAXO and IAYO calculate X_S and Y_S where

$$X_S = IAXO + I_x (ISEPX + IMAXX) - IMAXX$$

$$Y_S = IAYO + I_y (ISEPY + IMAXY) - IMAXY$$

12. Generate a socket at origin (X_S, Y_S) for each pin array type of the modules for which sockets are being generated.

13. Increment IX by 1 and go to step 11 if IX $\leq$ JMX.

14. Set IX = 1, increment IY by 1 and go to step 11 if IY $\leq$ IMX.

2.3 Preprocessor Program Variables

The Preprocessor capability is limited only by the dimension of the ITAB array and the value of the variable MAXPKG. The ITAB array contains the board, modules and package data and should be dimensioned as large as possible. The variable MAXPKG indicates the maximum number of package types to be processed and must be increased in value if exceeded.

2.4 Preprocessor File Formats

The major output of the Preprocessor Program is the Placement Input File (PIF). This file contains card image records which are used as input to the Placement Program. The data contained in the PIF can be broken into three groups:

- Input control cards describing the circuit board area and board components.
- Module location file cards describing the modules to be placed on the board.
- Pin connection file cards describing the required interconnections between the pins on the board.

The sequence of generated cards on the PIF is:

- BORD card
- ARRY cards for first module

- EPER cards for first module (if any)

- ARRAY cards for last module (if no previous module of the same package type)
- EPER cards for last module (if no previous module of the same module type)
- POS cards for pre-placed and program generated sockets
- MPOS card for first socket (if required)
- LBAR card for first socket (if required)

  .
  .
  .
- MPOS card for first socket (if required)
- LBAR card for last socket (if required)
- ENDC card
- MLF initial card

  .
  .
  .
- MLF last card
- PCF first card

  .
  .
  .
- PCF last card

BLANK

3. PLACEMENT PROGRAM

The Placement Program is designed to produce the optimum placement of a related group of modules in a given set of sockets on a printed wiring board. Using a description of the circuit board and a list of the interconnection requirements between components, the program attempts to place the modules such that the distance in grid units necessary to connect all the specified pins is minimized.

3.1 Placement Algorithm

The placement algorithm is divided into three major functions

- o Module selection - determines the order in which modules are to be placed on the board.
- o Module placement - the placement and evaluation of modules in specific sockets.
- o Element permute - the interchanging of individual elements of a module with identical elements in the same module or in other modules of the same type.

3.1.1 Module Selection

The module selection process uses the pin pair connection data from the PCF to generate a list of the modules in the order in which they are to be placed. Criteria for the ordering of the modules in the list is:

1. Total number of connections to modules already placed.
2. Length constrained connections to modules already placed.
3. The number of sockets which are available to each module.
4. The number of sockets which are available for pre-placed modules.

Each of the above items are weighted by the program in selecting the next module to be entered in the ordered list. The module selection process requires that at least one module be pre-assigned to a specific socket. Normal procedure is to pre-assign the connector module to a specific socket(s).

3.1.2 Module Placement

The module placement process uses the following method to place the modules:

1. Generate an initial partial solution by assigning the pre-assigned modules to specific sockets.

2. Select the first module to be placed from the list of ordered modules.

3. Compute a placement score for the first module in each available socket (available sockets are those of the same type as the module and which are not occupied or overlapped by modules already placed or the module being placed).

4. Select the n best solutions (n is specified in columns 29-32 of the PSOL control card)

5. Generate n additional partial solutions by inserting the sockets from the n best solutions into partial solutions which are equivalent to the initial partial solution.

6. Select the next module from the list of ordered modules.

7. Select the next partial solution (P_i, i = 1, n) from the n partial solutions generated in step 5.

8. Compute a placement score for the new module in each available socket in P_1.

9. Select the m best solutions for P_1 and save (m is specified in columns 33-36 of the PSOL card such that m*n is greater than zero but less than the number of sockets allowed per socket type). Go to step 7.

10. When partial solution P_n has been processed a total of n*m best solutions will have been saved. From these solutions, select the n best partial solutions and generate a new set of n partial solutions by inserting the new sockets into partial solutions equivalent to the originating P_1's. Go to step 6.

11. Module placement is complete when the last module in the ordered list has been placed.

In the above process the placement scores used to determine the best solutions are computed from the following:

1. Total wire length (manhattan distance) for all connections.

2. X, Y orientation of the connecting wires. This factor is computed as the summation of the minimum components of the manhattan distance between components.

3. Number of length constraint violations.

4. Number of connections which must cross barriers, weighted by the number of layers on which the barrier is present.

3.1.3 Element Permute

The element permute function permits the interchanging of individual elements with identical elements in the same or other modules. From the element permute cards, the program generates a list of all elements of the same type. The first element is selected from the list and the following two measures are computed for the first element and the other elements in the list:

1. A placement score for the pair of elements as they are presently located, and

2. a placement score for the pair of elements assuming they had been interchanged.

After all possible pairs have been processed using the first element, the program determines if a better solution has been obtained by the interchanging of the first element with any other element in the list. If not, the same procedure is followed for the second element in the list. If one or more better solutions are found, the first element and the element from the best solution are interchanged and their new locations are used in the computations for the second element. When all elements in the list have been processed in the above manner, the element permute function is complete.

3.2 Placement Program Variables

The amount of data the Placement Program can process is restricted by the size of available memory. The number of modules that may be defined, number of barrier descriptions allowable, etc., are all controlled by program variables. If any modifications to the program limits are required, the value of the controlling variables must be modified and, in addition, the dimensions of the affected arrays must also be modified. Table 3-1 describe the variables which determine the capacity of the program.

PLACEMENT PROGRAM CONTROL VARIABLES

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MAXTYP	Maximum number of pin array types	25	ISPLID, LXREFS, LYREFS, IPAID, ISKCOL, ISKROW, ISKLAY, ISKORN, ISKCNT, IPINUM, LXREFP, LYREFP, IPNCNT, MSKCNT, JSKCNT, JOVER, ITEMP
MAXNUM	Maximum number of sockets per pin array type	100	ISPLID, ISKLAY, ISKORN, LXREFS, LYREFS, JOVER, KSCORE, KVECT, KSOCK, JSCORE, JVECT, JSOCK, LSCORE, MSCORE
MAXPIN	Maximum number of pins per pin array type	50	IPINUM, LXREFP, LYREFP
MAXPER	Maximum number of element permute descriptions	50	LEPERB, LEPERN, LEPERP, LEPERS
MXMPOS	Maximum number of pre-specified placement restrictions	100	LMPOS1, LMPOS2, LMPOS3, LMPOS4

Table 3-1

PLACEMENT PROGRAM CONTROL VARIABLES
(continued)

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MXRPOS	Maximum number of placement restrictions	100	LRPOS1, LRPOS2, LRPOS3, LRPOS4
MXLBAR	Maximum number of barrier descriptions	100	LBARLY, LXBAR1, LYBAR1, LXBAR2, LYBAR2
MAXMOD	Maximum number of modules in the MLF	100	MDISER, MDILOG, LISTV, MDIARY, MDLIST, INITV, LISTT, IMRPOS, MODAVL, MODTOT
MXTLTH	Maximum number of pseudo logic name length constraints	20	LTHRES, LTHMOD
MXPIN2	Maximum number of pins per logic string	100	JMODS, JPID, JSWAP, JTAG, JTEMP, KTEMP
MAXPNP	Maximum number of pin pairs per board	500	ICESR1, ICESR2, ICEPN1, ICEPN2, ICETYP, ICESK1, ICESK2, ICEPT1, ICEPT2, LGSPNP
MAX	Maximum number of permutable elements per module type	100	IELSOK, IELPIN, IELMIN

Table 3-1
(continued)

PLACEMENT PROGRAM CONTROL VARIABLES
(continued)

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MAXSOL	Maximum of solutions	10	LISTV, LISTT
MAXCON	Maximum number of connections between a module and its connected pins or connections to a pair of permutable elements	99	ITAB1, ITAB2, ITAB3, ITAB4, ITAB5, ITAB6, ITAB7, ITAB8, ITAB9, ITAB10, ITAB11, ITAB12

Table 3-1
(continued)

3.3 Placement Files

For each solution requested, the program will produce an updated Pin Connection File and Module Location File that can be used as input to the Organizer Program. A record is also generated of all pins permuted by the program.

3.3.1 Pin Connection File (PCF)

This output file is identical to the Pin Connection Input File except that the 4 character module serial number and the 3 character pin number in the Pin Location ID are replaced by a 4 character socket location and a 3 character pin number respectively for each legal logic string of two or more pins. Single pin and illegal logic strings will have only the module serial number replaced. The data in the PCF will be repeated for each requested placement solution.

The program provides the option of writing the BORD card, the used POS and ARRY cards, and the ENDC card on the output PCF file in BCD card image format immediately before the logic string records are written. This is done for each solution requested to be recorded. If the logical unit address for the output PCF (ICEOUT) on the TAPE card is negative, the cards will be written on the tape. If it is positive, only the logic string records will be written.

3.3.2 Module Location File (MLF)

The output MLF file is identical to the input MLF except that the module location determined by the program is placed in columns 13-16. The data in the MLF file will be repeated for each requested placement solution.

3.3.3 Pin Connection Change File (LOADO)

This file contains pin assignment data for all pins permuted by the program. The records will be recorded in BCD format. The data for each board of each solution for an input PCF file will be delimited with a record containing sentinels (' ' ' ') as the first word. The following record is written for each pin permuted by the program:

Data Characters	Format	Contents
1-4	A4	1) The first record contains (' ' ' ') delimiting data for each board solution.
		2) The last record of the load file will contain (ENDT)

Data Characters	Format	Contents
		indicating the end of tape marker. 3) All other records will contain 4 blank characters.
5-8	A4	Number of pin items: must always contain "0001".
9-16	2A4	Pseudo Logic Name from the input PCF file will be used in this field; left justified.
17-32	4A4	Blank
33-35	A3	Always = "000"
36	A1	A1 = first character of the pin location ID (board identifier).
37-40	A4	Characters 2-5 of pin location ID (module serial number).
41	A1	Always = "0"
42-44	A3	Pin number.
45-48	A4	Always = "0000"
49-84		Blank

4. ORGANIZER PROGRAM

The Organizer Program converts the logic string data generated by the Placement Program into a format from which the required wire paths may be determined by the Router Program. The operations performed by the Organizer Program may be separated into three areas·

Pin Pairing

Layering

Ordering

A detailed description of the algorithm used in each phase is provided in the following paragraphs.

4.1 Pin Pairing Algorithm

Each logic string is read from the PCF and broken into pin pairs which are recorded on the Router Batch Input (RBI) File. If a network consists of a single pin it is either ignored, recorded on the RBI File, or terminated according to the program option selected. After all the logic strings have been processed, any unassigned connector and resistor pins are assigned. If the pin swap option is selected, the Organizer interchanges the pin assignments of equivalent pins. The permutation of pins that creates the fewest crossovers among the wires involved is the one selected. If this differs from the original assignment, the pin pair list is updated.

The pin pairings in a logic string are made by determining the set of pin pair connections which minimizes the total distance required to connect all pins in a logic string. The distance, D, required to connect a pin pair with coordinates (X_1, Y_1) and (X_2, Y_2) is determined as:

$$D = MIN\left(|X_2 - X_1|, |Y_2 - Y_1| \right) + \frac{W}{10} MAX\left(|X_2 - X_1|, |Y_2 - Y_1| \right)$$

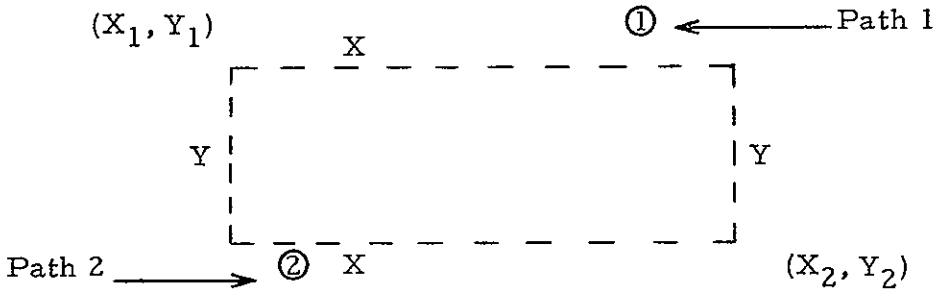
where W is an input weighting parameter specified by the user. When W equals 10, the total X, Y distance will be minimized. When W equals 0, the program will attempt to make all pin pair connections as horizontal or vertical as possible. Values for W between 0 and 10 will result in a trade-off between X, Y orientation and the shortest possible connection distance.

4.2 Layering Algorithm

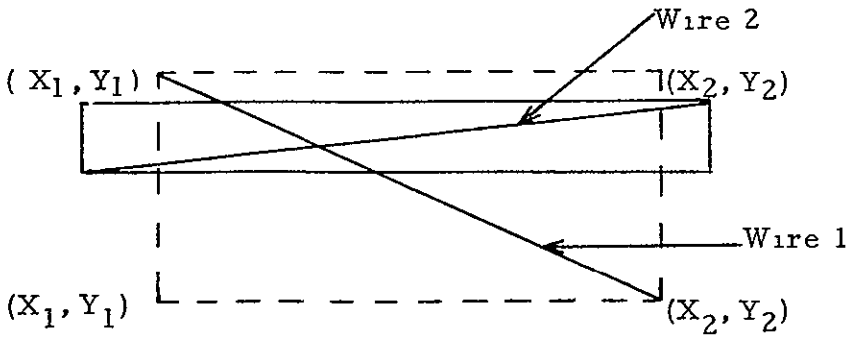
Layering consists of an initial assignment of each pin pair to a specific layer of the board. Single pins are not layered. In the multi-layer version a pin pair is assigned only to the layer where its optimum wire route makes the fewest number of intersections with the other pin pairs assigned to the layer. Pin pairs involving input and output connector pins may be placed on specified layers by exercising a program option. In the two-layer version, feedthrough holes are used to split wires into x-oriented segments and y-oriented segments. One layer contains x-runs and one layer contains y-runs.

For discussion of the algorithm the following definitions are presented:

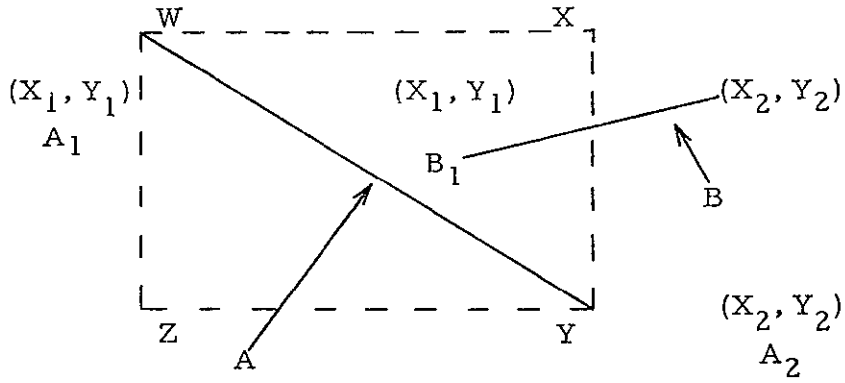
1. Wire path: Given the end points of the wire as (X_1, Y_1) and (X_2, Y_2). The X and Y projections define its path. Thus every wire has 2 possible paths of construction.



2. Intersection Given 2 wires· If the projections of 2 wires cross in each of the 2 possible paths they intersect.



3. Bracketing. Given the end points of wire A the end points of any other wire that fall within the projections of A are now bracketed by A.



B_1 is encompassed by the imaginary rectangle WXYZ, produced from the end points, A_1, A_2 and thus B_1 is bracketed by A.

At this point the following algorithm is used to determine the layer on which each of the remaining wires will be placed:

1. Determine the intersection of all the wires which are to be layered.

2. Select those wires whose total number of intersections is less than the number of available layers. If no wires satisfy this criteria proceed to Step 5.

3. Delete the wires selected in Step 2 from the wire list by assigning removal numbers to each. When wires are deleted using the criteria from Step 2, removal numbers are assigned in ascending order from an initial value of one for the first wire deleted.

4. Reduce from the remaining wires the intersection properties that had been generated by those wires which were most recently deleted. Repeat Step 2.

5. If Step 2 produced no wires that satisfied the criteria, then select a pre-specified number (MAXC) of wires which have maximum intersection properties.

6. Delete those wires selected in Step 5 by assigning removal numbers to each. When wires are deleted using

Step 6, removal numbers are assigned in descending order from an initial value equal to the total number of wires to be layered.

7. Repeat Step 4.

8. When all wires have been assigned removal numbers, the following procedure is used for assignment of wires to specific layers (wires are assigned to layers in order of descending removal number).

   - o Assign a wire to that layer for which the intersection properties are minimal with respect to the wires already residing on that layer.

   - o If 2 or more layers have the same minimal intersection properties, the wire is assigned to the layer which has the most previously assigned wires. If 2 or more layers meet this condition, the wire is assigned the lowest numbered layer.

9. Layering is complete when all wires have been assigned to specific layers.

4.3 Ordering Algorithm

After the pin pairs have been assigned to a specific layer, the organization of each layer is analyzed to determine the order in which the Router Program should attempt to wire the pin pairs. The analysis considers the bracketing effects of the wires as the criterion for determining the routing order.

In the ordering process the total number of points bracketed by each pin pair connection is calculated and saved. When all pin pairs for a layer have been processed, they are arranged in ascending order based on the number of points bracketed. The resultant list is then recorded on the RBI File for use by the Router Program.

4.4 Organizer Program Variables

The amount of data that the Organizer Program can process is dependent upon the value of certain limit variables defined in the program. The dimensions of certain program arrays are in turn dependent upon the variable values. If the program limits are to be

modified, the variable values and array dimensions must be revalued. Table 4-1 lists the limiting variable names and their dependent arrays. The variable values shown are typical but are dependent upon the amount of memory available.

4.5 Organizer Files

The Organizer Program uses the seven files listed below and described in the following paragraphs.

Pin Connection File (PCF)

Router Batch Input File (RBI)

Load List Output File (LOADO)

Module Output File (MODA1)

Temporary Output File

Input Cover File (COVER)

Output Cover File (COVER)

4.5.1 Pin Connection File

This file describes the wiring of each logic string. Logic strings may be broken into separate sections for special constraint purposes, provided that each section is given an individual logic string (signal) name. Any ordering of the PCF is allowable with the exception that continuation cards must follow in sequence for a given logic string. The PCF must follow the ENDC control card unless a PCF was created by the Placement Program. This option is controlled by the TAPE control card.

4.5.2 Router Batch Input File

The Router Batch Input (RBI) File is generated by the Organizer Program and used as input by the Router Program. This file contains the pin pair records grouped by layer and arranged in the order in which the Router Program will attempt to route them. In addition the RBI File contains records which indicate the unused sockets on the board. The RBI will contain the following record groups in the order listed.

ORGANIZER VARIABLES AND ARRAYS

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MAXS	Maximum number of socket locations on the board	145	IUSEDS, IDSPL, IDSPA, ISX, ISY, ISDIMX, ISDIMY, ISOCP, LACODE
MAXP	Maximum number of pins in all the pin types	255	IUSEDP, IPNUM, IDPA, IPX, IPY
MAXR	Maximum number of resistor sockets on the board	40	IUSEDR, IDSL, IDRPA, IRX, IRY, IRDEX, IRNUM, JRSUM, MDIST, LACODR
MAXPLS	Maximum number of pins in one logic string	42	JPID, JSWAP, JTAG, LX, LY, ILIST, IPINA, IPINB, IXIX, IXIY, IDID, ICIC, IX, IY, IB1, IB2, IPIN
MAXTER	Maximum number of logic strings to be terminated	200	LTER, IDTER, ICTER, ISTER, ITGTER, NRF, JPSUM, MXM, MYM, MDIST, ICLOSE, LDIST, LISTR
MAXTYP	Maximum number of pin array types	25	INDEX, IDEX, IDEY

Table 4-1

ORGANIZER VARIABLES AND ARRAYS
(continued)

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MAXWIR	Maximum number of pin pairs on the board	300	(in PERMUT) IX, IY, JX, JY, ISOC, IPIN, IPT, ISWAP, JSOC, JPIN, JPT, JSWAP, ICTAG, ISPN, LOGIC (in LAYORD, ORDERL, etc.) IX, IY, JX, JY, INTX, INTD, NAMEY, IPINA1, IPINA2, IPINB1, IPINB2, LOGPS1, LOGPS2, ICTAG
MAXLYR	Maximum number of board layers	16	LAYIO, LVL, LAYER, LWC, LAYW
MAXSWP	Maximum number of pins with the same swap tag	8	LLIST
MAXSPP	Maximum number of pins paired with pins with the same swap tag	50	JLIST, ID

Table 4-1
(continued)

ORGANIZER VARIABLES AND ARRAYS
(continued)

FORTRAN VARIABLE NAME	FUNCTION	VALUE	DEPENDENT ARRAYS
MAXRPN	Maximum number of pins in a resistor module	6	IRTR, IRES
MAXCON	Maximum number of unassigned connector pins	250	IALOC, IATT, IUA, IULOC, IUTT, ICLOG, ICICS
MAXPGF	Maximum number of potential feed-throughs, used feedthroughs, and ground pins	500	NFARRY, NFCODE, IFX, IFY, NFLYR
MAXBAR	Maximum number of barriers, temp-orary barriers, and feedthrough hole inhibits	500	NBTYPE, IBX, IBY, JBX, JBY, NBLYR
MAXLIN	Maximum number of line positions	50	ILX, ILY, JLX, JLY, NLLYR, LLNAME
MAXCPS	Maximum number of unassigned connector pins in the same socket	250	INUAP, INUUP, ICX, ICY

Table 4-1
(continued)

1. Batch ID record

Data Characters	Contents
1-8	BATCH NNN (NNN Obtained from batch input data card).
9-84	blank

2. Board ID record

Data Characters	Contents
1-44	blank
45-52	Board ID, 8 characters; obtained from Board ID input data cards.
53-57	COVER
58-60	Cover ID - obtained from Board ID input data cards.
61-84	blank

3. Layer ID record

Data Characters	Contents
1-28	blank
29-32	LAYR
33-36	Layer number, decimal right justified.
37-44	Direction compass for routing wires on this layer.
45-48	3 four character, right justified decimal integers to indicate preferred wire routing direction: 0001 = North (- Row direction) 0002 = East (+ column direction)

	0003 = South (+ Row direction)
	0004 = West (- column direction)
49-52	0000 = Any wire may be routed on this layer.
	0001 = Only wires assigned to this layer may be routed on this layer.
53-84	blank

4. Netlist records

Data Characters	Contents
1-8	blank
9-16	Pseudo signal name
17-20	4 character logic string type
21-28	Pin ID
29-31	blank
32	Source tag (from PCF)
33-36	X-coordinate, decimal right justified
37-40	Y-coordinate, decimal right justified
41-44	Layer Code, 0-99, 0=all layer; decimal right justified
45-52	Board ID (8 characters - obtained from board ID input data cards).
53-56	Number of pins per net or number of netlist records to be read into the interconnection matrix of the Router Program.
	= 1 for single pin network records

	= 2 for pin pair network records
57-60	blank
61-84	Real signal name (24 characters) obtained from the SNAME Subroutine

5. Netlist terminal record

Data Characters	Contents
1-4	♭ ♭ ♭ 0
5-44	blank
45-52	Board ID-8 characters
53-84	blank

6. Unused socket record

Data Characters	Contents
1-8	blank
9-16	UNSOCKET
17-24	blank
25-28	POSITION ID (4 characters)
29-32	blank
33-36	X-coordinate of pin or specified position, decimal integer, right justified.
37-40	Y-coordinate of pin for specified position, decimal integer, right justified.
41-44	Layer Code, 0-99, 0=all layer, decimal right justified.
45-52	Board ID - 8 characters
53-84	blank

7. Unused socket terminal record

Data Characters	Contents
1-4	ϕ ϕ ϕ 0
5-44	blanks
45-52	Board ID - 8 characters
53-84	blanks

8. Batch terminal record

Data Characters	Contents
1-44	blanks
45-52	ZERO'S
53-84	blanks

4.5.3 Load List Output File

This file contains 84 character BCD records which specify the connector pin, resistor pin, feedthrough, and permuted pin assignments made by the Organizer Program. The following record is written for each pin assigned by the program.

Data Characters	Format	Contents
1-4	A4	1. The first record contains (' ' ' ') delimiting resistor data. 2. The last record of the load file will contain (ENDT) indicating the end of file marker. No data is recorded in this record. 3. All other records will contain 4 blank characters.
5-8	I4	Number of pin items - must always contain "0001"
9-32	6A4	Real signal name - initially the

		pseudo logic name from the input logic file will be used in this field. Left justified.
33-36	00A2	A2 = board ID followed by a blank character
37-40	A4	Socket location ID
41-44	A3, 0	A3 = pin number ID
45-48	I4	Always = "0000"
49-84		blank

4.5.4 Module Output File

This file will consist of a record in the following format describing the location of each termination resistor module assigned.

Data Characters	Format	Contents
1-4	A4	1. Number of termination modules within the record; always 0001 for a termination resistor location record. 2. (' ' ' ') delimiter record 3. (ENDT) indicates end of tape marker. No data is recorded in this record.
5-12		blank
13-16	00A2	A2 = board ID followed by a blank character.
17-20	A4	Socket location ID
21-84		blank

4.5.5 Temporary Output File

The temporary output file is used as a scratch file by the Organizer Program and contains the following binary records.

1. Netlist pin pair records

Word	Format	Contents
1-6	4 characters per word	24 character real signal name
7	Integer	Pin 1 X(column) grid coordinate
8	Integer	Pin 1 Y(row) grid coordinate
9	4 characters	1 character board ID, followed by 4 character socket location, followed by 3 character pin number
10	4 characters	
11	1 character left justified	Pin 1 type
12	3 characters left justified	Pin 1 swap tag
13-18	Same as words 7-12	Repeat words 7-12 for the second pin in the pin pair.
19-20	4 characters per word	Pseudo signal name
21	1 character left justified	1 character logic string type
22	Integer	= 1 for single pin networks = 2 for all other records

2. Netlist terminal record

Word	Format	Contents
1-18	Integer	Zero
19-20	4 characters per word	Pseudo signal name
21	1 character left justified	blank

22	Integer	Two

3. Unused socket records

Word	Format	Contents
1-2	4 characters per word	UNSOCKET
3	4 characters	Socket position identifier
4	Integer	X(column) socket reference coordinate
5	Integer	Y(row) socket reference coordinate
6	Integer	Zero

4. Unused socket terminal record

Word	Format	Contents
1-6	Integer	Zero

4.5.6 Input Cover File

This file contains a description of the obstacles on the board and is used as an input to the Organizer Program. The Cover File consists of four types of records:

1. ID record

Data Characters	Contents
1-5	COVER
6-8	Cover ID
9-84	blank

2. Obstacle records

This is the only type Cover File record used by the Router Program.

Data Characters	Contents
1-8	OBSTACLE
9-12	Blank
13-14	Blank or "T" or "TT" left justified. Blank = line records T = obstacles to be ignored by the Artwork system, but used as a barrier in the program. TT = temporary obstacles to be ignored by the Artwork system and used as barriers only for single layer routing.
15-19	Blank
20	From-point termination code: 1 = Pin 2 = Barrier or Temporary Barrier and lines 3 = Used feedthrough (equivalent to Code 1) 4 = Ground pin to be used by the Router, same as Code 1. 5 = Hole inhibit - indicating to the Router a hole cannot be drilled, but may be used for a line. 6 = Potential feedthrough to be used for multi-layer routing.
21-24	From-point X-Coordinate, decimal integer
25-28	From-point Y-Coordinate, decimal integer

29-32	To-point termination code; same as above.
33-36	To-point X-Coordinate, decimal integer.
37-40	To-point Y-Coordinate, decimal integer.
43-44	Layer Code 00 through 99, 0 = all layers.
45-52	Blank or first 4 characters of pseudo logic name for line records.
53-56	Pin array name for first pin termination code of 1, 3, 4 or 6 or last 4 characters of pseudo logic name for line records.
57-84	Blank

3. OBSX records

Data Characters	Contents
1-3	OBSX
4	Selected board number (1-9)
5-21	Blank
22-24	Integer format
25	Blank
26-28	Integer format
29-84	Blank

4. Terminal record

Data Characters	Contents
1-8	ENDCOVER
9-84	Blank

4.5.7 Output Cover File

This file is generated by the Organizer Program for use by the Router Program and is identical in format and contents to the Input Cover File described in Section 4.5.6.

5. ROUTER PROGRAM

The Router Program determines the wire paths which will connect the required pin pairs on the board. Input to the program is from the Cover and RBI File generated by the Organizer Program. The Cover File describes the surface area restrictions on each layer of the board. The surface area restrictions include the location of all existing connection pads and feedthrough pads, areas of the board layers which are not to be used for metalization paths, areas of the board where feedthroughs for metalization paths are not permitted, and, if desired, the locations on the board where feedthroughs are permitted, for metalization paths. The RBI File lists, by desired board layer, the point to point connections to be attempted by the Router Program. The number of board layers and unused pin pad locations are also specified on the RBI File.

The input control cards are read and checked for errors. If no errors are detected all manually placed wires are copied from the RBI File onto temporary file 1. The board description, which includes the pin, barrier, feedthrough, and feedthrough inhibit locations for all board layers, is then copied from the Cover File onto temporary file 1.

The following operations are then performed for each board layer encountered on the RBI File:

1. Any manually placed wires for the layer are copied from temporary file 1 to temporary file 2.

2. A layer image in memory is initialized using the board layer description on temporary file 1 and any manually placed wires for the layer which was recorded on temporary file 2.

3. The pin pairs assigned to this layer are read from the RBI File and an attempt is made to connect each pair as it is read. The wire paths for the pin pairs which are successfully connected are saved on temporary file 2 and the layer image is updated to include the wire path. Pin pairs which are not connected are saved on temporary file 1 for later processing by the multi-layer routing routines. When the X, Y routing option is requested single layer routing is not attempted and all pin pairs are recorded on temporary file 1.

After an attempt has been made to route all input pin pairs on their assigned single layer, those pin pairs which could not be connected on single layers are attempted again using the multi-layer routing routines.

The multi-layer board image in memory is initialized using the board description on temporary file 1 and all manually placed and program routed wires for the board on temporary file 2. The wire paths for the pin pairs which are successfully connected are added to the temporary 2 file and the multi-layer board image in memory is updated to include the wire paths. Any pin pairs which cannot be connected are listed in the program printer output and recorded on the RBO output file.

After all requested pin pairs for the board have been processed, a printed image of the artwork for each board layer may be obtained as part of the printer output.

The Router Program will then copy the remaining RBI input records for the board onto the RBO output file. The connected pin pair paths are then read from temporary file 2 and copied onto the RBO output file as network solution header and line segment records. The total path length including feedthrough lengths is accumulated in memory for all logic strings. Length checks for total, main chain, and stub chain lengths are then performed on each logic string and any length violations are recorded on the printer output.

5.1 Single Layer Routing Algorithm

The INITSG, SCGRID, SFAN and SPATH Subroutines are used to route single layer pin pair connections. The single layer image in memory is initialized by the INITSG Subroutine using the data recorded on the temporary 1 and 2 files. Two bits of memory are used to represent each grid unit area of the board layer. The 2 bits for each grid unit will contain one of the following numbers during the execution of the single pin pair routing routines:

0	The grid is unused and available for routing.
1, 2	Numbers which are stored in the unused grid location by the SFAN Subroutine in determining a possible wire path.
3	The grid location is used and is not available for routing a wire path. A grid location is determined to be used if it contains a barrier or pin or is used by a previously determined wire path.

The SCGRID Subroutine clears all the 1 and 2 digits from the layer image in memory prior to an attempt to connect each input pin pair.

The SFAN, Subroutine uses a version of Lee's Algorithm to find any existing paths between a pin pair. The algorithm is implemented

in the program by the following sequence of steps:

1. Start at one pin and determine all grid locations which can be reached in a length of 1 grid unit. A grid location can be reached in 1 grid unit if it is adjacent to the pin in a horizontal or vertical direction and contains the digit zero in its grid location in memory. A fanout bit is stored in these grid locations. The fanout bits are the digits 1 and 2 and are stored in a 2112 sequence. This sequence of fanout digits permits the SPATH Subroutine to retrace a wire path if the SFAN Subroutine determines that a path exists.

2. All grid locations which can be reached in M grid units ($M > 1$) are now those grid units which

   - Have not already been reached in M-1 or fewer grid units,
   - Contain a zero in their grid locations in memory, and
   - Are adjacent in a horizontal or vertical direction to a grid location which can be reached in M-1 grid units.

3. Step 2 is repeated until

   - The second pin is reached,
   - There is no grid location which can be reached in M-1 grid units, or
   - The allowable path length for the pin pair has been exceeded.

If the second pin is reached, one or more paths of length M have been found. The subroutine SPATH is then used to determine which wire path will be used to connect the pin pair. SPATH will also update the layer image in memory by assigning a value of 3 to all grid locations along the wire path. The wire path used is determined by subroutine SPATH by retracing the path starting at the second pin to a grid location M-1 grid units from the first pin then to a grid location M-2 grid units from the first pin etc., until the first pin is reached. Whenever there is a choice between more than one grid location as the next grid location along the retrace path the following rules determine which grid location is to be used·

1. If possible, retrace the path in the preferred layer direction sequence order as input on the RBI File Layer ID Record.

2. If there exists a grid location which is not adjacent to a previously used grid location use this grid location in place of any other useable grid locations which are adjacent to a previously used grid location.

5.2 Multi-Layer Routing Algorithm

The INITMG, CRGRID, MFFAN, TINIT and MPATH Subroutines are used to route multi-layer pin pair connections. The multi-layer board image in memory is initialized by the INITMG Subroutine using the data recorded on the temporary 1 and 2 files. Potential feedthrough locations may be specified in the board description on temporary file 1 or the program can search the entire grid for potential feedthrough locations as specified on the feedthrough inhibit and feedthrough option program control cards. Each grid location for all board layers on an N layer board requires 2N + 3 bits of memory. The 2N + 3 bits of memory for each grid location on the board contain the following information during the execution of the multi-layer routing routines.

- N bits for layer availability (LA) -- 1 bit for each layer

  The bit for each layer is set 1 if the grid on that layer can be used to route a wire path or 0 if the grid on that layer was previously used by a pin, barrier or wire path.

- N bits for layer entry (LE) -- 1 bit for each layer.

  The bit for each layer will contain 1 if one of the wire paths for the pin pair to be processed can enter the grid location on that layer. These bits are set by the MFFAN Subroutine.

- 1 bit for feedthrough locations:

  This bit is set to 1 if the grid location can be used as a potential feedthrough location.

- 2 bits for possible wire paths.

  These bits will contain:

  0 If the grid location has not yet been used by a wire path.

  1,2 To record wire paths in a 2,1,1,2 sequence.

3 - To identify all grid locations which are in a possible wire path M grid units from the starting pin location during the Mth step of the MFFAN Subroutine fanout process.

The CRGRID Subroutine clears the N(LE) bits and the 2 bits for recording the wire paths from the multi-layer board image in memory prior to making an attempt to connect each pin pair. The MFFAN Subroutine uses a modified version of Lee's Algorithm to find existing multi-layer paths between a pin pair. If requested by the T-ing option the TINIT Subroutine is called to search the temporary 2 file for wires which were previously connected to either of the pins. The MFFAN Subroutine is then permitted to find a path between any of the wires which were previously connected to the pins in addition to the possible pin to pin connection.

The modified version of Lee's Algorithm used by the program is described as follows

Let LA for a grid location equal all the layers which can be used to route a wire through the grid location.

Let LE for a grid location equal all layers in LA on which a wire route may enter a grid location as determined by the algorithm.

Let LL for a grid location equal all layers which can be used to leave a grid location. Note that LL will equal LE unless the grid location is a feedthrough location. Then LL is equal to all the layers in the board which are used for routing wires.

The implementation of the algorithm then proceeds through the following steps:

1. Start at one pin and the wires previously connected to the pin and determine all grid locations which can be reached in a length of 1 grid unit. A grid location can be reached in 1 grid unit if:

   - o It is adjacent to one of the starting grid locations in either a vertical or horizontal direction, and
   - o LA for the grid location contains a 1 on at least one of the layers on which the path may connect to the starting grid location.

Set LL=LE=LA for these grid locations.

2. All grid locations which can be reached in M grid units (M > 1) are now those grid locations which

   - o have not already been reached in less than M grid units, and

   - o are adjacent in a vertical or horizontal direction to a grid location which can be reached in M-1 grid units for which LA_M and LL_{M-1} have at least one layer in common.

   LE_M is then set equal to all layers which are common between LA_M of the grid location under consideration and LL_{M-1} from all adjacent grid locations which can be reached in M-1 grid units. LL_M is set to LE_M if the grid location M is not a feedthrough location or to all available layers if the grid location is a feedthrough location.

3. Step 2 is then repeated until

   - o the second pin or a wire previously connected to the pin is reached,

   - o there are no locations which can be reached in M grid units from any of the locations which can be reached in M-1 grid units and a path cannot be found, or

   - o the allowable path length for the pin pair has been exceeded.

When a subroutine MFFAN finds one or more paths of some grid distance M units in length, subroutine MPATH will determine a wire route by retracing a path from the termination point to a grid location M-1 grid units from the starting point, then to a grid location M-2 grid units from the starting point, etc., until the starting point is reached. The following rules determine the path chosen.

1. If possible always proceed in a straight line ignoring the possible feedthrough requirements.

2. Once the multiple layer path is determined by rule 1 pick the layer assignments along this path to minimize feedthroughs.

Subroutine MPATH will also update the multi-layer board image in memory by removing the layer bit of LA in each grid location on the wire path and by removing the feedthrough bits from the grid locations of all used feedthrough locations.

5.3 Router Program Limits

The program assumes a rectangular board layer wiring grid in which wire paths may only be routed in a vertical or horizontal direction. Any or all of the following program restrictions may be changed by changing the limits on the indicated FORTRAN variables which are defined at the beginning of the main program.

1. The maximum board size which can be processed by the program will vary with the number of board layers and the number of wiring grid locations on each layer. The program variable ISIZE determines the board size which can be processed.

   where ISIZE = (NCOL*NROW + NGPW-1)/NGPW

   NCOL = maximum number of grid columns

   NL = number of board layers

   NROW = maximum number of grid rows

   NBW = number of bits per computer word

   NGPW = (NBW + 1) / (2NL + 3)

   Typical maximum board sizes for several different computer memory sizes are presented in Table 5-1.

2. The maximum number of pins in a logic string or network is controlled by the program variable MPINPR.

3. The maximum number of manually placed logic strings for a single board is controlled by the program variable MBLOG.

4. The maximum number of length check parameter cards which can be used is controlled by the program variable ILGHN.

5. The maximum number of logic strings which can be processed by the program if logic string length checks are

MAXIMUM NUMBER OF GRID LOCATION PER LAYER

No. of Board Layers	32K Memory 24 Bits/Word ISIZE = 10,000	32K Memory 36 Bits/Word ISIZE = 10,000	64K Memory 36 Bits/Word ISIZE = 30,000	128K Memory 36 Bits/Word ISIZE = 80,000
2	30,000	50,000	150,000	400,000
3-4	20,000	30,000	90,000	240,000
5-7	10,000	20,000	60,000	160,000
8-10	10,000	10,000	30,000	80,000
11-16	N/A	10,000	30,000	80,000

Table 5-1

to be performed is controlled by the program variable LOGICM.

6. The maximum number of pin pairs to be processed by the program if logic string length checks are to be performed is controlled by the program variable LDM/6.

7. The maximum number of straight line segments for each pin pair connection is controlled by the program variable NSA2. NSA2 also determines the maximum number of minimum wire paths which can be found by the program for a pin pair connection.

8. The length of the board description records written on temporary file 1 is controlled by the variable NSA.

5.4 Router Program Variables

The following table indicates the additional changes which must be made if any of the program limits are to be changed. The values to be used for the computer dependent variables are also shown. If the dimensions of any arrays are changed, the appropriate program EQUIVALENCE statements must also be changed to be consistent with the COMMON storage allocation.

Program Variable	Additional Changes or Description
NBW	Number of bits per memory word
NCW	Number of characters per memory word = 4 for 24 bit memory word = 6 for 36 bit memory word
ISIZE	The array IGRID is dimensioned ISIZE
MPINPR	The arrays MPIN1, MPIN2, MLAY1, MLAY2, MDIS, LNCK1, LNCK2, LNCK3, LNCK4 are dimensioned MPINPR
MBLOG	The array LOGIC is dimensioned MBLOG by 2
ILGHN	The arrays ITLGTH, IMLGTH, ISLGTH, and ILLGTH are dimensioned ILGHN

Program Variable	Additional Changes or Description
LOGICM	The arrays LOGNA, LOGNB, LINKA, LINKB, and LTYPE are dimensioned ILGHN

5.5 Router Files

The Router Program uses the five files listed below and described in the following paragraphs.

Router Batch Input File

Input Cover File

Router Batch Output File

Temporary File 1

Temporary File 2

5.5.1 Router Batch Input File

The Router Batch Input File (RBI) is generated by the Organizer Program and used by the Router Program as a description of the point to point connection to be attempted. A detail description of the format of the RBI File is presented in Section 4.5.2.

5.5.2 Input Cover File

The Cover File generated by the Organizer Program describes the surface area restrictions for each layer of the printed wiring board. The surface area restrictions include the location of all existing pin connection pads and feedthrough pads, areas of the board layers which are not to be used for routing metalization paths, areas of the board where feedthroughs for metalization paths are not permitted, and the locations on the board where feedthroughs are permitted. A detail description of the format of the Cover File is presented in Section 4.5.6.

5.5.3 Router Batch Output File

The RBO File contains records which describe the solutions derived by the Router Program. The data recorded on the file includes netlist records, input option cards, can't connect records, unused socket records, and artwork solution records. The solution records describe the line segments of the connecting wires. All records are recorded in 84 character BCD format. The file will contain the following record groups in the order listed.

1. Batch ID record identical to the batch ID record on the RBI File.

2. Board ID record identical to the netlist records on the RBI File.

3. Netlist records identical to the netlist records on the RBI File.

4. Netlist terminal record identical to the netlist terminal record on the RBI File.

5. Board Description record of the following format:

Data Characters	Contents
1-8	Blank
9-12	BDES
13-16	Number of grid columns in the board wiring grid (Integer format, right justified).
17-20	Number of grid rows in the wiring grid (Integer format, right justified).
21-24	Number of layers on the board.
25-84	Blank

6. Grid distance between layer record in the following format:

Data Characters	Contents
1-8	Blank
9-12	LAYD
13	Number of grid units separating layers 1 and 2.
14	Number of grid units separating layers 2 and 3

.
.
.

27	Number of grid units separating layers 15 and 16
28-84	Blank

7. Feedthrough inhibit option record of the following format:

Data Characters	Contents
1-8	Blank
9-13	FIHBT
14-15	Blank
16	=1, feedthroughs will be inhibited within $\sqrt{2}$ grid locations from any pin or feedthrough.
	=2, feedthroughs will be inhibited within 2 grid locations from any pin or feedthrough.
	=3, feedthroughs will be inhibited within $\sqrt{5}$ grid locations from any pin or feedthrough.
	=4, feedthrough will be inhibited within $\sqrt{8}$ grid locations from any pin or feedthrough.
17-84	Blank

8. Feedthrough option record of the following format·

Data Characters	Contents
1-8	Blank
9-12	FOPT
13-15	Blank

16	=1, only potential feedthrough as indicated by unused sockets on the RBI File and feedthrough locations on the COVER File will be used.
	=2, the program will search the grid for all potential feedthroughs which satisy the feedthrough inhibit option specified.

9. Unused socket option record of the following format:

Data Characters	Contents
1-8	Blank
9-12	OPTO
13-15	Blank
16	=1, use unused socket locations as potential feedthroughs.
	=2, remove unused socket pins from the board description.
	=3, ignore unused socket records.

10. Can't connect wire records. Any number of these records may be present. They describe the end-points for connections that could not be made by the Router Program.

Data Characters	Contents
1-4	Blank
5-8	4 Character logic string type
9-16	Pseudo logic name
7-24	From pin ID
25-28	Column grid coordinate; (integer, right justified).

29-32	Row grid coordinate, (integer, right justified).
33-36	From pin layer code. Zero indicates all layers, (integer, right justified).
37-44	To pin ID
45-48	Column grid coordinate; (integer, right justified).
49-52	Row grid coordinate, (integer, right justified).
53-56	To pin layer code. Zero indicates all layers.
57-60	Layer for this wire, =0 if any layer can be used.
61-84	Real logic name

11. Unused socket records identical to the unused socket records on the RBI File.

12. Unused socket terminal record identical to the unused socket terminal record on the RBI File.

13. Solution signal header record of the following format:

Data Characters	Contents
1-8	Blank
9-16	Pseudo logic name (8 characters)
17-20	4 character logic string type. Characters 21-24 of PCF File logic string data.
21-28	"From" symbolic pin ID (8 characters)
29-32	Blank

33-40	"To" symbolic pin ID (8 characters)
	NOTE - Data Characters 21-28, 33-40 can be X-Y coordinates if T-ing has occurred.
41-44	Blank
45-52	Board ID (8 characters)
53-56	Blank
57-60	Connection cell count, decimal integer. This is a tally of grid units of specified line segments following this record.
61-84	Real logic name (24 characters)

14. Solution line segment records of the following format

Data Characters	Contents
1-8	Blank
9-16	Pseudo logic name (8 characters)
17-20	First end point termination code, (1 = pin, 2 = line, 3 = feedthrough); decimal integer.
21-24	First end point X-coordinate; decimal integer.
25-28	First end point Y-coordinate; decimal integer.
29-32	Second end point termination code, (same as above).
33-36	Second end point X-coordinate; decimal integer.
37-40	Second end point Y-coordinate; decimal integer.

41-44	Layer Code (01 through 99), decimal integer.
45-52	Board ID (8 characters)
53-60	Blank
61-84	Real logic name (24 characters)

15. Solution terminal record of the following format.

Data Characters	Contents
1-4	♭ ♭ ♭ 0
5-43	Blank
44-52	Board ID (8 characters)
53-84	Blank

16. Batch terminal record identical to the batch terminal record on the RBI File.

5.5.4 Temporary File 1

Temporary File 1 is used to save the manually placed wire segment input control cards for all boards to be processed, a compact board description for the board being processed by the program, and a list for all pin pairs which cannot be connected during the single layer routing pass for the board being processed.

This file contains the following record groups in the order listed. All records are recorded in binary word format.

1. Manually placed wire segment records

Data Words	Contents
1-2	8 character pseudo logic name
3	First end point X grid coordinate
4	First end point Y grid coordinate
5	Second end point X grid coordinate

6	Second end point Y grid coordinate
7	First end point termination code: 1 = pin 2 = line 3 = feedthrough
8	Second end point termination code: 1 = pin 2 = line 3 = feedthrough
9	Board layer
10-11	8 character FROM symbolic pin ID
12-13	8 character TO symbolic pin ID
14-15	8 character Board ID
16-21	24 character real logic name

2. End of manually placed wire segments record

Data Word	Contents
1	ENDT
2-21	blank

3. Board description records

Data Word	Contents
1	Record type, 1 character left justified. P = Pin location record. Generated from Cover File records with

	end point termination codes of 1, 3 and 4.
	B = Barrier location records. Generated from Cover File records containing T, blank in characters 13 and 14 and an end point termination code of 2.
	T = Temporary barrier location records. Generated from Cover File records with TT in characters 13 and 14 and an end point teimination code of 2.
	I = Feedthrough inhibit location record. Generated from Cover File records with an end point termination code of 5.
	F = Potential Feedthrough location record. Generated from Cover File records with an end point termination code of 6.
	FF = RBI File unused socket records when the unused socket option is 1.
	U = Unused pin location records. Generated from unused socket records when the unused socket option is 2.
2	Layer number for this board description record. 0 = all layers
3	Number of grid positions which follow for the board description record.
4 to 3 + contents of word 3	Position in the wiring grid which contains the parameter described in word 1. grid position = (grid row-1) x (number grid columns) + grid column

4. End of board description record

Data Word	Contents
1	ENDT
2-4	Zero

5. Can't connect wire records

Data Word	Contents
1-2	8 character pseudo logic name.
3	4 character logic string type
4-5	8 character first pin symbolic pin ID
6	First pin source tag preceded by 3 blank characters.
7	First pin X grid coordinate
8	First pin Y grid coordinate
9	Layer on which first pin is located, zero indicates all layers.
10-11	8 character Board ID
12	Binary integer 2
13	Layer on which second pin is located, zero indicates all layers.
14-19	24 character real logic name
20-21	8 character second pin symbolic pin ID.
22	Second pin X grid position
23	Second pin Y grid position

24	Layer number to which pin pair was assigned by Organizer Program.
25	Allowable layers for this pin pair. 0 = all layers, if non-zero the nth bit from the right equal to 1 indicates layer n can be used to route this pin pair.
26	Optimum distance between the pin pair in grid units.
27	Second pin source tag preceded by three blank characters.

6. End of can't connect wire records

Data Word	Contents
1	ENDT
2-27	blanks

5.5.5 Temporary File 2

Temporary file 2 is used to save a compact form of the wire routes for all connected pin pairs for the board being processed. This file contains the following records in the order listed. All records are recorded in binary word format.

1. Board ID record

Data Word	Contents
1-2	8 character Board ID
3-4	8 character Cover ID
5-10	Blank

2. Artwork solution records

Data Word	Contents
1	M = Number of straight line segments used to route the pin pair connection.

2-7	24 character real logic name
8-9	8 character first pin symbolic pin ID
10-11	8 character record pin symbolic pin ID. This field may contain the X, Y grid coordinates if T-ing has occurred. Y(row) position of the next route segment end point coordinate.
13	X(column) position of the next route segment end point coordinate. Layer for the next route segment. This word will contain the layer number for wires routed on a single layer and will contain a zero for wires routed on multi-layers. Words 12, 13, and 14 are repeated for each of the M-1 end points of the M straight line segments.
3M + 14, 3M + 15	8 character pseudo logic name,
3M + 16	4 character logic string type,
3M + 17	First pin source code preceded by 3 blank characters.
3M + 18	Second pin source code preceded by 3 blank characters.
3M + 19	First end point termination code.
3M + 20	Second end point termination code.

REFERENCES

1. "Banning Artwork Program User's Manual", M&S Computing, Inc., Report No. 70-0012, April 24, 1970.

2. "Programmer's Manual for the Banning Artwork Program", M&S Computing, Inc., Report No. 70-0013, June 5, 1970.

3. "Computer Graphics System for Aiding the Design of Printed Circuit Boards: User's Manual", General Electric Company, Contract No. 8-21413; January 31, 1970.

4. "Computer Graphics System for Aiding the Design of Printed Circuit Boards: Placement Program Description"; General Electric Company, Contract No. 8-21413; January 31, 1970.

5. "Computer Graphics System for Aiding the Design of Printed Circuit Boards: Organizer Program Description", General Electric Company, Contract No. 8-21413; January 31, 1970.

6. "Computer Graphics System for Aiding the Design of Printed Circuit Boards: Router Program Description"; General Electric Company, Contract No. 8-21413; January 31, 1970.

7. "Printed Wiring Board System: User's Manual", M&S Computing, Inc., Report No. 73-0014, 1973.

☆U S GOVERNMENT PRINTING OFFICE 1976-641-255/552 REGION NO 4