

NASA CONTRACTOR  
REPORT

NASA CR-144174

LOGSIM PROGRAMMER'S MANUAL

By C. L. Mitchell and J. F. Taylor  
M & S Computing, Inc.  
Huntsville, Alabama

February 1976

(NASA-CR-144174) LOGSIM PROGRAMMER'S MANUAL  
(M&S Computing, Inc.) 103 p HC \$5.50  
CSCL 09B

N76-18821

Unclas  
G3/61 14308

REPRODUCED BY  
NATIONAL TECHNICAL  
INFORMATION SERVICE  
U. S. DEPARTMENT OF COMMERCE  
SPRINGFIELD, VA 22161

**PRICES SUBJECT TO CHANGE**

Prepared for

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

N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM THE  
BEST COPY FURNISHED US BY THE SPONSORING  
AGENCY. ALTHOUGH IT IS RECOGNIZED THAT CER-  
TAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RE-  
LEASED IN THE INTEREST OF MAKING AVAILABLE  
AS MUCH INFORMATION AS POSSIBLE.

1. REPORT NO. NASA CR-144174		2. GOVERNMENT ACCESSION NO.		3. RECIPIENT'S CATALOG NO.	
4. TITLE AND SUBTITLE LOGSIM PROGRAMMER'S MANUAL				5. REPORT DATE February 10, 1976	
				6. PERFORMING ORGANIZATION CODE	
7. AUTHOR(S) C. L. Mitchell and J. F. Taylor				8. PERFORMING ORGANIZATION REPORT # 72-0002	
9. PERFORMING ORGANIZATION NAME AND ADDRESS M & S Computing, Inc. P. O. Box 5183 Huntsville, Alabama 35805				10. WORK UNIT NO.	
				11. CONTRACT OR GRANT NO. NAS8-25621	
12. SPONSORING AGENCY NAME AND ADDRESS National Aeronautics and Space Administration George C. Marshall Space Flight Center Marshall Space Flight Center, Alabama 35812				13. TYPE OF REPORT & PERIOD COVERED Contractor Report January 18, 1972	
				14. SPONSORING AGENCY CODE	
15. SUPPLEMENTARY NOTES Electronics Development Division, Electronics and Control Laboratory Design Techniques Branch					
16. ABSTRACT This document is a programmer's manual for a Logic Simulator (LOGSIM) computer program that is a large capacity event simulator with the capability to accurately simulate the effects of certain unknown states, rise and fall times, and floating nodes in large scale Metal Oxide Semiconductor logic circuits. A detailed description of the software with flow charts is included within the report.					
17. KEY WORDS			18. DISTRIBUTION STATEMENT Unclassified-Unlimited COR: <i>John W. Gould</i>  EC01 <i>for F. B. Moore</i> <i>Director, E&amp;C Lab</i>		
19. SECURITY CLASSIF. (of this report) Unclassified		20. SECURITY CLASSIF. (of this page) Unclassified		21. NO. OF PAGES 108	22. PRICE

*NU*

## TABLE OF CONTENTS

<u>Section</u>	<u>Page No.</u>
1. INTRODUCTION	1
2. PREPROCESSOR	4
2.1 Program Structure	4
2.2 Subroutine Descriptions	4
2.2.1 CARDPRO: Card Processing Subroutine - Level 1	4
2.2.2 CHEKLIST: Error Check and Data List Subroutine - Level 1	6
2.2.3 PACSTORE: Array Packing and Storing Subroutine - Level 1	6
2.2.4 FROMTO: Connectivity List Generation Subroutine - Level 1	6
2.2.5 READ: Data Record Reading and Identi- fication Subroutine - Level 2	6
2.2.6 NAMCON: Name and Control Option Processing Subroutine - Level 2	7
2.2.7 SPEC: Circuit Specification Subroutine - Level 2	7
2.2.8 TIME: Time Record Processing Sub- routine - Level 2	7
2.2.9 NEWGATE: NEWGATE Specification Processing Subroutine - Level 2	8
2.2.10 ROM: Read-Only-Memory Specifica- tion Processing Subroutine - Level 2	8
2.2.11 CREST: Element Description Records Processing Subroutine - Level 2	8
2.2.12 SAVETAPE: Data Tape Generation Subroutine - Level 2	9
2.2.13 SUBERR: Data Error and Substitution Subroutine - Level 3	9
2.2.14 ERROR: Indeterminate and Misplaced Data Skip Subroutine - Level 3	9
2.2.15 NETPRO-NET: Data Processing Sub- routine - Level 3	9
2.2.16 GENER: Generator Data Processing Subroutine - Level 3	10
2.2.17 GENFUN: Generator Function Data Processing Subroutine - Level 3	10

TABLE OF CONTENTS  
(continued)

<u>Section</u>		<u>Page No.</u>
2.3	Arrays and Variables	10
2.4	Secondary Data Storage	22
	2.4.1 Temporary Data Files	22
	2.4.2 Simulator Input Data File	22
	2.4.3 Postprocessor Input Data File	26
3.	SIMULATOR	30
3.1	Program Structure	30
3.2	Subroutine Description	30
	3.2.1 INITIAL: Gate Initialization Sub- routine - Level 1	30
	3.2.2 SIMRUN: Event Simulation Sub- routine - Level 1	32
	3.2.3 TERM: Termination Subroutine - Level 1	32
	3.2.4 SUB103: Logic State Determination Subroutine - Level 2	32
	3.2.5 REGEN: Generator Events Restoration Subroutine - Level 2	32
	3.2.6 BUFOUT: Event Output Subroutine - Level 2	32
	3.2.7 ROM: Read-Only-Memory State Determination Subroutine - Level 3	33
	3.2.8 INDEL: Delay Gate and Inverter State Determination Subroutine - Level 3	33
	3.2.9 TYPVALS: Logic Gate State Determin- ation Subroutine - Level 3	33
	3.2.10 ENTRE: Event Storage Subroutine Level 3	33
	3.2.11 INENTRE: Initial Conditions Storage Subroutine - Level 4	33
	3.2.12 HOLD: Holding Mode Event Processing Subroutine - Level 4	34
	3.2.13 MCLEAN: FEC Clearing Subroutine - Level 4	34

TABLE OF CONTENTS  
(continued)

<u>Section</u>	<u>Page No.</u>
3.3 Arrays and Variables	34
3.4 Debug Output	38
4. POSTPROCESSOR	40
4.1 Program Structure	40
4.2 Subroutine Description	40
4.2.1 POSTCARD: Card Processing Subroutine - Level 1	40
4.2.2 PREPRINT: Timing Diagram Setup Subroutine - Level 1	40
4.2.3 INITIAL: Element Initial Conditions Processing Subroutine - Level 1	42
4.2.4 PREVENT: Event Processing Subroutine - Level 1	42
4.2.5 CONTROL: Control and Output Character and Shift Card Processing Subroutine - Level 2	42
4.2.6 COMCARD: Compare and Compare Function Card Processing Subroutine - Level 2	43
4.2.7 PNTSLT: PNT/SLOT Card Set Processing Subroutine - Level 2	43
4.2.8 COMPARE: Level Comparison Processing Subroutine - Level 2	43
4.2.9 SPIKE: Spike Notice Processing Subroutine - Level 2	43
4.2.10 ERROR: Error Message Processing Subroutine - Level 2	44
4.3 Arrays and Variables	44
APPENDIX A LOGSIM PREPROCESSOR PROGRAM FLOW-CHARTS	A-1
APPENDIX B LOGSIM SIMULATOR PROGRAM FLOWCHARTS	B-1
APPENDIX C LOGSIM POSTPROCESSOR PROGRAM FLOW-CHARTS	C-1

## 1. INTRODUCTION

The Logic Simulation Program (LOGSIM) is a system of computer programs written in FORTRAN IV that check the functional correctness of a logic design by simulating the logic at the logic gate level. In addition it checks the propagation delay through the various logic nets and will generate printouts indicating that timing or "race" conditions exist which should be examined more carefully by the system or logic designer. The program can be used to check logic for virtually any technology in which the logic is expressed in Boolean logic or an effective equivalent.

The LOGSIM Program has an internal logic library of 15 logic element types and provides the user with the capability of defining any number of new logic functions as new gate types or read-only-memories (ROM). The LOGSIM User's Manual provides the information required to format a circuit description in terms of LOGSIM inputs.

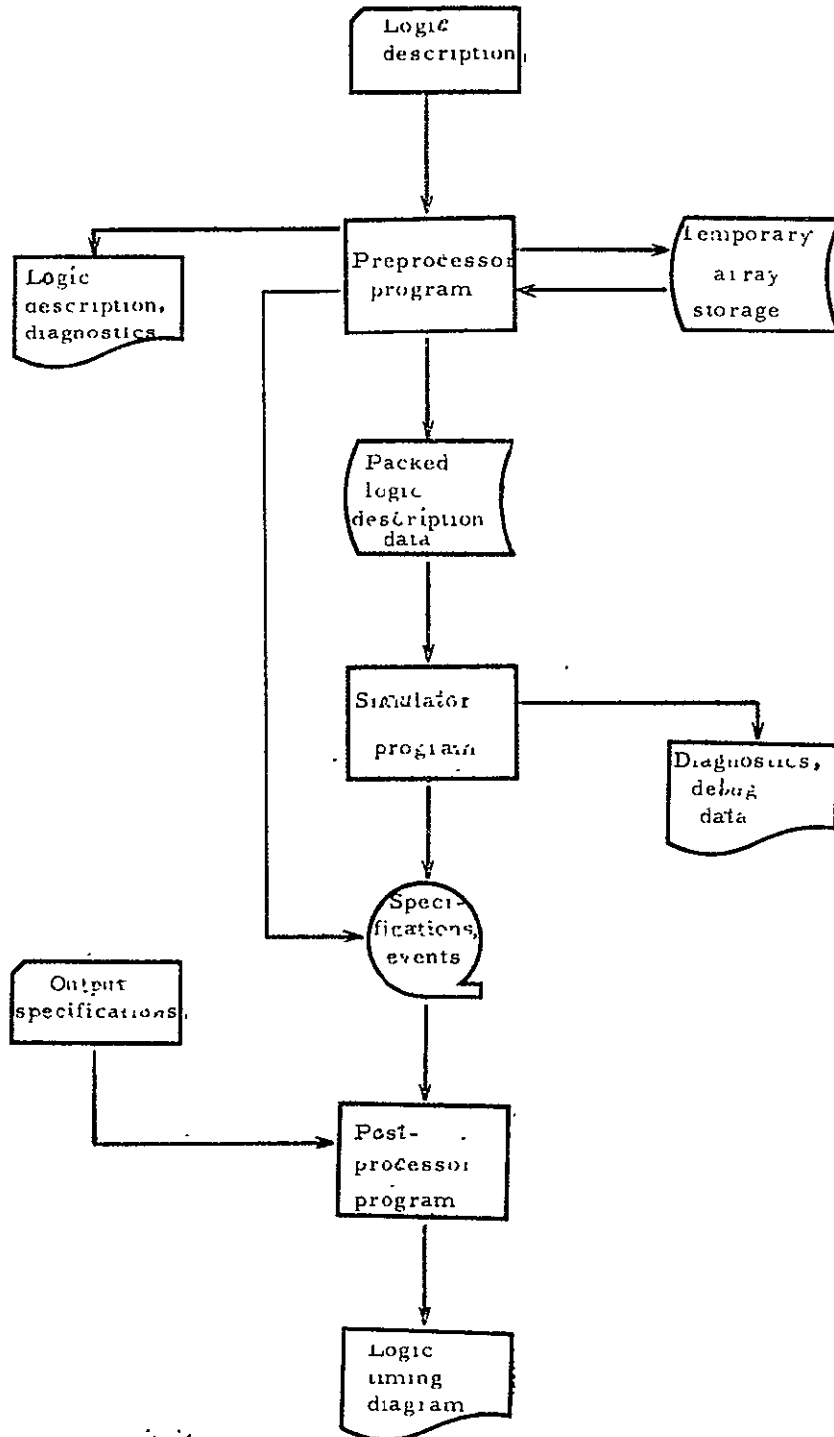
LOGSIM is capable of simulating up to 5000 logic gates within approximately 50,000 words of 32-bit computer memory. To achieve the capability of simulating large networks, LOGSIM was structured into three separate stand-alone programs: Preprocessor, Simulator, and Postprocessor. Figure 1-1 illustrates the general flow of execution of the LOGSIM system of programs.

This document is intended to serve as the Programmer's Manual for LOGSIM and to describe the program structure and internal logic flow in detail. It is recommended that every user become familiar with this document although it is not required to effectively use the program. The LOGSIM User's Manual is intended to serve as the basic user guide to the program and contains the information required to successfully use the program. This document assumes the reader has read the User's Manual and has some prior experience in the use of the LOGSIM Program.

This Programmer's Manual follows the structure of the LOGSIM program as the 3 major sections correspond to the 3 major LOGSIM Programs:

- o Preprocessor
- o Simulator
- o Postprocessor

LOGSIM FLOW OF EXECUTION



ORIGINAL PAGE IS  
OF POOR QUALITY

Figure 1 - 1



Each section describes the program structure, subroutines, arrays and variables, and size. The subroutine descriptions refer to the program flowcharts contained in Appendices A, B, and C. The subroutines described may refer to blocks of coding in the main program as well as actual program subroutines.

## 2. PREPROCESSOR

The Preprocessor Program reads the input data describing the logic network and packs the information into a shorthand format compatible with the Simulator Program's input requirements.

### 2.1 Program Structure

Figure 2-1 illustrates the functional structure of the Preprocessor (PREPRO) Program. The program is structured into four levels of functional subroutines with the first functional level corresponding to the four main program functions:

- (1) Processing input data cards
- (2) Verifying input data cards
- (3) Packing and storing Simulator input data
- (4) Constructing the connectivity list.

Appendix A contains the detail flowchart of the Preprocessor Program.

### 2.2 Subroutine Descriptions

Each subroutine illustrated in Figure 2-1 represents a particular feature of the program and may include a block of the main program coding and/or one or more actual program subroutines. The functional subroutines are discussed in order from Level 1 through Level 4 and their respective flowcharts are illustrated in Appendix A in that order.

#### 2.2.1 CARDPRO: Card Processing Subroutine - Level 1

The functional subroutine CARDPRO reads the input data and sorts it according to data types into the proper data arrays. In the actual program CARDPRO includes:

- o MAIN - first major coding block
- o SUBROUTINES - HEAD1, HEAD2 (output page heading subroutines)

PREPROCESSOR (PREPRO) PROGRAM STRUCTURE

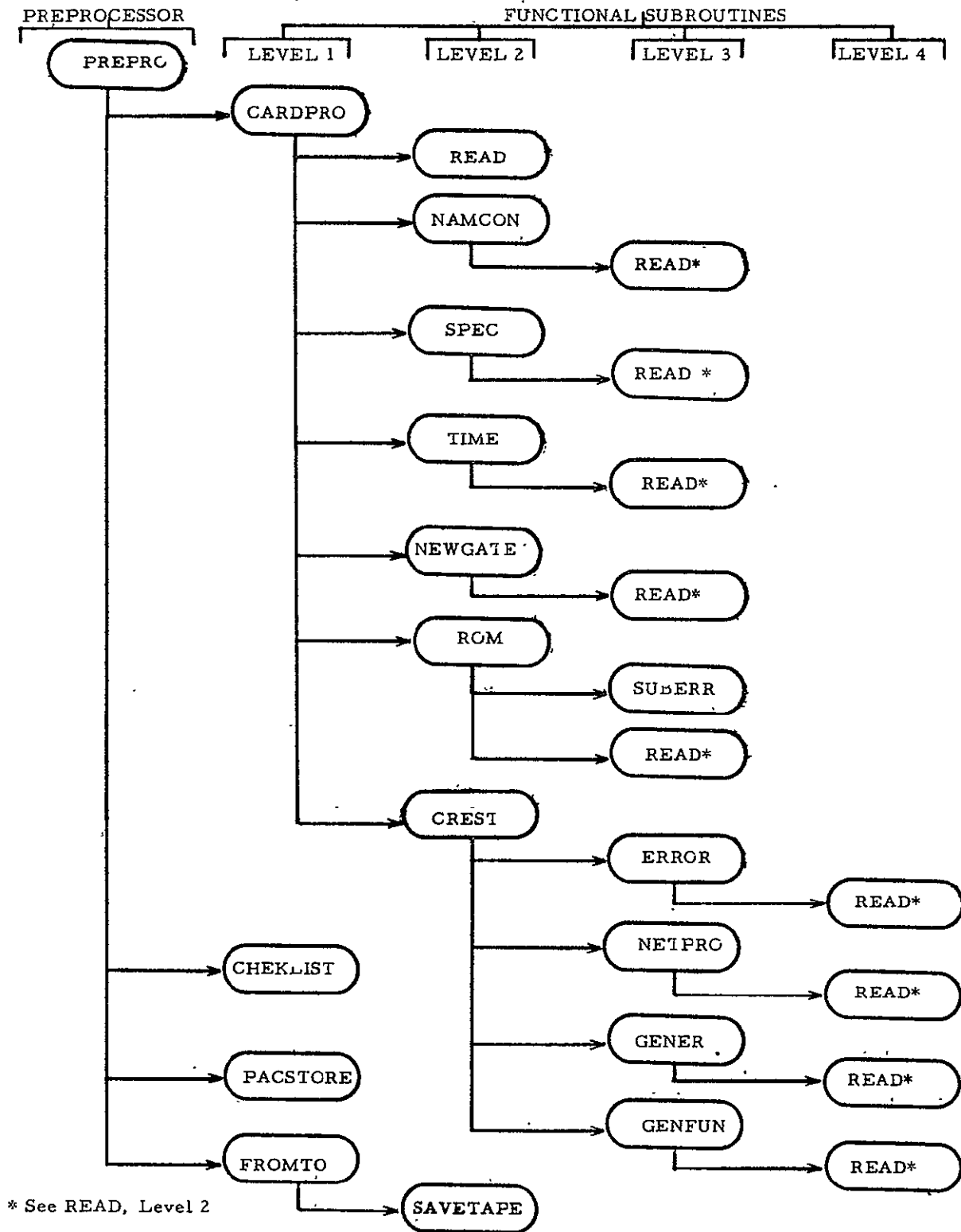


Figure 2-1

### 2.2.2 CHEKLIST: Error Check and Data List Subroutine - Level 1

After the data cards are processed a CHEKLIST function is performed to identify any substitutions made for indeterminate data or any input data errors. The CHEKLIST function summarizes input errors, control options, specifications, constructed delay gates, and constructed inverters. In the actual program CHEKLIST includes:

- o MAIN - second major coding block
- o SUBROUTINES - HEAD1, HEAD2

### 2.2.3 PACSTORE: Array Packing and Storing Subroutine - Level 1

The PACSTORE subroutine sorts the processed input data, packs it according to the input requirements of the Simulator, and stores it in a disk or magnetic tape file for access by the Simulator. PACSTORE includes the actual program blocks:

- o MAIN - third major coding block
- o SUBROUTINE - PACK

### 2.2.4 FROMTO: Connectivity List Generation Subroutine - Level 1

The functional subroutine FROMTO generates a list of the load elements associated with each input element. In the actual program FROMTO includes:

- o MAIN - fourth major coding block
- o SUBROUTINES - HEAD1, HEAD2

### 2.2.5 READ: Data Record Reading and Identification Subroutine Level 2

The READ function reads a data record, identifies the record type, stores the record image on a scratch disk file, and prints the record image. READ includes the actual program blocks:

- o MAIN - record type decode and type routing instructions in the first major coding block.
- o SUBROUTINES - READ1, WRITE1, RECORD

## 2.2.6 NAMCON: Name and Control Option Processing Subroutine - Level 2

The NAMCON functional subroutine processes the title (NAME) card and the program control options (CONT) card. If the NAME card is omitted, the title "LOGIC SIMULATION" is substituted. If the control card is omitted, it is assumed that none of the program's options (IGNORE, FROMTO, NONCON, FOLLOW, or DEBUG) are desired. In the actual program NAMCON includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2, RECORD

## 2.2.7 SPEC: Circuit Specification Subroutine - Level 2

The SPEC subroutine identifies circuit specifications from a specifications (SPEC) record. The SPEC record is a required card that relays vital parameters to the three LOGSIM programs. The SPEC record defines the time unit, maximum simulation time, the number of gates excluding automatically constructed delays and inverters, and the number of generators and generator functions. In the actual program SPEC includes:

- o MAIN - part of the first major coding block
- o SUBROUTINES - HEAD1, HEAD2

## 2.2.8 TIME: Time Record Processing Subroutine - Level 2

The functional subroutine TIME performs all delay rise time (DTMR), delay fall time (DTMF), and decay time (DCTM) record processing. This logic interprets each time type, assigns each an index number, and generates the first three records of the Simulator input file:

- Record 1 = decay times
- Record 2 = fall delay times
- Record 3 = rise delay times

In the actual program TIME includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2

#### 2.2.9 NEWGATE: NEWGATE Specification Processing Subroutine - Level 2

The NEWGATE functional subroutine processes the newgate type specification records. In the actual program NEWGATE includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2

#### 2.2.10 ROM: Read-Only-Memory Specification Processing Subroutine - Level 2

The ROM functional subroutine processes the ROM specification records and constructs the next three records of the Simulator input file:

Record 4 = LOGSIM logic library truth tables; NEWGATE truth tables; keys to the start of each ROM truth table in Record 5.

Record 5 = ROM truth tables

Record 6 = Special code values for logic library and NEWGATE logic types.

In the actual program, the ROM functional subroutine includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2, PACK

#### 2.2.11 CREST: Element Description Records Processing Subroutine - Level 2

The functional subroutine CREST processes logic gate (NET) records, generator (GEN) records, and generator function (GENF) records. The processing includes data reading, sorting, routing, and validating. In the actual program CREST includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2

#### 2.2.12 SAVETAPE: Data Tape Generation Subroutine - Level 2

The SAVETAPE functional subroutine generates a data file to be passed to the Postprocessor Program. This file contains a parameter record, images of the Preprocessor input cards, and a list of all element names. In the actual program, the SAVETAPE function is contained in the fourth coding block of MAIN.

#### 2.2.13 SUBERR: Data Error and Substitution Subroutine - Level 3

The SUBERR functional subroutine performs the error checking involved in ROM specification processing. Substitutions are made for indeterminate data when possible. Error and substitution messages are always output. In the actual program SUBERR includes:

- o MAIN - part of the first major coding block which includes ROM processing error messages and the subsequent routing decisions.
- o SUBROUTINES - HEAD1, HEAD2

#### 2.2.14 ERROR: Indeterminate and Misplaced Data Skip Subroutine - Level 3

The function of ERROR is to inform the user when an indeterminate or misplaced data record has been encountered and ignored. ERROR includes the actual program parts:

- o SUBROUTINES - ERR1, HEAD1, HEAD2

#### 2.2.15 NETPRO: NET Data Processing Subroutine - Level 3

The functional subroutine NETPRO reads and verifies all logic element description (NET) records and outputs a block of sorted NET data to a temporary scratch file. After all NET data is processed, all automatically constructed delay gate and inverter data is output to the same temporary file. In the actual program NETPRO includes:

- o MAIN - NET record type routing decisions in first major coding block
- o SUBROUTINES - NETPRO, OUT1, BUILD, HEAD1, HEAD2

#### 2.2.16 GENER: Generator Data Processing Subroutine - Level 3

The functional subroutine GENER reads and verifies the generator description (GEN) records and outputs blocks of sorted GEN data to a scratch file. In the actual program GENER includes:

- o MAIN - GEN record type routing decisions in the first major coding block
- o SUBROUTINES - GENER, OUT2, HEAD1, HEAD2

#### 2.2.17 GENFUN: Generator Function Data Processing Subroutine - Level 3

The GENFUN functional subroutine reads and verifies the generator function description (GENF) records and outputs blocks of sorted GENF to a scratch file. In the actual program GENFUN includes:

- o MAIN - GENF record type routing decisions in first major coding block
- o SUBROUTINES - GENFUN, OUT3, HEAD1, HEAD2

### 2.3 Arrays and Variables

The following is a list of array and variable names used in the Preprocessor. Each name is accompanied by a detailed definition including size and relative storage location information. Notations such as: (I), (J), etc. indicate that the name refers to an array. When more than one variable or array is assigned to the same storage location or locations, a special attempt has been made to identify the point in the program where the overlapping occurs. The detailed structure of the disk files referenced in the following definitions are presented in Section 2.4.



Not mentioned in the variable list are a number of intermediate program variables (N, M, etc.) which are used repetitively as control variables, temporary counters, etc. These variables are obvious in the program listing and any definitions of these would be meaningless.

- IDATA (I)        dummy array; 12,500 words, to which most of the program's arrays are equivalent. After all of the input data is read, sorted, and output to scratch file #1, it is read back and packed to the proper Simulator input format. IDATA (I) is used to read in blocks of data from scratch storage. IDATA (I) is passed to the subroutine PACK, packed into an intermediate array LDATA (I), and returned to the main program as IDATA (I) in the proper packed form.
- IDECTM (I) -    array of decay time values; I = 1, total number of decay times. The maximum number of decay times is 255 because of the 8-bit byte orientation of the host computer. IDECTM occupies storage locations IDATA (J), J = 1, 256 and is not needed after it is output to the Simulator input disk file, (record 1).
- IDTMF (I)    -    array of fall delay time values; I = 1, total number of fall delay times. The maximum number of fall delay times is 255 because of the 8-bit byte orientation of the host computer. IDTMF occupies storage locations IDATA (J), J = 257, 512 and is not needed after it is output to the Simulator input disk file (record 2).
- IDTMR (I)    -    array of rise delay time values; I = 1, total number of rise delay times. The maximum number of rise delay times is 255 because of the 8-bit byte orientation of the host computer. IDTMR occupies the storage locations IDATA (J), J = 513, 768 and is not needed after it is output to the Simulator input disk file (record 3).
- IDTYPF (I)    -    index to each gate's fall decay time; I = the gate number. The maximum size of IDTYPF is 250 words since only 250 NET descriptions are processed before the data is output to scratch file #1. After all the NET data is processed the array is no longer needed. It occupies the storage locations IDATA (J), J = 1751, 2000.

- IDTYPR (I) - index to each gate's rise decay time; I = the gate number. The maximum size of IDTYPR is 250 words since only 250 NET descriptions are processed before the data is output to scratch file #1. After all the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1501, 1750.
- IFECTM (I) - time of first generator or generator function change; I = the generator or generator function number. The maximum size of IFECTM is 250 words since only 250 GEN or GENF descriptions are processed before the data is output to scratch file #1. After all the GEN and GENF data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1001, 1250.
- IFECVL (I) - logic state after first generator or generator function change; I = the generator or generator function number. The array elements may have the values 1 (OFF) or 2 (ON). The maximum size of IFECVL is 250 words since only 250 GEN or GENF descriptions are processed before the data is output to scratch file #1. After all the GEN and GENF data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1251, 1500.
- IGATYP (I) - index to each gate's logic type; I = the gate number. Each array element may have any value from 1 to the number of logic gate types (library, NEWGATE, and ROMs). The maximum size of IGATYP is 250 words since only 250 NET descriptions are processed before the data is output to scratch file #1. After the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 501, 750.
- IGENTM (I) - list of generator change time values; I = number of generator changes. The maximum value of IGENTM is 4000, which limits the number of generator changes to 4000 for each block of 250 generators. After all the GEN data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1751, 4750.
- IGENVL (I) - list of generator state changes; I = 1, number of generator changes. The array elements may have the values 1 (OFF) or 2 (ON). The maximum size of IGENVL is 4000 which limits the number of generator changes to 4000 for each block of 250 generators. After all GEN data

is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 4751, 8750.

- IGNORE - control option flag for IGNORE option; 1 = option chosen, 0 = option not chosen.
- INAME (I) - list of input gate names, double precision; I = 1, number of inputs. The maximum size of INAME is 4000 names, which limits the inputs list to 4000 names for each block of 250 NET descriptions. After the NET data is processed the array is no longer needed. It occupies storage locations IDATA (J), J = 2251, 6250.
- INAMED (I) - list of delay input gate names, double precision; I = 1, number of delay gates. The maximum size of INAMED is 500 names, which limits the number of delay gates to 500 for each block of 250 NET descriptions. After the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 9251, 10250.
- INAMEI (I) - list of inverter input gate names, double precision; I = 1, number of inverters. The maximum size of INAMEI is 500, which limits the number of inverters to 500 for each block of 250 NET descriptions. After all NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 7251, 8250.
- INCARD - error counter for indeterminate input data types.
- INDEX - input buffer for a delay or decay time index.
- INNOTE - error counter for indeterminate notations in input data records.
- INPTKY (I) - key to start of each gate's inputs in the input names list; I = gate number. The maximum size of INPTKY is 250, since only 250 NET descriptions are processed before the data is output to scratch file #1. After all NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 2001, 2250.

- INVERT (I) - list of inverter names, double precision; I = 1, number of inverters. The maximum size of INVERT is 500, which limits the number of inverters to 500 for each block of 250 NET descriptions. After all the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 6251, 7250.
- IOVAL (I) - array of initial states of logic gates, generators, or generator functions; I = element number. Each array element may have a value of 0 (indeterminate), 1 (OFF), or 2 (ON). The maximum value of IOVAL is 250, since only 250 NET, GEN, or GENF descriptions are processed before the data is output to scratch file #1. After all NET, GEN, and GENF data are processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 751, 1000.
- ITPVLTL (I) - array of special code values for the logic library and NEWGATE types; I = logic type number. The maximum size of ITPVLTL is 46, which limits the number of permissible NEWGATE specifications to 30. The array is no longer needed after it is output to the Simulator input disk file (record 6). It occupies storage locations LOADS (J), J = 887, 932.
- ITRUTH (I, J)- array of NEWGATE truth tables; I = 1, 8 and J = 1, number of NEWGATES. The maximum size of J is 30 since only 30 NEWGATES are permitted. The array is no longer needed after it is output to the Simulator input disk file (record 4). It occupies storage locations IDATA (K), K = 4001, 4240.
- ITYPEF (I) - index to each gate's fall delay time; I = the gate number. The maximum size of ITYPEF is 250 words since only 250 NET descriptions are processed before the data is output to scratch file #1. After all the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1251, 1500.
- ITYPER (I) - index to each gate's rise delay time; I = the gate number. The maximum size of ITYPER is 250 words since only 250 NET descriptions are processed before the data is output to scratch file #1. After all the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 1001, 1250.

- JCODE1 (I) - permissible NEWGATE special code input characters; I = 1, 6. Each array element is one alphanumeric character. The array is initialized in a DATA statement.
- JCODE2 (I) - permissible NEWGATE truth table input characters; I = 1, 7. Elements 1, 2, 3, and 6 are the permissible ROM truth table input characters. Each array element is one alphanumeric character. The array is initialized in a DATA statement.
- JCODE3 (I) - all possible NEWGATE truth table entry and special code values; I = 1, 7. Each array element is a one-digit hexadecimal number. The array is initialized in a DATA statement.
- JDATA (I) - dummy array, double precision, 200 words. JDATA is used as a buffer array for passing data to or from disk files. It occupies storage location IDATA (J), J = 10001, 10200.
- JN (I) - four word alphanumeric array spelling LOGIC SIMULATION. It is used as a substitute page header title if a title is not specified by a NAME card. The array is initialized in a DATA statement.
- JNONE - four blank characters initialized in a DATA statement.
- JTRUTH (I) - logic library truth table array; I = 1, 16. Each array element is eight hexadecimal digits as initialized in a DATA statement.
- KARDR (I) - the number of cards in each ROM specification; I = the number of the ROM specification. The maximum size of KARDR is 100, the maximum number of ROM specifications. The array is no longer needed after the ROM data is output to the Simulator input disk file (records 4 and 5). It occupies storage locations IDATA (J), J = 4401, 4500.
- KCARD - input buffer for input data record; first four characters of the record.
- KDATA (I) - buffer array for a 72-character input data record, double precision; I = 1, 9. The array is not needed after all data is input. It occupies storage locations LOADS (J), J = 1, 18.

- KDEBUG - control option flag for DEBUG option; 1 = option chosen, 0 = option not chosen.
- KDTIME (I) - array of decay time index; I = 1, total number of decay times. The maximum number of decay times is 255 because of the 8-bit byte orientation of the host computer. It occupies storage locations LOADS (J), J = 531, 786, and is not needed after the delay gate data is output to scratch file #1.
- KEY (I) - key to start of each record in the Simulator input disk file and simulator option flags; 36 words. KEY is output to the Simulator input disk file as the last record on the file.
- KEYBUG - debug option flag; the flag is passed to the Simulator to determine the type of Simulator debug data to be output.
- KFILE - number of generator function data blocks on scratch file #1.
- KFOLLOW - control option flag for FOLLOW option; 1 = option chosen, 0 = option not chosen.
- KFTIME (I) - array of fall delay time indexes; I = 1, total number of fall delay times. The maximum number of fall delay times is 255 because of the 8-bit byte orientation of the host computer. It occupies storage locations LOADS (J), J = 275, 530, and is not needed after the delay gate data is output to scratch file #1.
- KODE1 - input buffer for NEWGATE special code character.
- KODE2 (I) - input buffer array for NEWGATE truth tables; I = 1, 8. It occupies IDATA (J), J = 772, 779, and is not needed after the NEWGATE input data processing is complete.
- KONECT - control option flag for the FROMTO option; 1 = option chosen, 0 = option not chosen.
- KONTIN - input buffer for the ROM specifications card number.

- KRTIME (I) - array of rise delay time indexes, I = 1, total number of rise delay times. The maximum number of rise delay times is 255, because of the 8-bit byte orientation of the host computer. It occupies storage locations LOADS (J), J = 19, 274 and is not needed after all of the delay gate data is output to scratch file #1.
- LDATA (I) - dummy array; 4000 words. LDATA is used as a buffer array for transferring data to and from disk files and as an intermediate array in the subroutine PACK. It occupies storage locations IDATA (J), J = 8001, 12000.
- LEND - end of input data flag. LEND = 0 until all data cards have been read and is set to 1 on a READ statement END return indicating all cards have been read.
- LFIELD - number of generator data blocks on scratch file #1.
- LINE - output printing line counter. When LINE  $\geq$  50, a new page is started with a page heading.
- LOADS (I) - list of gate load numbers; I = 1, number of loads: The maximum size of LOADS is 1000 since the total list of loads is output to scratch file #2, 1000 at a time.
- LOGGEN (I) - the start of each generator's changes in the generator times and states array and the start of each generator function's change sequence in the total generator change sequence; I = generator or generator function number. The maximum size of LOGGEN is 250 since only 250 GEN or GENF descriptions are processed before the data is output to scratch file #1. It occupies storage locations IDATA (J), J = 1501, 1750, and is not needed after all GEN and GENF data has been output to scratch file #1.
- LOG (I) - array of gate type names; I = the gate type number, in the order in which they are defined (library, NEWGATE, and ROM). The logic library elements of LOG are initialized in a DATA statement. The maximum size is 146.
- LUNIT - input logical unit number. LUNIT = 105 unless changed by a CHANGE TO input data record.

- MAXTIM - total logic time for which simulation is to occur.
- MROMTB (I) - ROM truth table entries array; I = 1, total number of truth table entries. The maximum size of MROMTB is 2000, because the data is packed and output to scratch file #2 in blocks of 2000. It occupies storage locations IDATA (I), I = 2001, 4000 and is no longer needed after all ROM specifications have been processed.
- NAMDEL (I) - list of delay gate names, double precision; I = 1, number of delay gates. The maximum size of NAMDEL is 500, which limits the number of delay gates to 500 for each block of 250 NET descriptions. After all NET data is processed the array is no longer needed. It occupies storage locations IDATA (J), J = 8251, 9250.
- NAME (I) - element names array, double precision; I = element number. The maximum size of NAME is 5000 names. It occupies storage locations IDATA (J), J = 1, 10000.
- NAMROM - input buffer for ROM gate type name.
- NCARDS - total number of input data records.
- NCONT - ROM specification continuation card counter.
- NDATA (I) - dummy array, double precision, 1000 double words. NDATA is used as a buffer array to pass double word data to and from disk files. It occupies storage locations IDATA (J), J = 6001, 8000.
- NDE - number of delay gates in a data block on scratch file #1.
- NDEL (I) - number of delay gates in each block of NET data on scratch file #1; I = the data block number. The maximum size of NDEL is 20 which allows (20 x 250) 5000 NET descriptions. It occupies storage locations LOADS (J), J = 847, 866, and is no longer needed after delay gate construction is completed.
- NDT - total number of decay times specified.
- NEDE - gate number of last delay gate.
- NEGF - gate number of last generator function.



- NEGN - gate number of last generator.
- NEIN - gate number of last inverter.
- NELG - gate number of last library or NEWGATE type logic gate.
- NEROM - gate number of last ROM type gate.
- NEW - input buffer for NEWGATE logic type name.
- NEXT - input data record type routing flag.
- NFALL (I) - index to each delay gate's fall delay time; I = delay gate number. The maximum size of NFALL is 500, which limits the number of delay gates to 500 for each block of 250 NET descriptions. After all NET data is processed the array is no longer needed. It occupies storage locations IDATA (J), J = 10751, 11250.
- NFILE - number of NET data blocks on scratch file #1.
- NFT - number of fall delay times specified.
- NGATES - number of gates declared.
- NGENS - number of generators and generator functions declared.
- NGF - generator function counter.
- NGFS (I) - the number of entries in the generator function sequence of each GENF data block on scratch file #1. The maximum size of NGFS is 20 which allow (20 x 250) 5000 GENF descriptions. It occupies storage locations LOADS (J), J = 787, 806.
- NGN - generator counter.
- NGNC (I) - the number of generator changes in each GEN block of data on scratch file #1. The maximum size on NGNC is 20 which allows (20 x 250) 5000 GEN descriptions. It occupies storage locations LOADS (J), J = 807, 826.
- NGT - number of logic gate types.

- NHEAD (I) - 72 character page header title; I = 1, 18. NHEAD is formed from the data on the title (NAME) record.
- NIN - number of inverters in a data block on scratch file #1.
- NINPUT (I) - number in the input list in each block of NET data on scratch file #1; I = the block number. The maximum size of NINPUT is 20 which allows for (20 x 250) 5000 NET descriptions. It occupies storage locations LOADS (J), J = 827, 846.
- NINV (I) - number of inverters in each block of NET data on scratch file #1; I = the block number. The maximum size of NINV is 20 which allows for (20 x 250) 5000 NET descriptions. It occupies storage locations LOADS (J), J = 867, 886 and is no longer needed after inverter construction is completed.
- NLG - number of logic gates in a block of NET data on scratch file #1.
- NLNG - total number of library and NEWGATE logic types.
- NNAMES - number of network elements declared.
- NNG - number of NEWGATE specifications.
- NOCARD - error counter for missing essential input data records.
- NOENUF - error counter for missing ROM continuation cards.
- NOMANY - error counter for extra ROM continuation cards based on number of inputs.
- NONCON - control option flag for NONCON option; 1 = option chosen, 0 = option not chosen.
- NONO - error counter for errors prohibiting simulation.
- NRG - number of ROM specifications.

- NRIN (I) - number of inputs of each ROM gate type; I = the ROM gate type number. The maximum size of NRIN is 100 limiting the number of ROM specifications to 100. It occupies storage locations LOADS (J), J = 970, 1069.
- NRINPT - input buffer for number of inputs of an ROM gate type.
- NRISE (I) - index to each delay gate's rise delay time; I = delay gate number. The maximum size of NRISE is 500 which limits the number of delay gates to 500 for each block of 250 NET descriptions. After the NET data is processed, the array is no longer needed. It occupies storage locations IDATA (J), J = 10251, 10750.
- NROM - ROM type logic gate counter.
- NRT - number of rise delay times specified.
- NRTE (I) - key to the start of each ROM's truth table in the combined ROM truth table array; I = the ROM gate number. The maximum size of NRTE is 100, allowing only 100 ROM specifications. It occupies storage locations IDATA (J), J = 4241, 4340, and is no longer needed after it is output to the Simulator input disk file (record 4).
- NRTT (I) - input buffer for ROM truth table entries; I = 1, 56. NRTT occupies storage locations IDATA (J), J = 4341, 4396, and is no longer needed after all ROM specification input data is processed.
- NSCALE - four character alphanumeric time unit on timing diagram.
- NTDE - total number of delay gates constructed.
- NTG - total number of network elements.
- NTGFS - total number of all generator function sequences.
- NTGNC - total number of generator changes.
- NTIN - total number of inverters.
- NTINPT - total number of gate inputs.

## 2.4 Secondary Data Storage

The Preprocessor makes extensive use of secondary storage. The program creates three temporary files, the Simulator input file, and part of the Postprocessor input file. The following discussion illustrates the contents and structure of each of these data files.

### 2.4.1 Temporary Data Files

The Preprocessor creates three scratch files to store sorted input data until all the input data is read. The number and length of each record in each file depends on the size and complexity of the network being simulated.

#### 2.4.1.1 Scratch File #1

Scratch File #1 is used for storing the data from NET, GEN, and GENF descriptions. Table 2-1 illustrates the structure and array content. The array subscript values illustrated refer to the maximum array dimensions.

#### 2.4.1.2 Scratch File #2

Scratch File #2 is used throughout preprocessing for temporary storage requirements resulting from overflow conditions; for example, if the array MROMTB (I) exceeds 2000 the first 2000 entries are packed and stored in file #2. Hence the frequency with which it is used depends on the complexity of the network being simulated. Table 2-2 illustrates the data which may be stored on file #2.

#### 2.4.1.3 Scratch File #3

Scratch File #3 is used for the temporary storage of the input data record images, and the element names list. This data must be stored until a Postprocessor input parameters record can be constructed.

### 2.4.2 Simulator Input Data File

The only Simulator input is the data file created by the Preprocessor. Therefore, this file must contain all of the data necessary to simulate the network. The file is organized into 24 records. The first

SCRATCH FILE #1 CONTENT AND STRUCTURE

NET DATA BLOCK		
1	NAME (250)	<p>NET Data Blocks</p> <ul style="list-style-type: none"> <li>- each block contains 11 records.</li> <li>- there may be 1 to 20 blocks.</li> <li>- each block corresponds to 250 NET descriptions which may contain less than 250 NETs.</li> <li>- each block is constructed by one pass through SUBROUTINE OUT1.</li> </ul>
2	IOVAL (250)	
3	INAME (4000)	
4	ITYPER (250)	
5	ITYPEF (250)	
6	IDTYPR (250)	
7	IDTYPF (250)	
8	INPTKY (250)	
9	IGATYP (250)	
10	(NAMDEL (I), INAMED (I), NRISE (I), NFALL (I) ) I = 1, 500	
11	(INERT (I), INAME (I), I = 1, 500	

DELAY GATE BLOCK		
1	NAMDE L from all NET blocks	<p>Delay Gate Data Block</p> <ul style="list-style-type: none"> <li>- the block contains 7 records.</li> <li>- there may be only 1 block; no block if there are no delay gates.</li> <li>- the block is constructed by SUBROUTINE BUILD.</li> </ul>
2	initial values	
3	INAMED from all NET blocks	
4	NRISE from all NET blocks	
5	NFALL from all NET blocks	
6	rise decay time indices	
7	fall decay time indices	

INVERTER DATA BLOCK		
1	INVERT from all NET blocks	<p>Inverter Data Block</p> <ul style="list-style-type: none"> <li>- the block contains 3 records.</li> <li>- there may be only 1 block, no block if there are no inverters.</li> <li>- the block is constructed by SUBROUTINE BUILD</li> </ul>
2	initial values	
3	INAMEI from all NET blocks	

ORIGINAL PAGE IS  
OF POOR QUALITY

Table 2-1

SCRATCH FILE #1 CONTENT AND STRUCTURE (Cont'd)

GEN DATA BLOCK		
1	NAME (250)	<p>GEN Data Blocks</p> <ul style="list-style-type: none"> <li>- each block contains 11 records.</li> <li>- there may be 0 to 20 blocks.</li> <li>- each block corresponds to 250 GEN descriptions except the last block which may contain less than 250 GENs.</li> <li>- each block is constructed by one pass through SUBROUTINE OUT2.</li> </ul>
2	IOVAL (250)	
3	LOGGEN (250)	
4	IFECTM (250)	
5	IFEOVL (250)	
6	IGENTM (4000)	
7	IGENVL (4000)	

GENF DATA BLOCK		
1	NAME (250)	<p>GENF Data Blocks</p> <ul style="list-style-type: none"> <li>- each record contains 11 records.</li> <li>- there may be 0 to 20 blocks.</li> <li>- each block corresponds to 250 GENF descriptions except the last block which may contain less than 250 GENFs.</li> <li>- each block is constructed by one pass through SUBROUTINE OUT3.</li> </ul>
2	IOVAL (250)	
3	LOGGEN (250)	
4	IFECTM (250)	
5	IFECVL (250)	
6	<p>Word #1 - number of GENF sequence repetitions.</p> <p>Word #2 - time interval between changes.</p> <p>Word #3 - number in change sequence.</p> <p>Word #4 to n- change sequence packed 4 bits/state</p> <p>- All repeated for each GENF</p>	

Table 2-1 continued

SCRATCH FILE # 2 CONTENT AND STRUCTURE

CONTENT	STRUCTURE
MROMTB (I)	2000 words packed 125 words/record.
ITYPEF (I), NFALL (J)	from NET and Delay Gate blocks on file #1, 1000 words packed 250 words/record.
ITYPER (I), NRISE (J)	from NET and Delay Gate blocks on file #1, 1000 words packed 250 words/record.
IDTYPF (I)	from NET blocks on file #1, 1000 words packed 250 words/record.
IDTYPR (I)	from NET blocks on file #1, 1000 words packed 250 words/record.
IGENVL (I)	from GEN blocks on file #1, 1000 words packed 125 words/record.
IFECTM (I) IFECVL (I)	from GEN and GENF blocks of file #1, one record each.
IGATYP (I) LOGGEN (J)	from NET blocks and GEN and GENF block of file #1 respectively, one record of each.
All element names INPTKY (I) All input names All input gate numbers Key to start of each load in LOADS array. LOADS (I)	- one record. from all NET blocks on file #1, one record - 100 names/record. - 100 words/record. - one record. - 1000 words/record.

Table 2-2

word of each of the first 23 records is the number of words in the record. The 24th record is a key array to the position and size of the other records and a list of network parameters. Table 2-3 illustrates the organization of the Simulator input data file.

### 2.4.3 Postprocessor Input Data File

The first part of the Postprocessor input data file is created by the Preprocessor. This data file contains:

- Parameters record - including a four-character time unit, NSCALE; the number of Preprocessor input data records, NCARDS; the total number of network elements, NTG; and the maximum Simulation time, MAXTIM.
- Preprocessor input data record images, one image per record; transferred from scratch file #3. The number of records = NCARDS.
- Element names, one name per record, transferred from scratch file #3. The number of records = NTO.



SIMULATOR INPUT DISK FILE ORGANIZATION

RECORD	CONTENT
1	1-word decay time values, IDECTM.
2	1-word fall delay time values, IDTMF.
3	1-word rise delay time values, IDTMR.
4	8 4-bit entries for each logic library truth table, JTRUTH. 8-4bit entries for each NEWGATE truth table, ITRUTH, and 1 word indices to the start of each ROM truth table in record 5, NRTE.
5	2-bit ROM truth table states, MROMTB.
6	4-bit special code values for logic library and NEWGATE logic types, ITPVLT.
7	1-byte index to each gate's fall delay time, ITYPEF, NFALL.
8	1-byte index to each gate's rise delay time, ITYPER, NRISE.
9	1-byte index to each gate's fall decay time, IDTYPF.
10	1-byte index to each gate's rise decay time, IDTYPR.
11	1-word generator change times, IGENTM.
12	4-bit generator state changes, IGENVL.
13	1-word number of repetitions, 1-word time interval, 1-word sequence pointer and limit, list of 4-bit state sequence entries, repeated for each generator function.
14	list of half-word element numbers in the order in which future events are to occur. Generator and generator functions numbers are entered, all other entries = 0.
15	1-word time of next change for each element. First generator and generator function change times, IFECTM, are entered, all other entries = -1.

Table 2-3

SIMULATOR INPUT DISK FILE ORGANIZATION (Cont'd)

RECORD	CONTENT
16	4-bit state of next event for each element, first generator and generator function change states, IFECVL, are entered, all other entries = 0.
17	4-bit entry for the initial state of all network elements, IOVAL.
18	half-word reference to each gate's logic type, IGATYP and index to its associated data in record 4.
19	half-word index to each generator's data in records 11 and 12 and each generator function's data in record 13, LOGGEN.
20	half-word element number for each element in the list of inputs.
21	half-word index to the start of each gate's inputs in the list of inputs, INPTKY.
22	half-word element number for each element in the list of loads, LOADS.
23	half-word index to the start of each element's loads in the list of loads.
24	<p>KEY (I) array:</p> <p>I = 1, 23 - key to start of records 1 through 23.</p> <p>I = 24 - element number of last non-ROM type logic gate, NELG.</p> <p>I = 25 - element number of last ROM type logic gate, NEROM.</p> <p>I = 26 - element number of last delay gate, NEDE</p> <p>I = 27 - element number of last inverter, NEIN.</p> <p>I = 28 - element number of last generator, NEGN.</p> <p>I = 29 - element number of last generator function, NEGF</p>

Table 2-3 continued

SIMULATOR INPUT DISK FILE ORGANIZATION (Cont'd)

RECORD	CONTENT
<p style="text-align: center;">24 cont'd</p>	<p>I = 30 - IGNORE option flag, IGNORE                      I = 31 - NONCON option flag, NONCON                      I = 32 - FOLLOW option flag, KFOLOW                      I = 33 - end of key, number of elements                                +40.                      I = 34 - maximum simulation time,                                MAXTIM                      I = 35 - number of generators and gener-                                ator functions.                      I = 36 - DEBUG option flag, KDEBUG</p>

Table 2 - 3 continued

### 3. SIMULATOR

The Simulator portion of the LOGSIM Program simulates the operation of a logic network based on the network description generated by the Preprocessor and outputs a sequence of logic events describing the operation for interpretation by the Postprocessor.

The Simulator operates using the concept of a "Future Events Chain" (FEC). When an event occurrence is recognized, the program examines the effect which the state change has on other elements in the circuit and schedules future events accordingly. The future events are stored in the FEC, and the current time is advanced to the time of the next scheduled event. This concept will be referenced extensively in the following Simulator discussions.

#### 3.1 Program Structure

Figure 3-1 illustrates the functional structure of the Simulator (SIMUL) Program. The program is structured into five functional levels with the first level of subroutines corresponding to the following major program functions:

- (1) Initialization of element states
- (2) Simulation of logic events
- (3) Termination of the simulation

Appendix B contains the detailed flowchart of the Simulator Program.

#### 3.2 Subroutine Description

Each subroutine illustrated in Figure 3-1 represents a particular function of the program and a specific block of coding in the main program. The only actual subprograms of the Simulator are short functions used for bit manipulation. These subprograms are frequently used Boolean functions which are important for packing and unpacking data, but are not represented as major functional program blocks in this discussion.

##### 3.2.1 INITIAL: Gate Initialization Subroutine - Level 1

The functional subroutine INITIAL determines the initial state of as many network elements as possible. If all indeterminate states

SIMULATOR (SIMUL) PROGRAM STRUCTURE

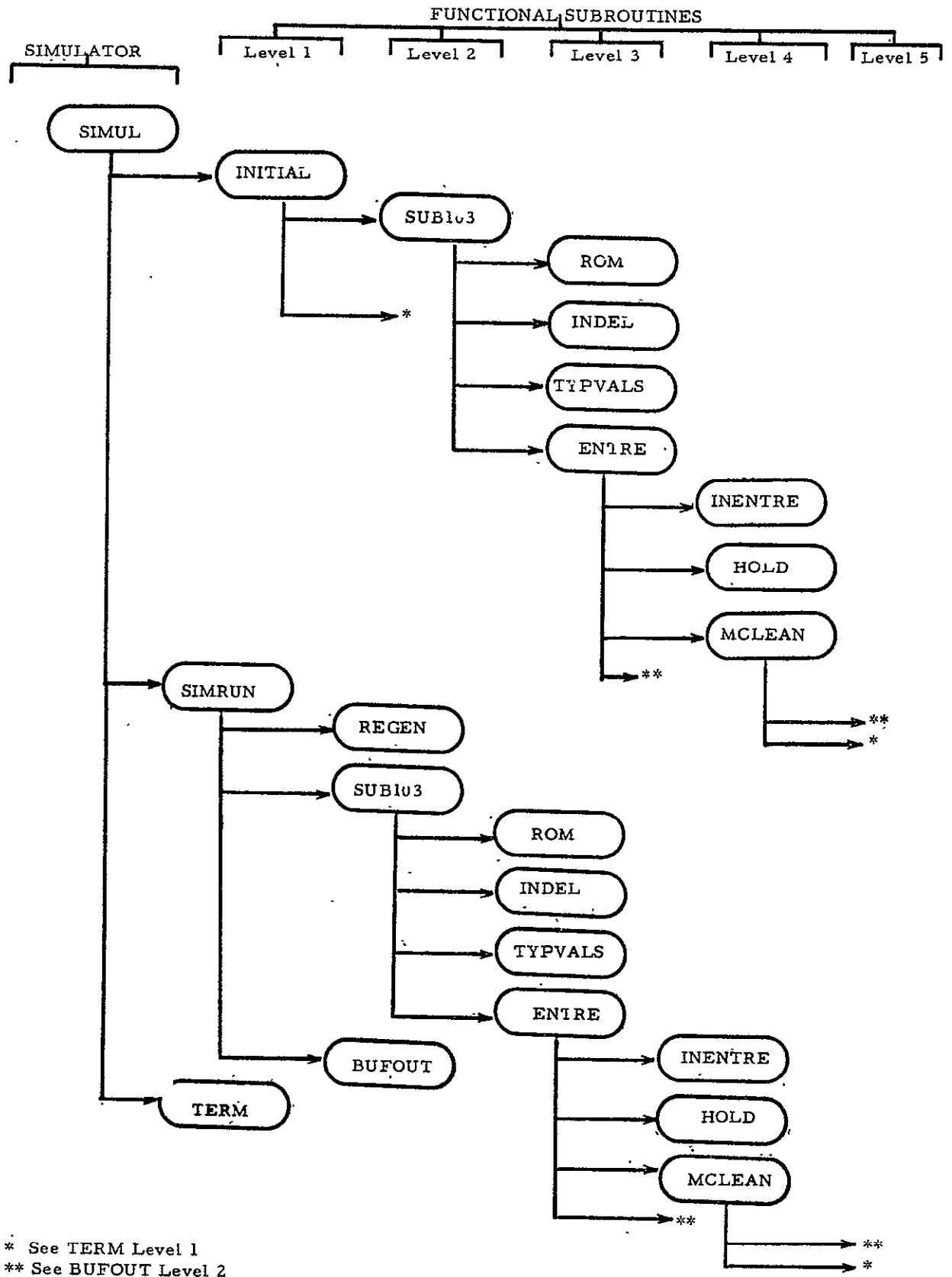


Figure 3 - 1

ORIGINAL PAGE IS  
 OF POOR QUALITY

cannot be resolved, the Simulator action taken depends on the value of the IGNORE option flag. The initial states determined by the Simulator are examined for any inconsistencies with those specified by the user, and any action taken depends on the value of the NONCON option flag.

### 3.2.2 SIMRUN: Event Simulation Subroutine - Level 1

The functional subroutine SIMRUN performs the operations necessary to simulate the network operation from zero time to the maximum simulation time, MAXTIM.

### 3.2.3 TERM: Termination Subroutine - Level 1

The functional subroutine TERM outputs any data in the Simulator output data buffers and brings the simulation to an orderly terminate.

### 3.2.4 SUB103: Logic State Determination Subroutine - Level 2

The functional subroutine SUB103 determines the output value of a network element. It is first utilized during initialization to resolve indeterminate initial gate output levels. During simulation SUB103 predicts future gate output changes to be entered in the FEC.

### 3.2.5 REGEN: Generator Events Restoration Subroutine - Level 2

When a generator or generator function event occurs, the subroutine REGEN locates that element's next event in the input data array. If there is a future event scheduled it is then entered into the FEC.

### 3.2.6 BUFOUT: Event Output Subroutine - Level 2

In the operation of BUFOUT any data which is to be output to the Postprocessor input data tape is entered into one of two buffers. Data is entered into the first available buffer, while the other buffer may be busy with output. After the buffer is filled, the intrinsic subroutine BUFFER OUT is called and the data is output to tape.

### 3.2.7 ROM: Read-Only-Memory State Determination Subroutine - Level 3

The functional subroutine ROM determines the output level of a read-only-memory element by searching its truth table according to the input levels. An attempt is made to resolve the output for any indeterminate inputs only if requested by the FOLLOW option.

### 3.2.8 INDEL: Delay Gate and Inverter State Determination Subroutine - Level 3

The subroutine INDEL determines the output levels for delay gates and inverters. This is done by examining the level of the gate's input and scheduling its output the same as its current input if it is a delay gate, or setting its current value at the current input complement value if it is an inverter.

### 3.2.9 TYPVALS: Logic Gate State Determination Subroutine - Level 3

The functional subroutine TYPVALS determines the output of logic elements of the logic library or NEWGATE type. The determination is made on the basis of the type of gate and the input levels. The gate type special code and 8-character truth-table is accessed from the input data array. TYPVALS attempts to evaluate outputs even when indeterminate inputs exist. If the output state cannot be determined, it is set to indeterminate.

### 3.2.10 ENTRE: Event Storage Subroutine - Level 3

The subroutine ENTRE enters an element's new value into the NIT list or schedules a future event by an entry in the FEC. It determines the event delay, and identifies holding events and spike conditions. When a spike condition cancels a future event, the FEC entry is deleted and a spike message is output.

### 3.2.11 INENTRE: Initial Conditions Storage Subroutine - Level 4

The INENTRE subroutine enters initial gate levels into an initial conditions array. A holding initial event is considered indeterminate. If an initial conditions conflict occurs a message is output.

### 3.2.12 HOLD: Holding Mode Event Processing Subroutine - Level 4

When a holding mode event is encountered, the subroutine HOLD examines the event data, determines if the gate's output should be held at its current state, determines its current state, and finds the proper rise or fall decay time. If the state is to be held, the event is entered in the FEC.

### 3.2.13 MCLEAN: FEC Clearing Subroutine - Level 4

The functional subroutine MCLEAN maintains the FEC storage area, by insuring against an overflow. If the FEC is filled, any positions where events have been deleted are filled and any duplicate entries in the FEC are eliminated. If the FEC is still full, it is an indication that the simulation is attempting to process current events from a zero delay feedback loop. In this case a message is output and a normal termination occurs.

## 3.3 Arrays and Variables

The following is a list of array and variable names used in the Simulator Program. Each is defined in detail including any dimensional or equivalence information. Notations such as (I), (J), etc. refer to array names. Variable names which are used repetitively with different meanings are not included in this list.

- IBASE - address of a ROM truth table in the input array M (I).
- IBIT (I) -  $I = 0, 16$ , where  $IBIT (I) = 2^I$ ; initialized in a DATA statement and used for masking the debug word KDEBUG.
- IDECTM - a decay time value.
- IDTMF - a fall delay time value.
- IDTMR - a rise delay time value.
- IDTYPF - a gate's fall decay time index.
- IDTYPR - a gate's rise decay time index.



IER1	} error type flags used in error messages passed to tape; initialized in a DATA statement.
IER2	
IER3	
IER4	
IFECKY	- key to future events, entry in FEC.
IFECTM	- time of a gate's future event.
IFECVL	- a gate's future event value.
IGATE	- a gate number.
IGATS	- the number of the gate whose current event is being processed.
IGATYP	- the value identifying the current logic type and its truth table location.
IGENF	- a generator function event value scheduled at preprocessing.
IGENTM	- a generator event time scheduled at preprocessing.
IGENVL	- a generator event value scheduled at preprocessing.
IGNORE	- option flag for control option IGNORE set at preprocessing.
IHIS	- counter for the number of HIGH inputs to a gate.
INCLOC	- flag for a gate's clock value.
INDET (I)	- I = 0, 16, indicating the indeterminate inputs to a ROM.
INDETS	- counter for the number indeterminate inputs to a gate.
INFLAG	- a count of indeterminate initial states left after a pass at attempting to resolve indeterminate states during initialization.
INITIA	- a flag indicating whether or not initialization is taking place.

- INPTKY - address of a gate's input list.
- INPUTS - the number of a gate's inputs.
- IOBUFF - output buffer array, I = 1, 1000; IOBUFF is divided I = 1, 500 and I = 501, 1000 and used as two arrays to buffer blocks of output data.
- IOVAL - the initial state of a gate.
- IPRINT (I) - a small intermediate buffer array, I = 1, 6; used to store data temporarily until one of the output buffers is available.
- ISIGN - flag to indicate direction in binary search of FEC.
- ITER - a mask word initialized by a DATA statement; used by Boolean type intrinsic functions.
- ITPVAL - a logic library or NEWGATE type truth table or address of a ROM truth table.
- ITPVL T - a gate's special code identifying how its truth table is to be manipulated.
- ITYPEF - index to a gate's fall delay time.
- ITYPER - index to a gate's rise delay time.
- K (I) - input buffer array, I = 1, 28; for reading the last record on the Simulator input data file. Each of the first 23 entries of K (I) are equivalent to the variable name representing the data in each of the first 23 records on that file.
- KDEBUG - debug data print flag.
- KDLEND - the number of the last delay gate.
- KEYEND - size of the FEC.
- KFOLLOW - option flag for control option FOLLOW set.
- KGEND - the number of the last generator.

KGENFD - the number of the last generator function.  
 KGTEND - the number of the last inverter.  
 KLEAN - count on the number of passes through functional sub-  
 routine MCLEAN.  
 KONFLI - flag on conflicting initial states.  
 KROMND - number of the last ROM type logic gate.  
 KTYPE - indicator of a gate's special code.  
 KURTIM - current time during simulation.  
 KVLEND - number of the last non-ROM type logic gate.  
 LASTFG - count of indeterminate initial conditions prior to the  
 last pass at attempting to resolve indeterminate states.  
 LOADKY - address of a gate's loads list.  
 LOADS - number of a gate's loads.  
 LOGGEN - address of a generator or generator function event  
 scheduled at preprocessing.  
 LOWS - counter for the number of LOW inputs to a gate.  
 LPOINT - pointer to the start of data in the array IPRINT (I) to  
 be transferred to the output buffer IOBUFF (J).  
 L1 - address of first load in a gate's load list.  
 L2 - address of last load in a gate's load list.  
 L2BITS - a mask word, initialized by a DATA statement; used  
 by the Boolean type intrinsic functions.  
 M (I) - input buffer array for reading the first 23 records of  
 the Simulator input data file. I = 1, 20000 limiting  
 the size of the input data file to 20,000.  
 MASK I1 - a mask word, initialized by a DATA statement; used  
 by Boolean type intrinsic functions.

- MAXTIM - maximum simulation time.
- MCLEAN - variable statement number assignment.
- MROMTB - 16 2-bit ROM truth table entries.
- MSK (I) - mask array for reading ROM truth tables, I = 0, 16; contains the number of bits which a ROM word must be shifted to right justify the desired value.
- MSKCNT - counter on number of masks constructed for reading a ROM truth table.
- NIT - index to location of a current time event in the FEC.
- NONCON - option flag for control option NONCON set at preprocessing.
- NOWSR - flag used in processing 4th and 5th inputs of a CJKF type gate. NOWSR = 1 indicates output must change because either the 4th or 5th input is set.
- NPOINT - pointer on the last entry in the output buffer array being filled.
- NWORDS - number of entries in the intermediate array IPOINT (I) to be transferred to an output buffer.
- N1FLAG } BUFFER OUT operation error flags for the two output  
N2FLAG } buffer array.
- PNTRET - variable statement number assignment.
- SUBRET - variable statement number assignment.
- XI - floating point representation of increment used in binary search of FEC.

### 3.4 Debug Output

Detailed debug information can be obtained from a Simulation run by specifying the debug option to the Preprocessor. The Simulator debug word KDEBUG is set to indicate the type of debug output to be printed. Table 3-1 illustrates the use of debug word KDEBUG in the Simulator.

## SIMULATOR DEBUG WORD BIT INTERPRETATION

KDEBUG Bit No.	Description of Debug Output When Bit is Set
0	Data from Simulator input disk file is printed.
2	All data output to Preprocessor input data tape by the the simulator is printed.
3	All events scheduled for the current time are printed.
4	The gate number whose loads are about to be checked for future events is printed.
5	The current state of the gate which is about to be checked for a future event is printed
6	The future state of the gate which is about to be checked for a future event is printed.
7	The current FEC status is printed.
8	Status of the FEC clearing operation performed by the Subroutine MCLEAN is printed.
9	ROM Processing data is printed.
KDEBUG ≠ 0	Final gate initialization is printed.

Table 3 - 1

## 4. POSTPROCESSOR

The primary function of the Postprocessor is to output logic timing diagrams corresponding to the events simulated.

### 4.1 Program Structure

Figure 4-1 illustrates the functional structure of the Postprocessor (POSTOR) Program. The program is structured into two levels of functional subroutines with the first functional level corresponding to the four main program functions:

- (1) Processing input data cards.
- (2) Timing diagram setup.
- (3) Processing element initial conditions.
- (4) Processing simulation event.

Appendix C contains the detailed flowchart of the Postprocessor Program.

### 4.2 Subroutine Description

Each subroutine illustrated in Figure 4-1 represents a particular function of the program and may include a block of the main program coding and/or one or more actual program subroutines.

#### 4.2.1 POSTCARD: Card Processing Subroutine - Level 1

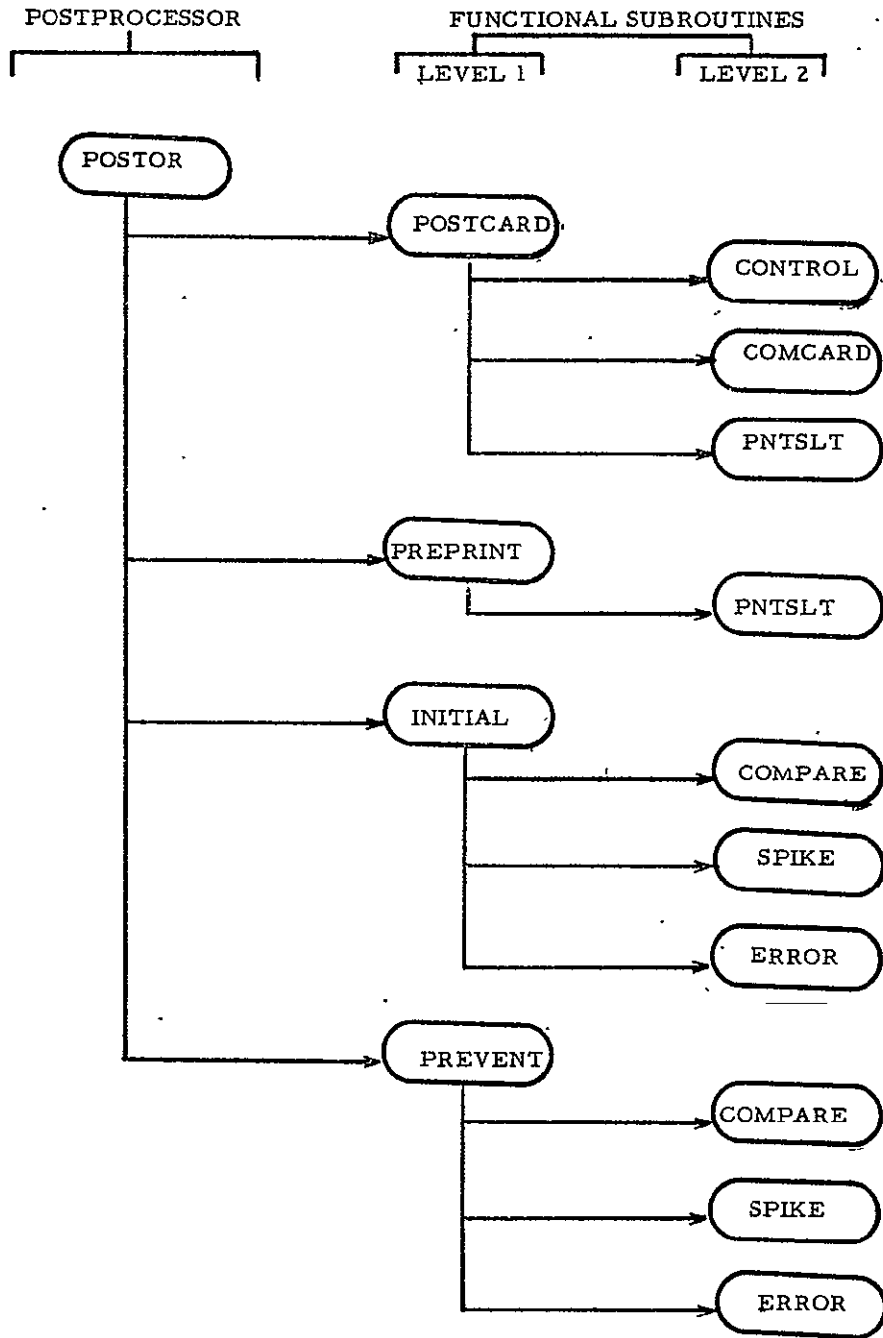
The functional subroutine POSTCARD processes all input data cards except PNT/SLOT cards. This subroutine also processes the data on the Postprocessor input data tape generated by the Preprocessor. In the actual program POSTCARD includes:

- o MAIN - first major coding block
- o SUBROUTINES - HEAD1, HEAD2, HEAD3, HEAD5.

#### 4.2.2 PREPRINT: Timing Diagram Setup Subroutine - Level 1

The PREPRINT subroutine outputs user reference data, and

POSTPROCESSOR (POSTOR) PROGRAM STRUCTURE



ORIGINAL PAGE IS  
OF POOR QUALITY

Figure 4-1

starts the printout of each timing diagram. In the actual program PREPRINT includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD7, HEAD8, HEAD9, HEAD10, HEAD11, HEAD13.

#### 4.2.3 INITIAL: Element Initial Conditions Processing Subroutine - Level 1

The functional subroutine INITIAL accesses the Postprocessor input data tape for the first events blocks containing element initial states output by the Simulator. The initial conditions are processed according to the INITIAL control option. In the actual program INITIAL includes:

- o MAIN - second major coding block
- o SUBROUTINES - HEAD1, HEAD2, HEAD12, HEAD11.

#### 4.2.4 PREVENT: Event Processing Subroutine - Level 1

The subroutine PREVENT begins with the first event following the initial conditions on the tape and processes the events in blocks of 250. A line of data is printed on the timing diagram for every time at which any event occurred. PREVENT includes the actual program parts:

- o MAIN - third major coding block
- o SUBROUTINES - HEAD11, HEAD14, OUTPUT.

#### 4.2.5 CONTROL: Control and Output Character and Shift Card Processing Subroutine - Level 2

The functional subroutine CONTROL reads and verifies the control (CONT), character (CRCT), and shift (SHFT) cards. In the actual program CONTROL includes:



- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2.

#### 4.2.6 COMCARD: Compare and Compare Function Card Processing Subroutine - Level 2

The subroutine COMCARD reads and verifies the level comparison (CMP) and compare function (CMPF) input data cards. COMCARD includes the actual program parts:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2.

#### 4.2.7 PNTSLT: PNT/SLOT Card Set Processing Subroutine - Level 2

The functional subroutine PNTSLT reads and verifies the PNT and SLOT data cards, one PNT/SLOT set at a time. In the actual program PNTSLT includes:

- o MAIN - part of first major coding block
- o SUBROUTINES - HEAD1, HEAD2, NOPNT, PNTSLT, PNT, SLOT.

#### 4.2.8 COMPARE: Level Comparison Processing Subroutine - Level 2

The COMPARE subroutine performs the level comparisons requested by the level comparison (CMP) and compare function (CMPF) cards. COMPARE includes the actual program parts:

- o MAIN - routing decision in 2nd and 3rd major coding blocks
- o SUBROUTINES - HEAD11, COMPARE

#### 4.2.9 SPIKE: Spike Notice Processing Subroutine - Level 2

The functional subroutine SPIKE searches the table of predicted spike conditions for any corresponding to the current event time. In the actual program SPIKE includes:

- o MAIN - routing decisions in 2nd and 3rd major coding blocks
- o SUBROUTINES - HEAD11, SPIKE.

#### 4.2.10 ERROR: Error Message Processing Subroutine - Level 2

The subroutine ERROR identifies error messages which are passed from the Simulator among the events, and processes them according to the error type. Spike predictions are included in the error messages passed. ERROR includes the actual program parts:

- o MAIN - routing decisions in 2nd and 3rd major coding blocks
- o SUBROUTINES - HEAD11, ERROR1, ERROR2, ERROR3

#### 4.3 Arrays and Variables

The following is a list of array and variable names used in the Postprocessor Program. Each is defined in detail including any array dimension information. Notations such as (I), (J), etc. refer to an array name. Not included in this list are a number of variable names which are used repetitively as control variables and temporary counters.

- IDATA (I) - input buffer array for preprocessor input data records, I = 1, 18.
- INFO - first word of a sequence of error data from the Simulator.
- INFO1 (I) - first word of any error message word pair; I = word pair number. The maximum value of I is 5 since no error message given by the Simulator exceeds 5 word pairs.
- INFO2 (I) - second word of any error message word pair; I = word pair number. The maximum value of I is 5 since no error message given by the Simulator exceeds 5 word pair.
- ITIME (I) - list of beginning time values for time slots to be printed. I = 1, 60, limiting the number of time slots per timing diagram to 60.

JCARDA - character card identifier; four alphanumeric characters - CRCT, initialized by a DATA statement.

JCARDB - shift card identifier; four alphanumeric characters - SHFT, initialized by a DATA statement.

JCARD1 - control card identifier; four alphanumeric characters - CONT, initialized by a DATA statement.

JERROR - error message word pair counter.

JNONE - four blank characters, initialized in a DATA statement.

JNUM - PNT card counter.

JOPTN1 - SPIKE option identifier; 4 characters - SPIK, initialized by a DATA statement.

JOPTN2 - PREPIN option identifier; 4 characters - PREP, initialized by a DATA statement.

JOPTN3 - POST option identifier; 4 characters - POST, initialized by a DATA statement.

JOPTN4 - COMSTOP option identifier; 4 characters - COMS, initialized by a DATA statement.

JOPTN5 - INITIAL option identifier; 4 characters - INIT, initialized by a DATA statement.

JSHFTL - left shift identifier; 1 character - L, initialized by a DATA statement.

JSHFTR - right shift identifier; 1 character - R, initialized by a DATA statement.

JTYPE1 - compare card identifier; 4 characters - CMPE, initialized by a DATA statement.

JTYPE2 - compare function card identifier; 4 characters - CMPF, initialized by a DATA statement.

JTYPE3 - PNT card identifier; 4 characters - PNT5, initialized by a DATA statement.

- JTYPE4 - SLOT card identifier; 4 characters - SLOT, initialized by a DATA statement.
- KCARDA  
 KCARDB  
 KCARD1  
 KCARD2  
 KCARD3  
 KCARD4  
 KCARD5 } input buffer for data card types, first four characters on card.
- KDATA (I) - input buffer array for 72-character image of any data card; double precision, I = 1, 9.
- KDRCTN (I) - input buffer array of left or right shifts corresponding to how each character on the timing diagram, KLEV (J), is to be shifted; I = character number. The Simulator outputs only three possible states, indeterminate, LOW, and HIGH, but the array is dimensioned at 16 allowing for 16 different states.
- KGATE (I) - array of element number to which each event, in a block of 250 events, corresponds; I = 1, 250.
- KINDER - error message type flag, may assume values from 1 to 5 with the following correspondence:
- 1 - conflict of initial states
  - 2 - spike
  - 3 - indeterminate initial states
  - 4 - a stable gate operation
  - 5 - termination

- KLEV (I) - array of characters used to represent each possible output state on the timing diagram; I = state number. There are three possible states: 1 - indeterminate, 2 - LOW, 3 - HIGH, which the Simulator outputs. However, KLEV is dimensioned at 16 allowing 16 possible states.
- KLEVEL (I) - array of state values to which each event in a block of 250 events corresponds; I = 1, 250.
- KLOOP - counter for number of initial conditions processed since previous line of initial conditions were printed under the INITIAL option. When KLOOP = 10 another line is to be printed.
- KMSTOP - option flag for COMSTOP control option; 1 = program to be terminated, 0 = program to continue.
- KNAME (I) - double precision input buffer array of element names to be printed on a timing diagram. It is limited to 40 names because only 40 columns are allowed per timing diagram.
- KNUM - number of gate outputs to be printed on a timing diagram.
- KOPTN1 }  
 KOPTN2 }  
 KOPTN3 } input buffers for control option names.  
 KOPTN4 }  
 KOPTN5 }
- KRCTRA - one character, \*, initialized in DATA statement.
- KRCTRH - one character, H, initialized in DATA statement.
- KRCTRL - one character, L, initialized in DATA statement.
- KSKIP - four characters, SKIP, initialized in DATA statement.
- KTIME (I) - array of time values to which each event, in a block of 250 events, corresponds; I = 1, 250.

- KWORD - last word of events block of 250 word pairs.
- KWORD1 (I) - array of first words of each word-pair in a block of 250 event word pairs; I = 1, 250.
- KWORD2 (I) - array of second words of each word-pair in a block of 250 event word pairs; I = 1, 250.
- KWORD3 (I) - input buffer for 250 event word pairs; I = 1, 500.
- LABORT - error termination flag; 1 = program to be terminated, 0 = program to continue.
- LDRCTN (I) - output character shift value array, corresponding to shift character array, KDRCTN (I); 1 = shift to the right, 0 = shift to the left.
- LEVEL (I) - present state of each network element. Dimensioned at 2000 allowing for simulation of 2000 elements.
- LINE - printed line counter. When  $LINE \geq 50$  a new printer page is started with the proper heading.
- LNAME (I) - print matrix names array, double precision; I = 1, 40 column numbers.
- LNUM - number of time slots in a set of SLOT cards.
- LOPTN1 - option flag for SPIKE control option; 1 = option chosen, 0 = option not chosen.
- LOPTN2 - option flag for PREPIN control option; 1 = option chosen, 0 = option not chosen.
- LOPTN3 - option flag for POSTIN control option; 1 = option chosen, 0 = option not chosen.
- LOPTN4 - option flag for COMSTOP control option; 1 = option chosen, 0 = option not chosen.
- LOPTN5 - option flag for INITIAL control option; 1 = option chosen, 0 = option not chosen.
- LSKIP (I) - array of columns to be left blank on timing diagram. Limited to 40 since only 40 columns of output are allowed.

LTIME - current simulation time value.  
 LUTIME - current simulation time value, same as LTIME.  
 MASK - 8-character, OOOOFFFF, mask value used in intrinsic function LAND to extract a half-word.  
 MAXTIM - maximum simulation time.  
 MCOMP (I) - double precision array of element names to be compared. CMP cards limited to 100.  
 MNUM - number of time slots on a slot card.  
 MOUT (I) - array of levels to which the output of each element named in MCOMP (I) is to be compared. CMP cards limited to 1000.  
 MOVE - number of bits to be shifted, used in intrinsic function ISL to extract half-words.  
 MTIME (I) - array of times at which the outputs of elements named in array MCOMP (I) are to be compared to the levels in the array MOUT (I). CMP cards limited to 100.  
 NAME (I) - double precision array of all element names; dimensioned to 2000.  
 NCARD - number of Preprocessor input data records on the Post-processor input data file following the heading record, NCARDS -1.  
 NCARDS - number of Preprocessor input data records on the Post-processor input data file.  
 NCOMP (I) - double precision array of element names involved in compare function. CMPF cards limited to 100.  
 NERROR - error processing flag;  $> 0$  error being processed,  $\leq 0$  no error being processed.  
 NHEAD (I) - 72 character printout page heading; I = 1, 18.  
 NJUMP (I) - time interval between comparisons for each compare function. CMPF cards limited to 100.

- NNAMES - number of network elements.
- NO - flag for missing PNT card; 1 = terminate, 0 = continue.
- NOMORE - flag for last timing diagram; 1 = last PNT/SLOT set, 0 = more PNT/SLOT sets.
- NONEXT - flag for last events data block; 1 = last block, 0 = more blocks.
- NORDER (I) - input buffer array for 32-character compare function sequence; I = 1, 32.
- NREP (I) - array of the number of repetitions of the sequence for each compare function. CMPF cards limited to 100.
- NSBEG (I) - beginning time of spike condition for each spike notice kept on file. A maximum of 100 spike notices are kept on file at any time.
- NSCALE - four character time unit to be printed above time column on timing diagram.
- NSEND (I) - end time of spike condition for each spike notice kept on file. A maximum of 100 spike notices are kept on file at any one time.
- NSEQ (I, J) - array of the 32-entry comparison sequence for each compare function; I = compare function number, and J = 1, 32; CMPF cards are limited to 100.
- NSGATE (I) - gate number corresponding to each spike notice kept on file. A maximum of 100 spike notices are kept on file at any one time.
- NSLEV1 (I) - originally scheduled state before each spike condition for each spike notice kept on file. A maximum of 100 spike notices are kept on file at any one time.
- NSLEV2 (I) - new scheduled state after spike condition for each spike notice kept on file; I = 1, 100.
- NSPIKE - number of spike notices on file.



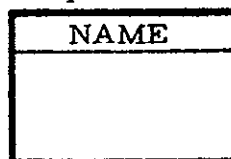
- NSTART (I) - starting time of each compare function. CMPF cards limited to 100.
- NUM1 - number of CMP cards.
- NUM2 - number of CMPF cards.
- NUM3 - number of PNT/SLOT card sets.
- NUTIME - first non-zero event time in an events block; = -1 when time = 0.
- NWORDS - number of words in error message to be read.
- DLEVEL (I) - a set of element outputs to be printed on one line of the timing diagram; I = the column number and is limited to 40.

## APPENDIX A

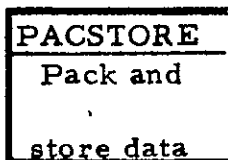
### LOGSIM PREPROCESSOR PROGRAM FLOWCHARTS

This Appendix presents the detail flowcharts of the LOGSIM Preprocessor Program. The flowcharts should provide sufficient explanation of the Preprocessor listing.

The "Picture on a Page" technique has been utilized, which allows the reader to study the flowcharts to the depth he desires. Each page is a complete representation of the area presented. Those functions that are expanded in more depth on subsequent sheets are identified with subroutine nomenclature blocks



For example, on page A-4, the block



indicates that

the activity defined by the block is discussed in more detail on a separate sheet with the entry PACSTORE (See page A-6).

Appendices A, B, and C refer to the following input/output data files by logical unit numbers:

- |    |                             |   |                       |
|----|-----------------------------|---|-----------------------|
| 1) | scratch file 1              | = | logical unit number 1 |
| 2) | scratch file 2              | = | logical unit number 2 |
| 3) | scratch file 3              | = | logical unit number 3 |
| 4) | Postprocessor<br>input file | = | logical unit number 7 |
| 5) | Simulator input<br>file     | = | logical unit number 9 |

Table A1 is an index to the flowcharts. Table A2 describes the flowchart symbol convention adhered to by these flowcharts and by the flowcharts presented in Appendices B and C.

# LOGSIM PREPROCESSOR PROGRAM

## FLOWCHART INDEX

	<u>Page</u>
Logic Simulation Preprocessor	A-4
Card Processing Subroutine	A-5
Error Check and Data List Subroutine	A-5
Arrays Pack and Store Subroutine	A-6
Connectivity List Generation Subroutine	A-6
Data Record Reading and Identification Subroutine	A-7
Circuit Specifications Subroutine	A-7
Name and Control Option Processing Subroutine	A-8
Time Record Processing Subroutine	A-9
Newgate Specification Processing Subroutine	A-10
Read-Only Memory Specification Processing Subroutine	A-11
Element Description Record Processing Subroutines	A-12
Data Tape Generation Subroutine	A-13
Data Error and Substitution Subroutine	A-14
Indeterminate and Misplaced Data Skip Subroutine	A-14
Net Data Processing Subroutine	A-15
Gen Data Processing Subroutine	A-16
Genf Data Processing Subroutine	A-17

Table A1

# FLOWCHART SYMBOL CONVENTION



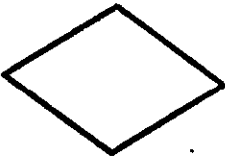
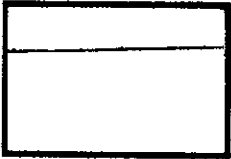

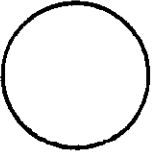


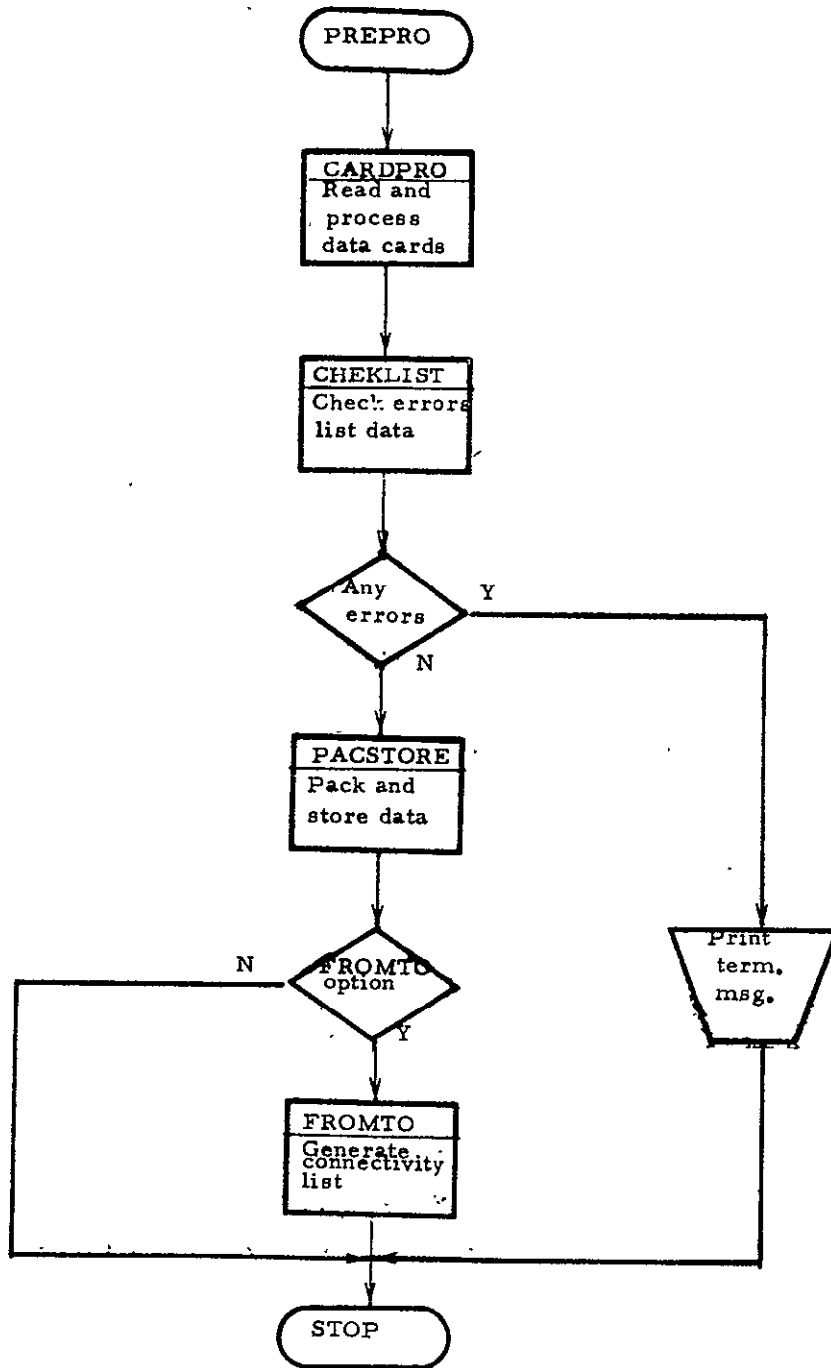
	Subroutine Terminal Points
	Process
	Decision
	Subroutine Call
	I/O Operation
	Magnetic Tape
	On Page Connector
	On Line Storage

Table A2

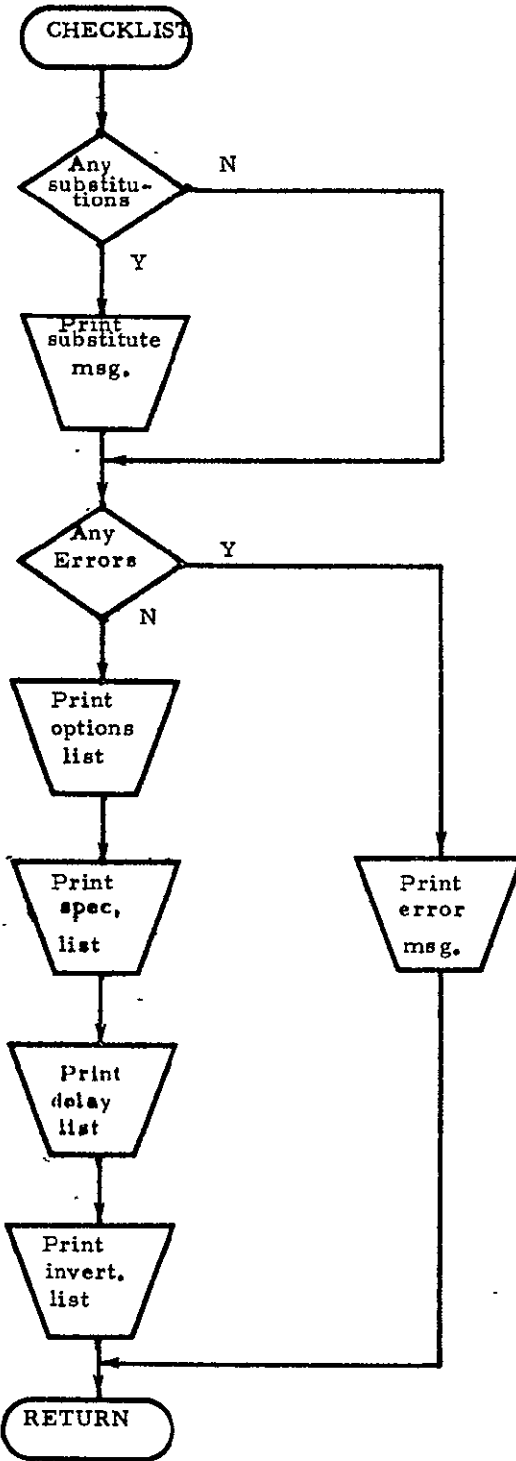
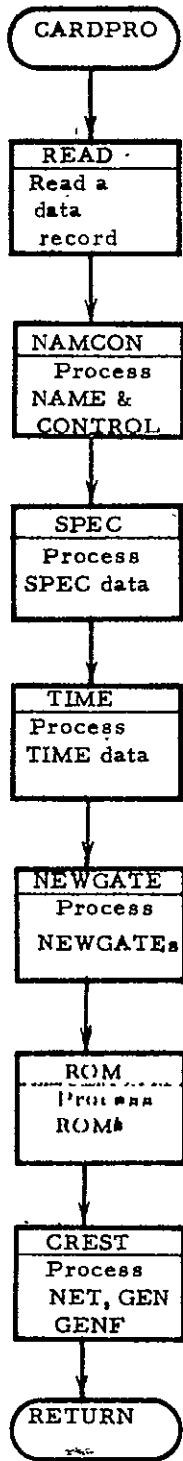
LOGIC SIMULATION PREPROCESSOR



ORIGINAL PAGE IS  
OF POOR QUALITY

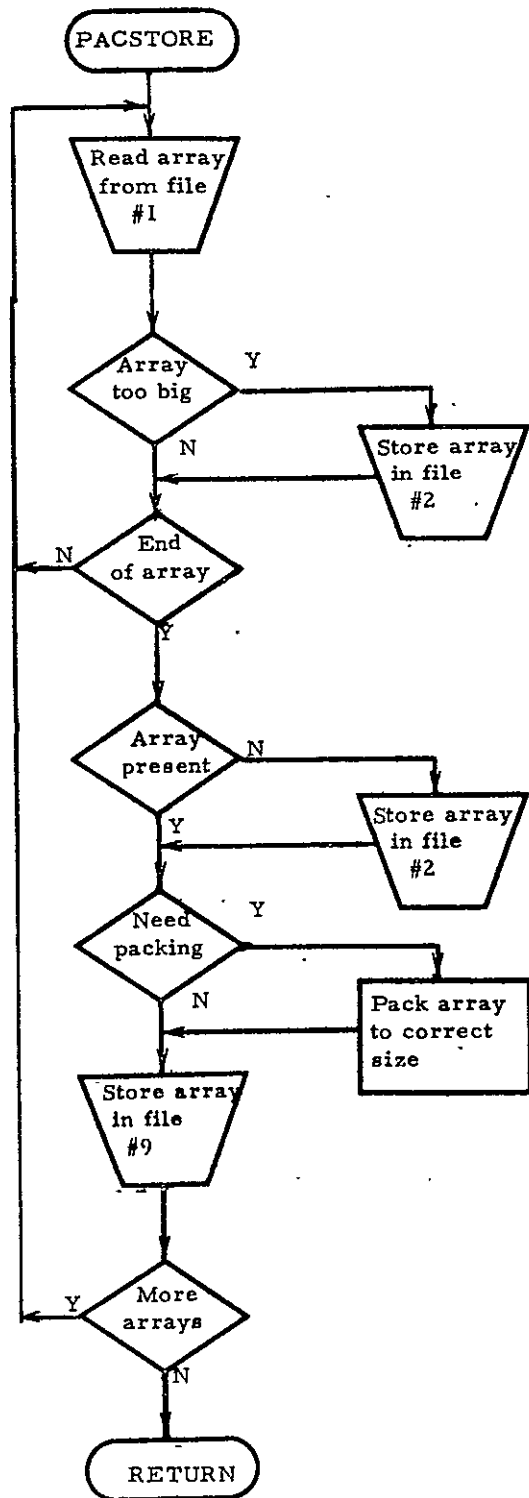
CARD PROCESSING SUBROUTINE

ERROR CHECK AND DATA LIST SUBROUTINE

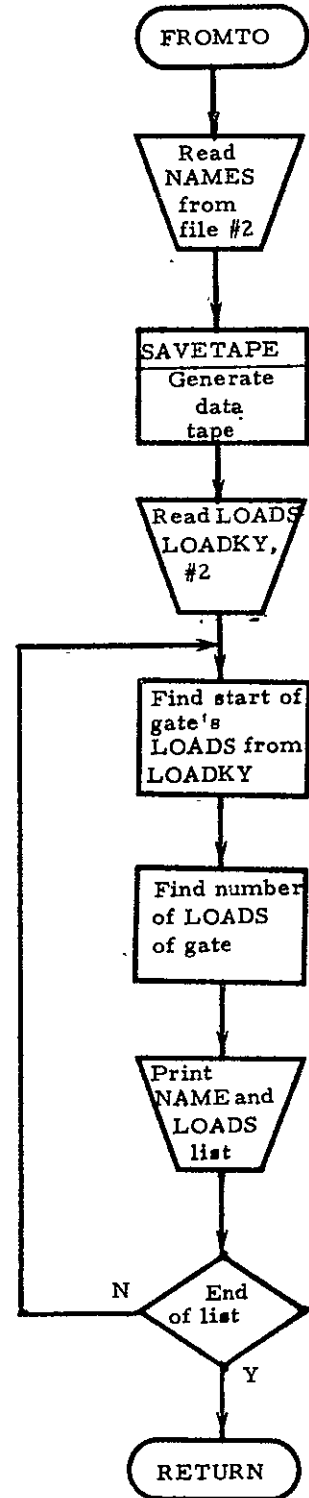


ORIGINAL PAGE IS  
OF POOR QUALITY

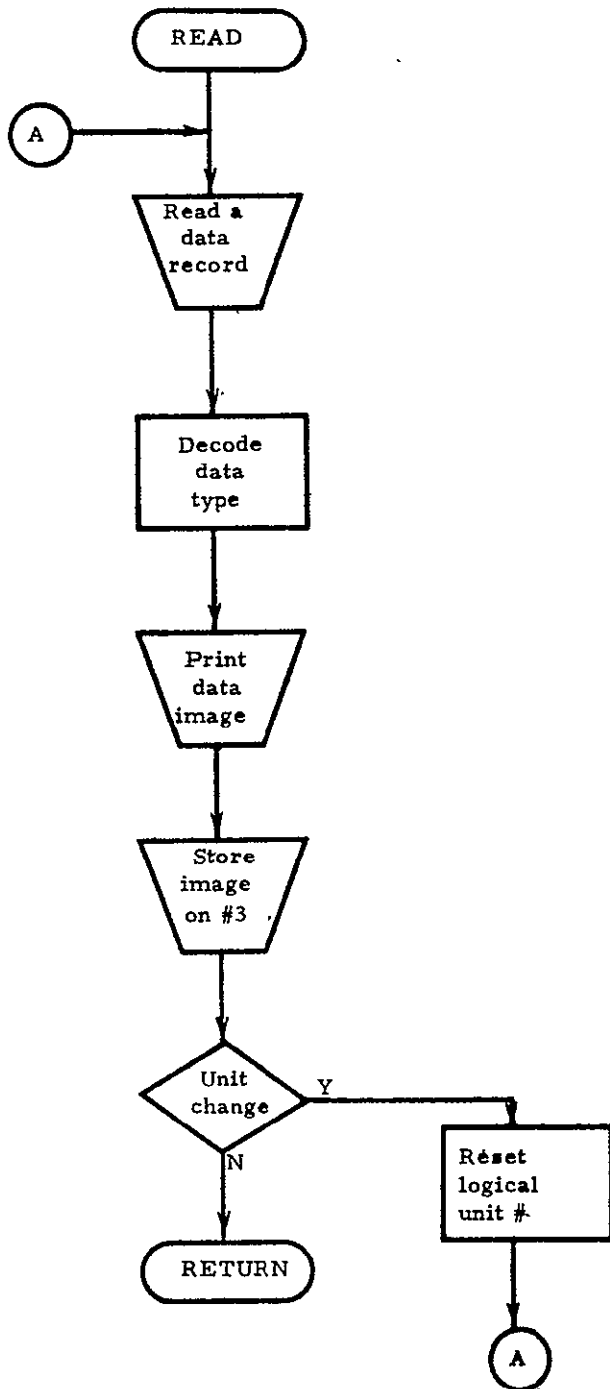
ARRAYS PACK AND STORE SUBROUTINE



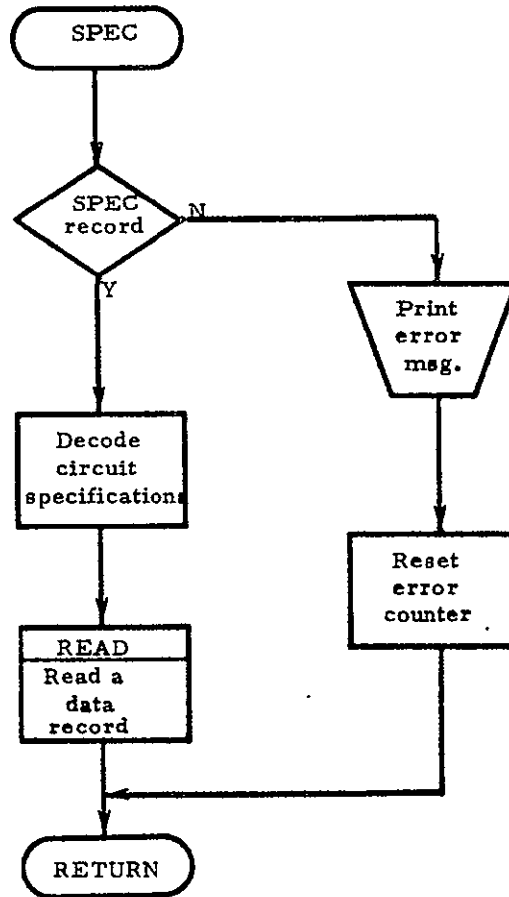
CONNECTIVITY LIST GENERATION SUBROUTINE



DATA RECORD READING AND IDENTIFICATION SUBROUTINE

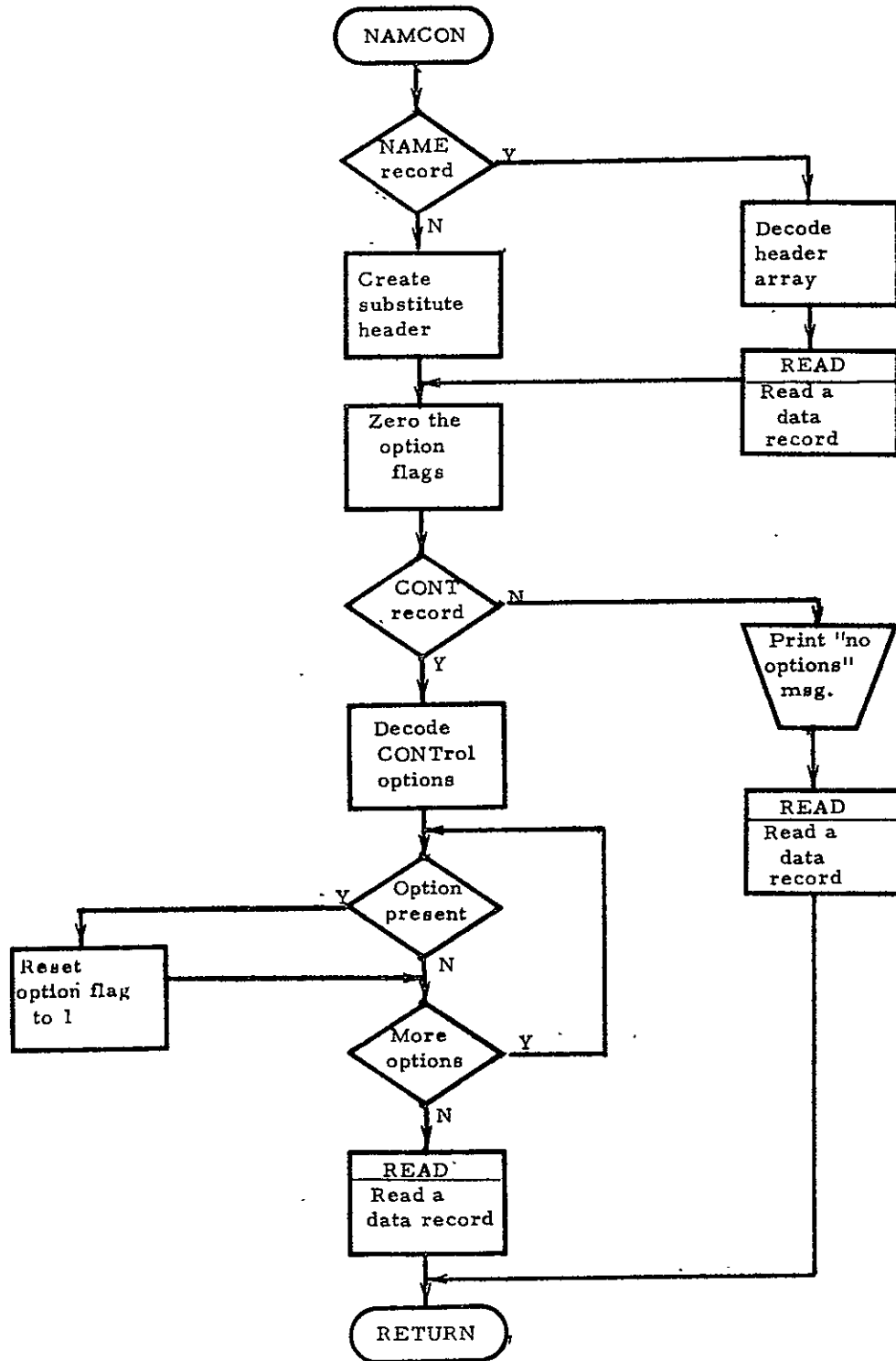


CIRCUIT SPECIFICATIONS SUBROUTINE

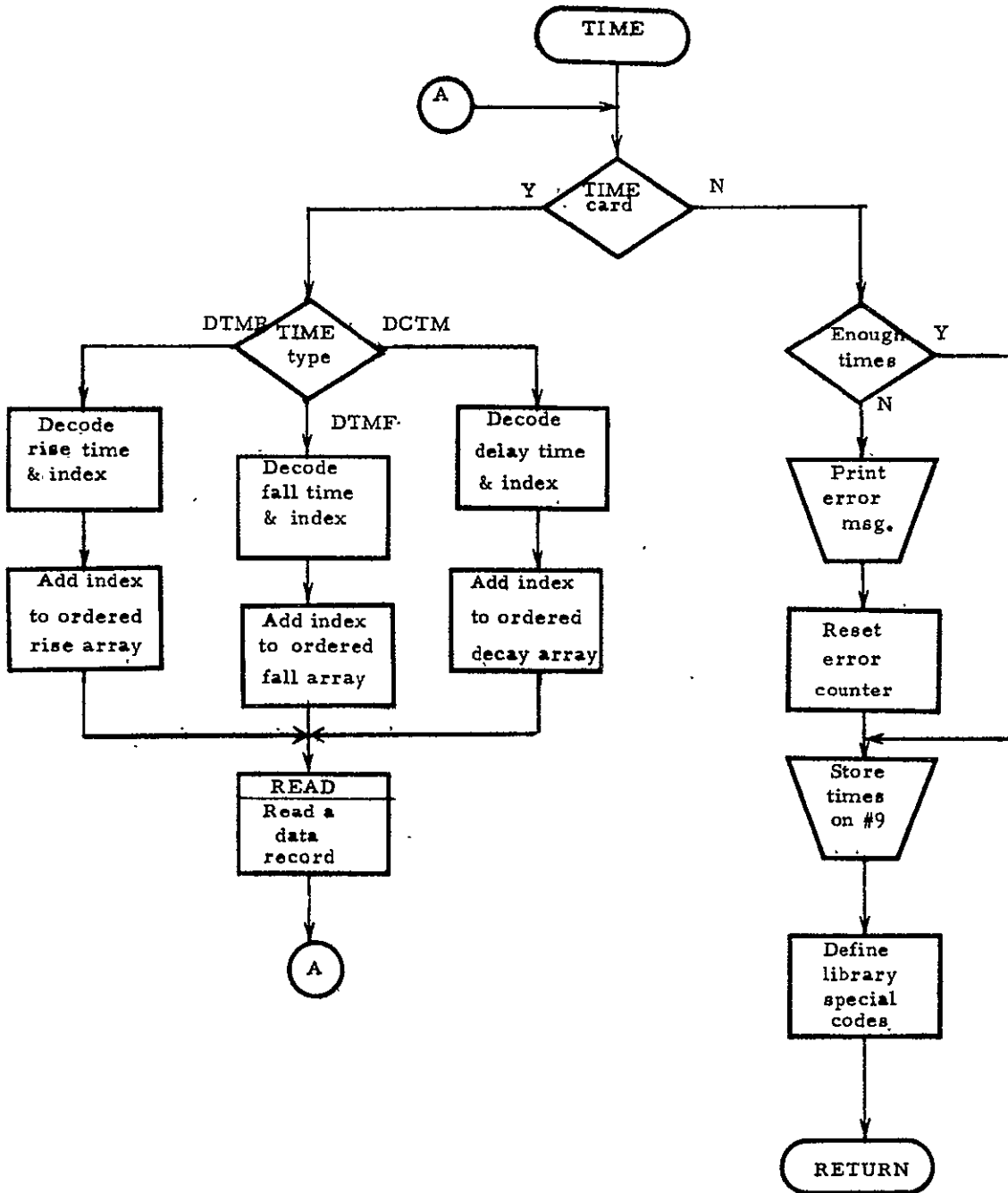




NAME AND CONTROL OPTION PROCESSING  
SUBROUTINE

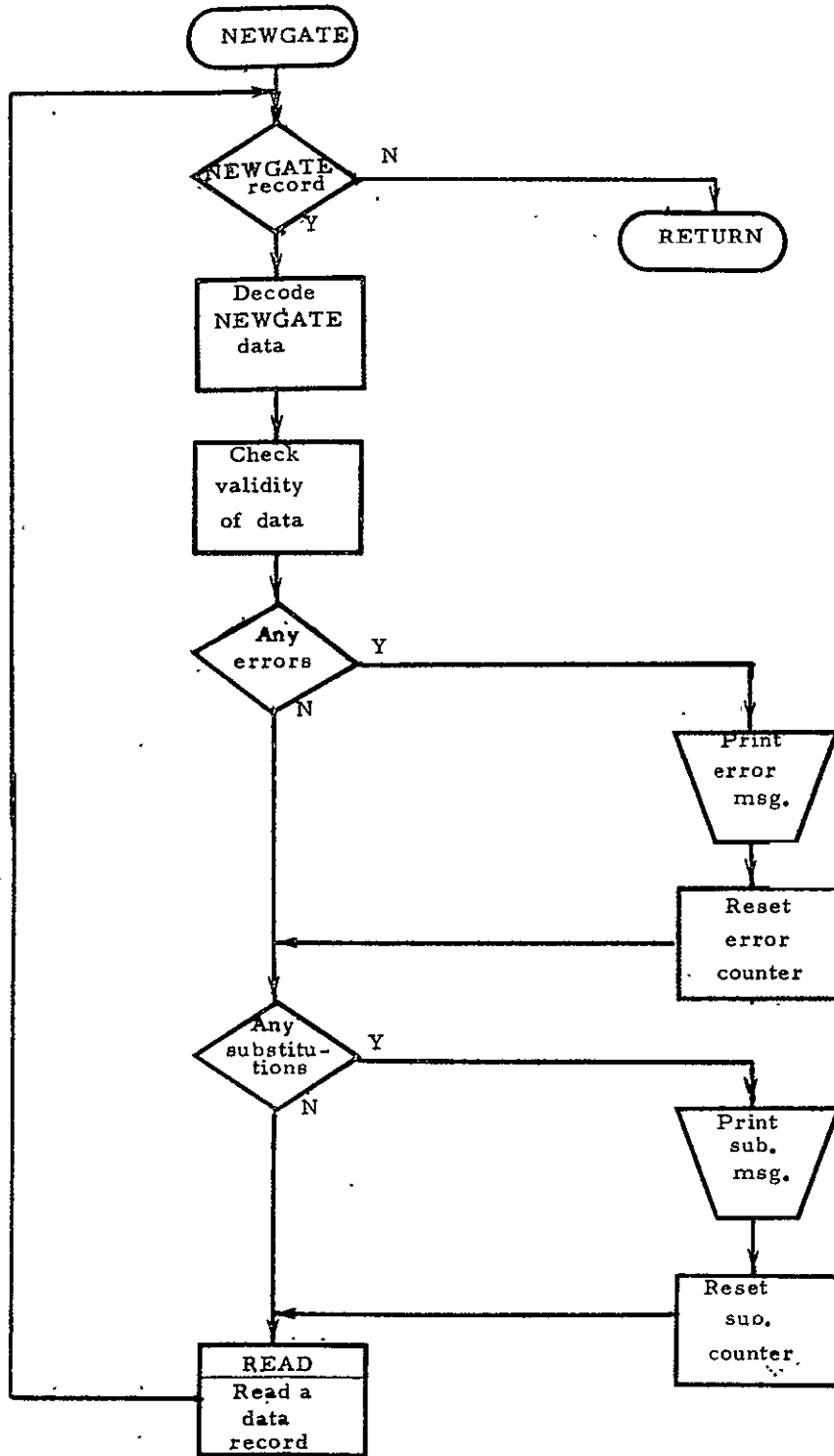


TIME RECORD PROCESSING SUBROUTINE

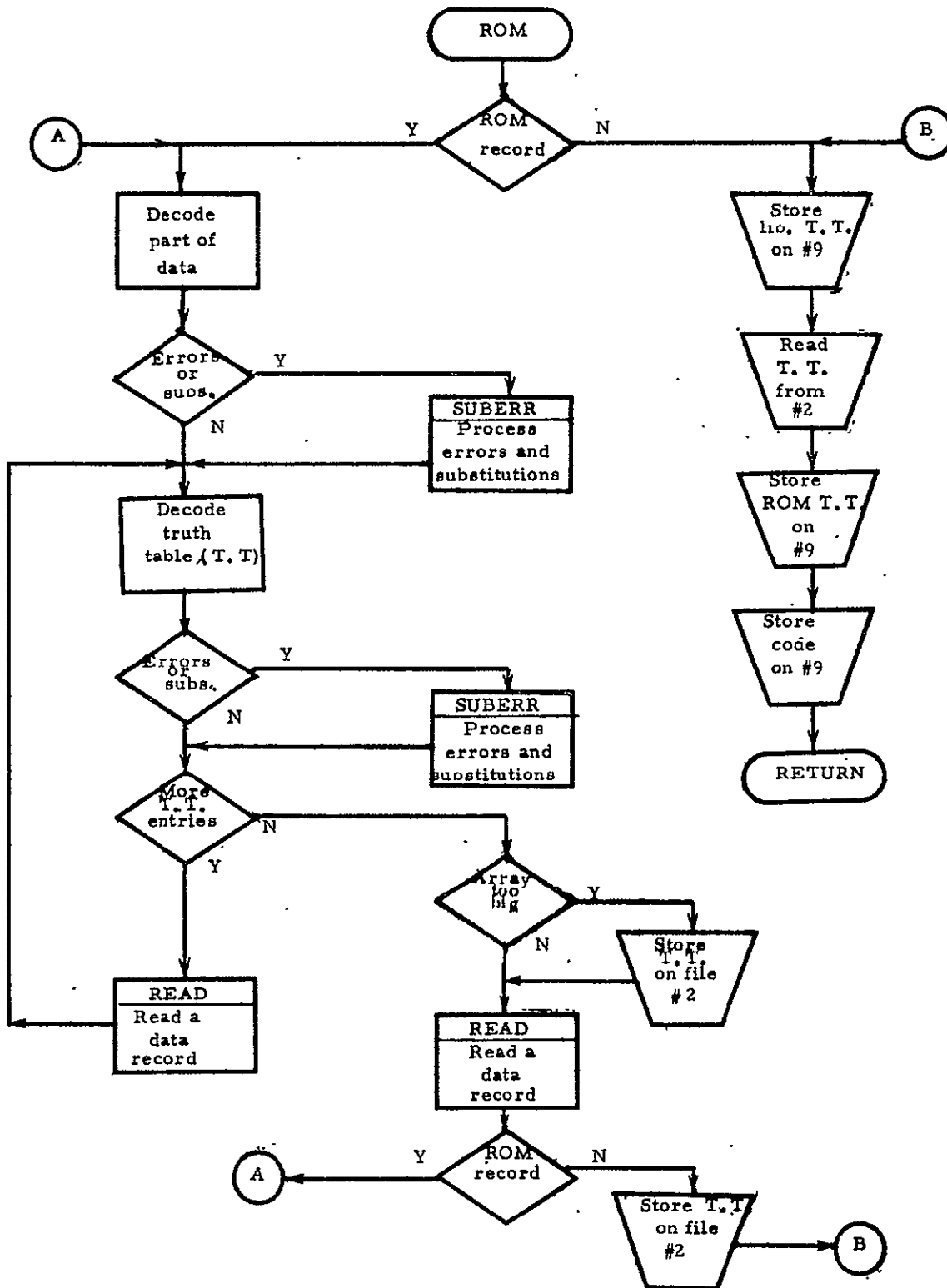


ORIGINAL PAGE IS  
OF POOR QUALITY

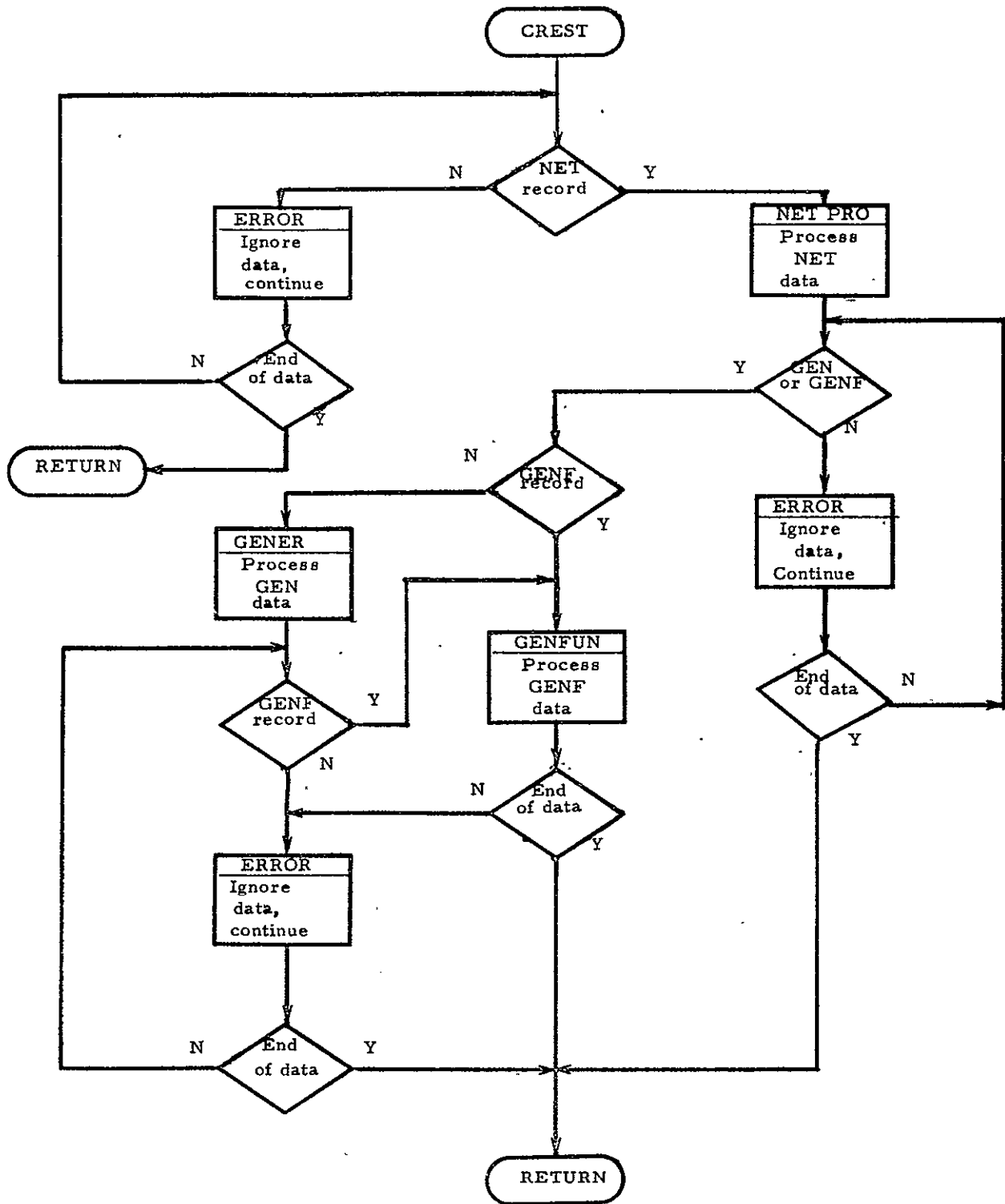
NEWGATE SPECIFICATION PROCESSING SUBROUTINE



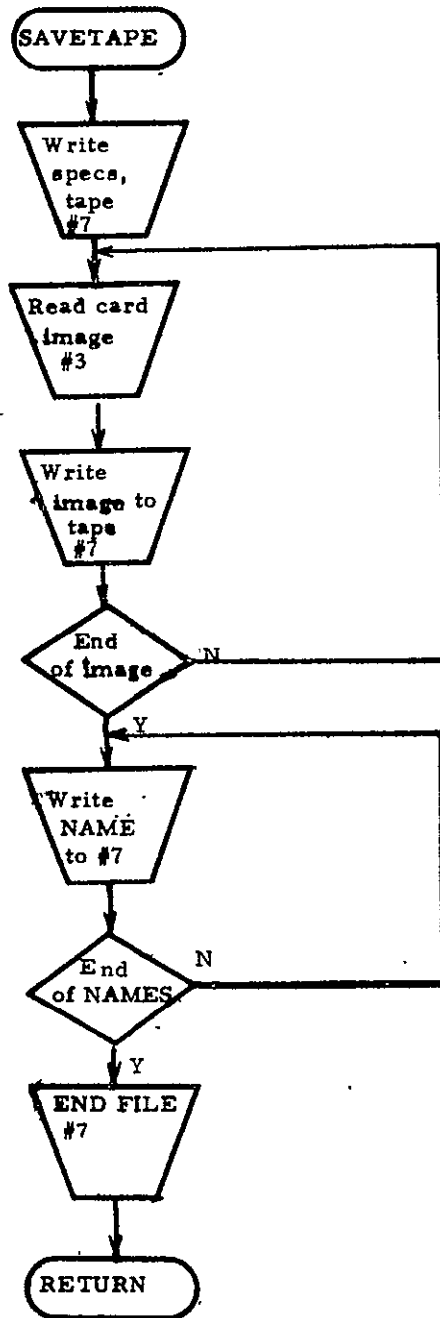
READ-ONLY MEMORY SPECIFICATION PROCESSING SUBROUTINE



ELEMENT DESCRIPTION RECORD PROCESSING SUBROUTINES

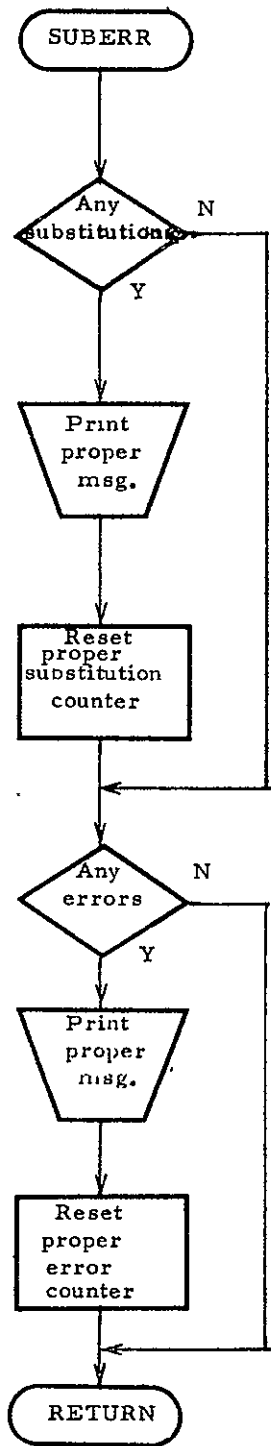


DATA TAPE GENERATION SUBROUTINE

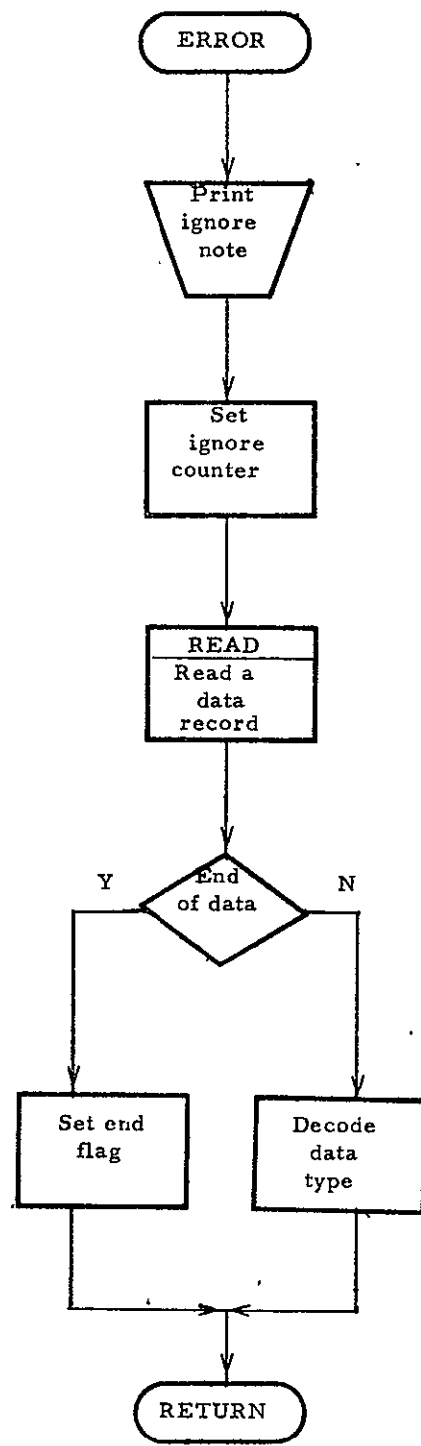


ORIGINAL PAGE IS  
OF POOR QUALITY

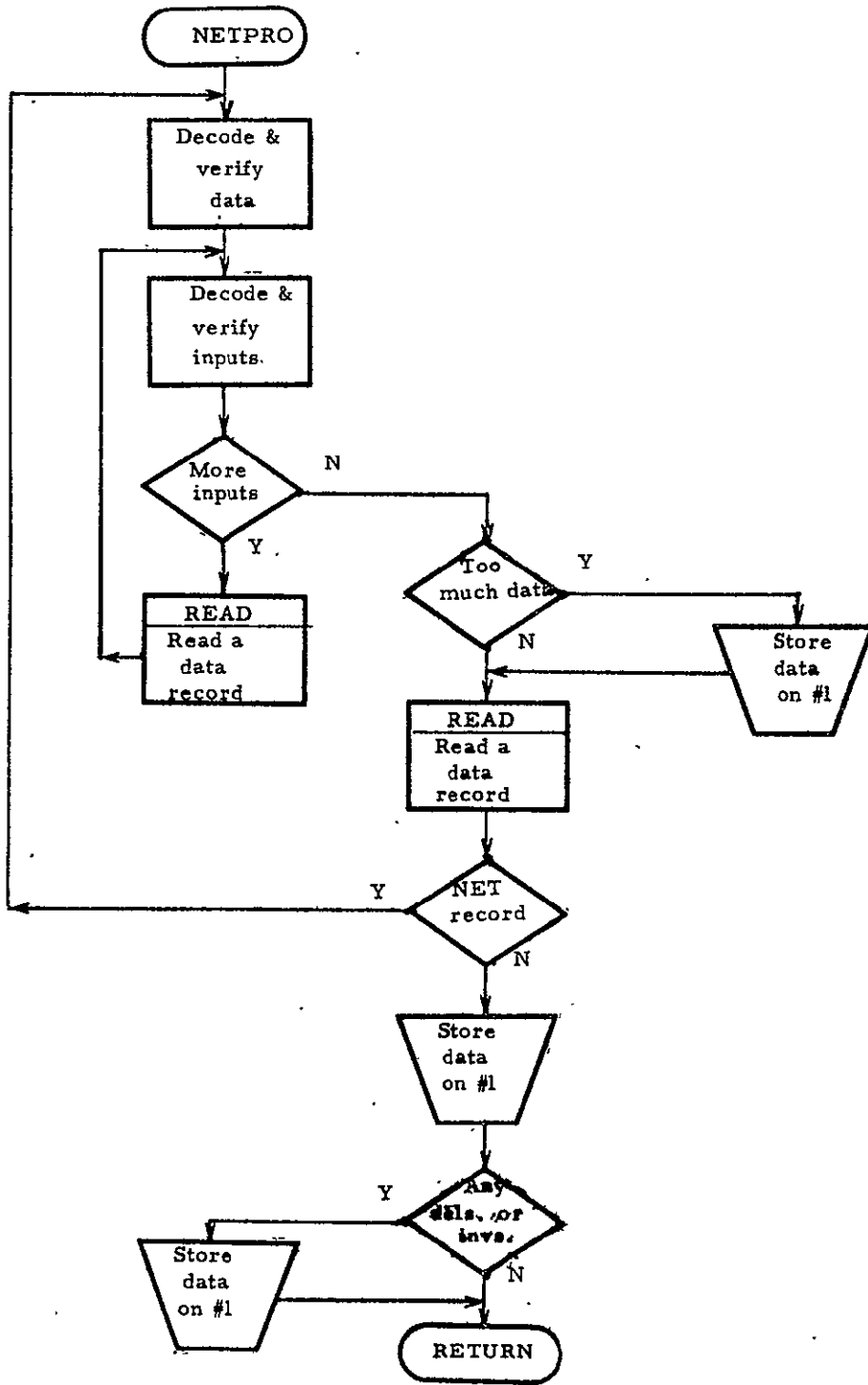
DATA ERROR AND SUBSTITUTION  
SUBROUTINE



INDETERMINATE AND MISPLACED  
DATA SKIP SUBROUTINE

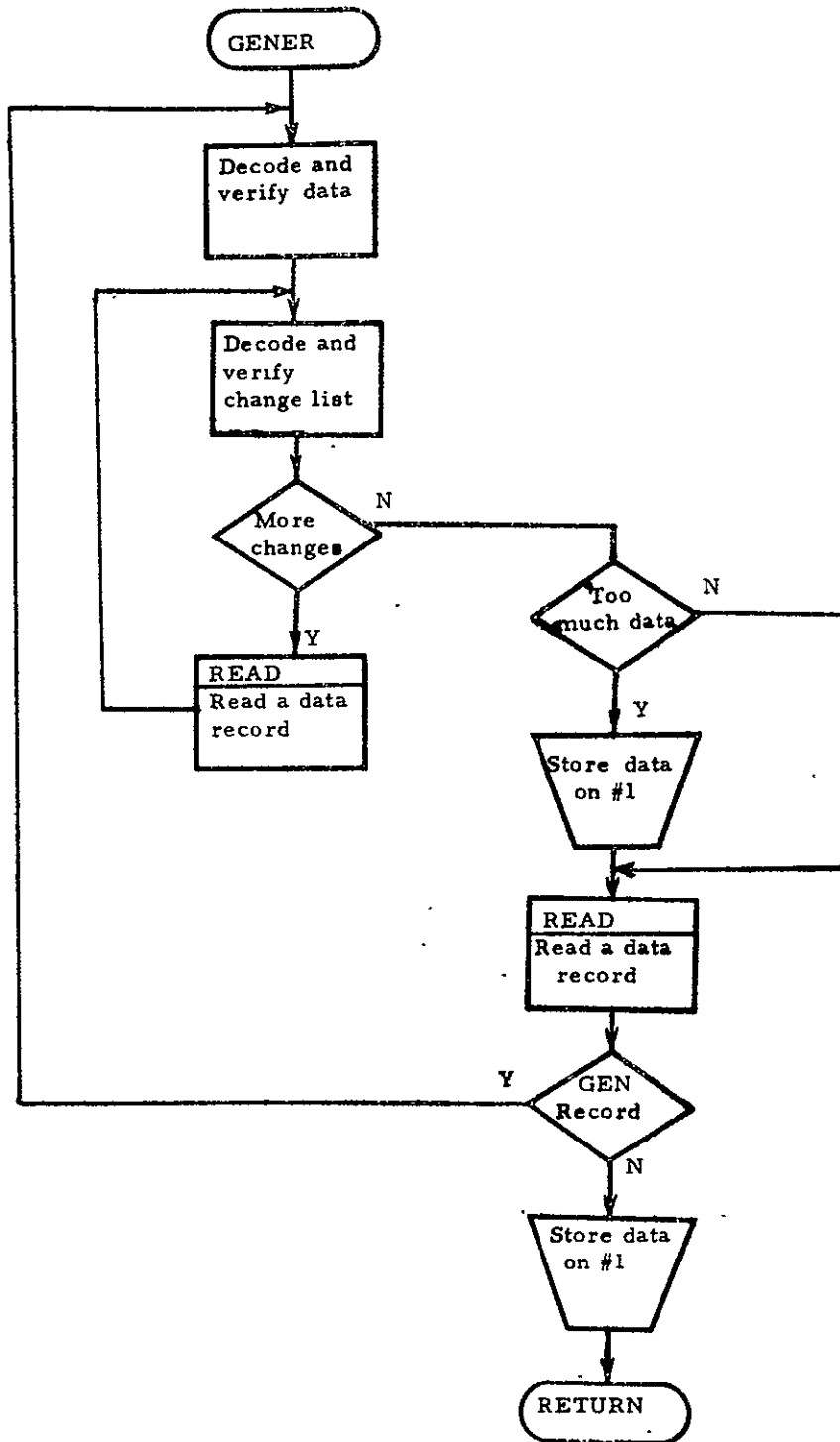


NET DATA PROCESSING SUBROUTINE

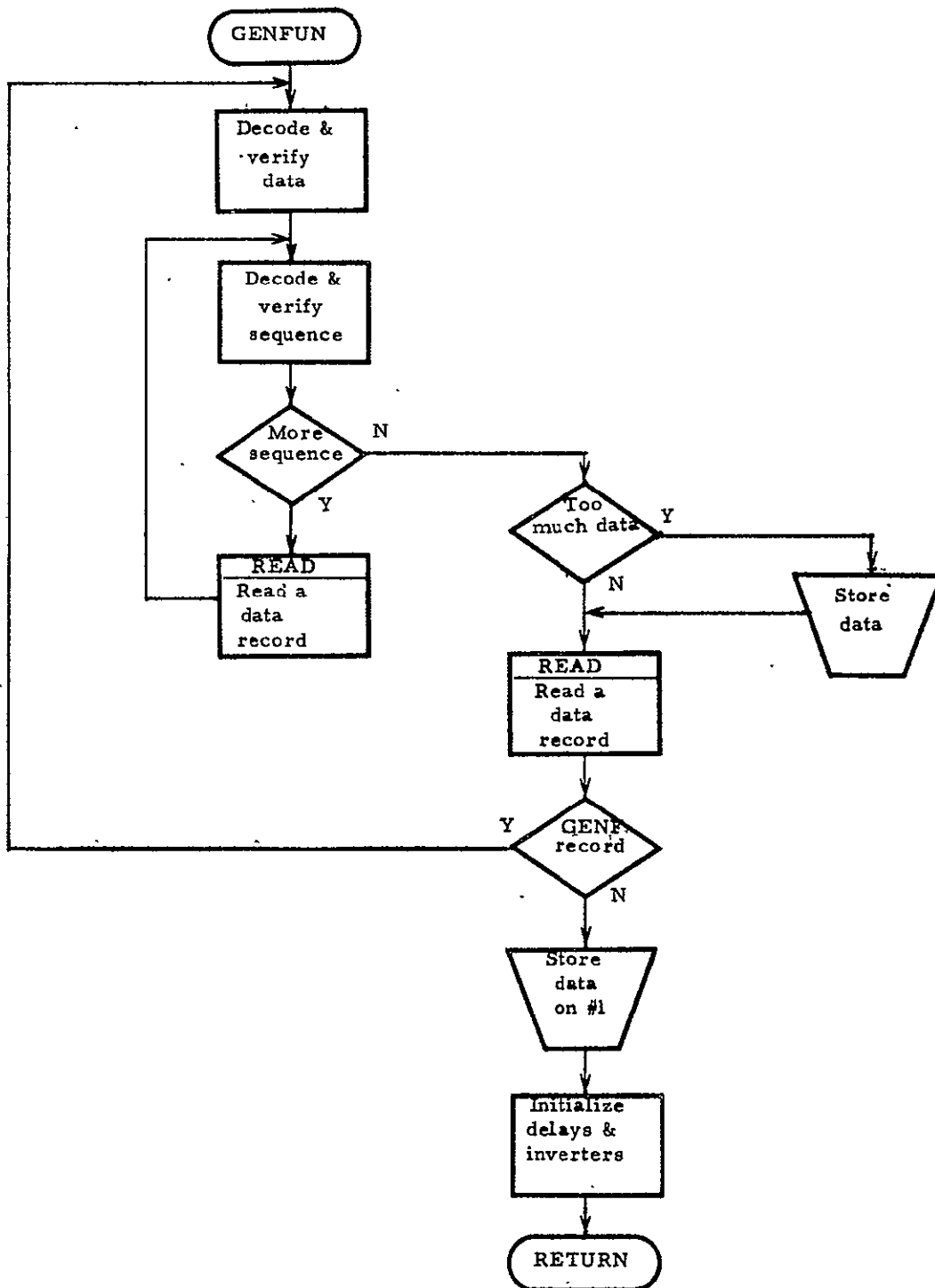




GEN DATA PROCESSING SUBROUTINE



GENF DATA PROCESSING SUBROUTINE



ORIGINAL PAGE IS  
OF POOR QUALITY

## APPENDIX B

### LOGSIM SIMULATOR PROGRAM FLOWCHARTS

This Appendix presents the detail flowcharts of the LOGSIM Simulator Program. As with the Preprocessor flowcharts in Appendix A, the "Picture on a Page" technique has been utilized in these flowcharts and is described in Appendix A, page A-1.

An index to the LOGSIM Simulator Program flowcharts is contained in Table B1 and the same flowchart symbol convention described in Table A2 is adhered to in the Simulator flowcharts.

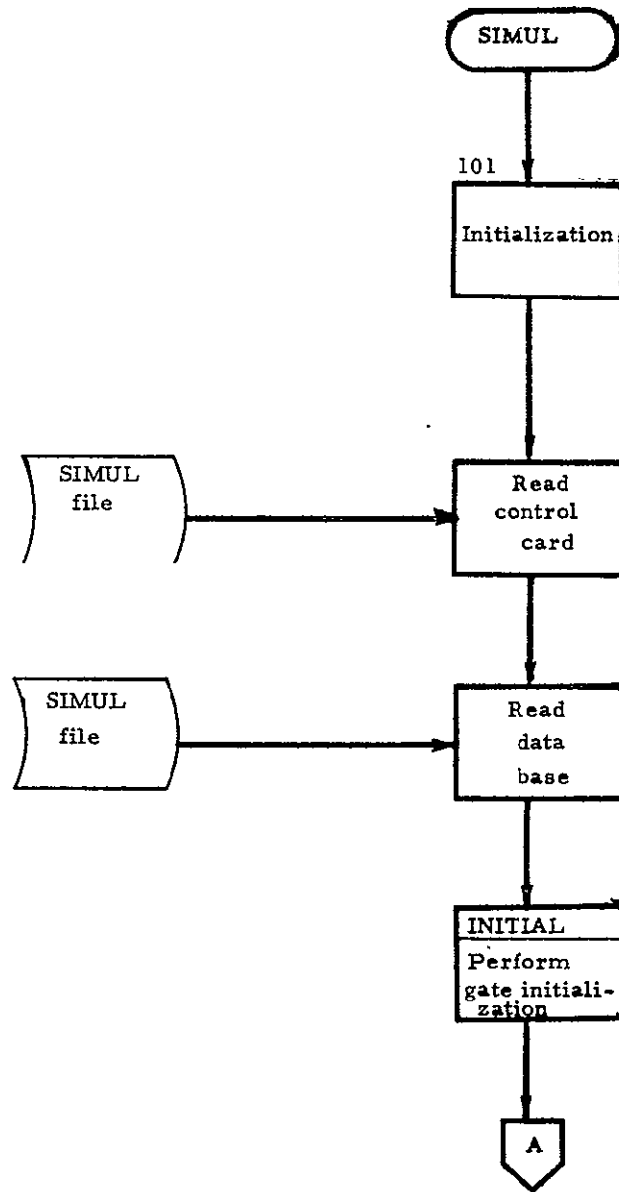
# LOGSIM SIMULATOR PROGRAM

## · FLOWCHART INDEX

<u>Routines</u>	<u>Page</u>
LOGSIM Simulator	B - 3
Gate Initialization Subroutine	B - 4
Event Simulation Subroutine	B - 5
Termination Subroutine	B - 6
Event Output Subroutine	B - 7
Logic State Determination Subroutine	B - 8
Generator Events Restoration Subroutine	B - 9
Read-Only-Memory State Determination Subroutine	B - 10
Delay Gate and Inverter State Determination Subroutine	B - 11
Logic Gate State Determination Subroutine	B - 12
Event Storage Subroutine	B - 13
Initial Conditions Storage Subroutine	B - 14
Holding Mode Event Processing Subroutine	B - 15
FEC Clearing Subroutine	B - 16

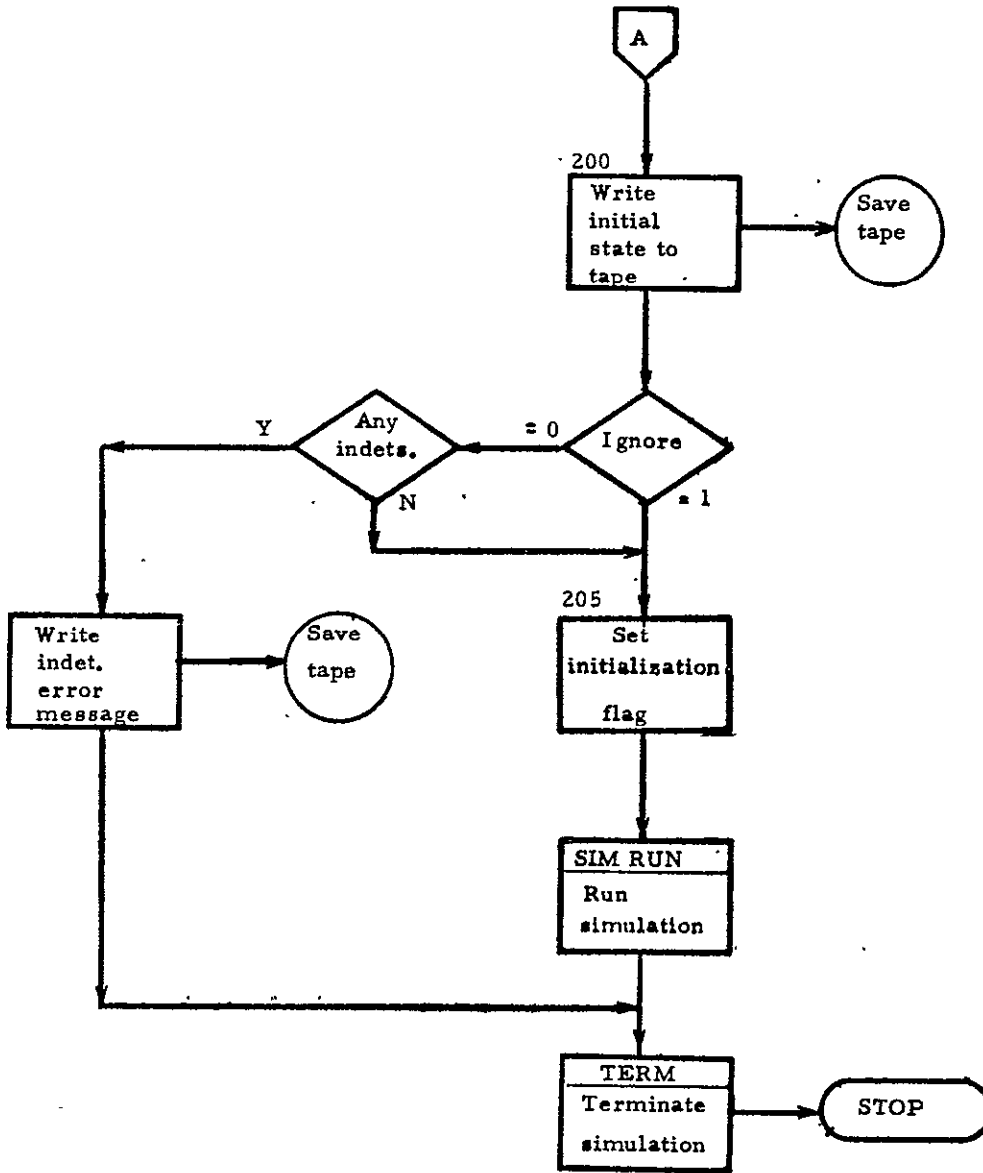
Table B1

LOGSIM SIMULATOR

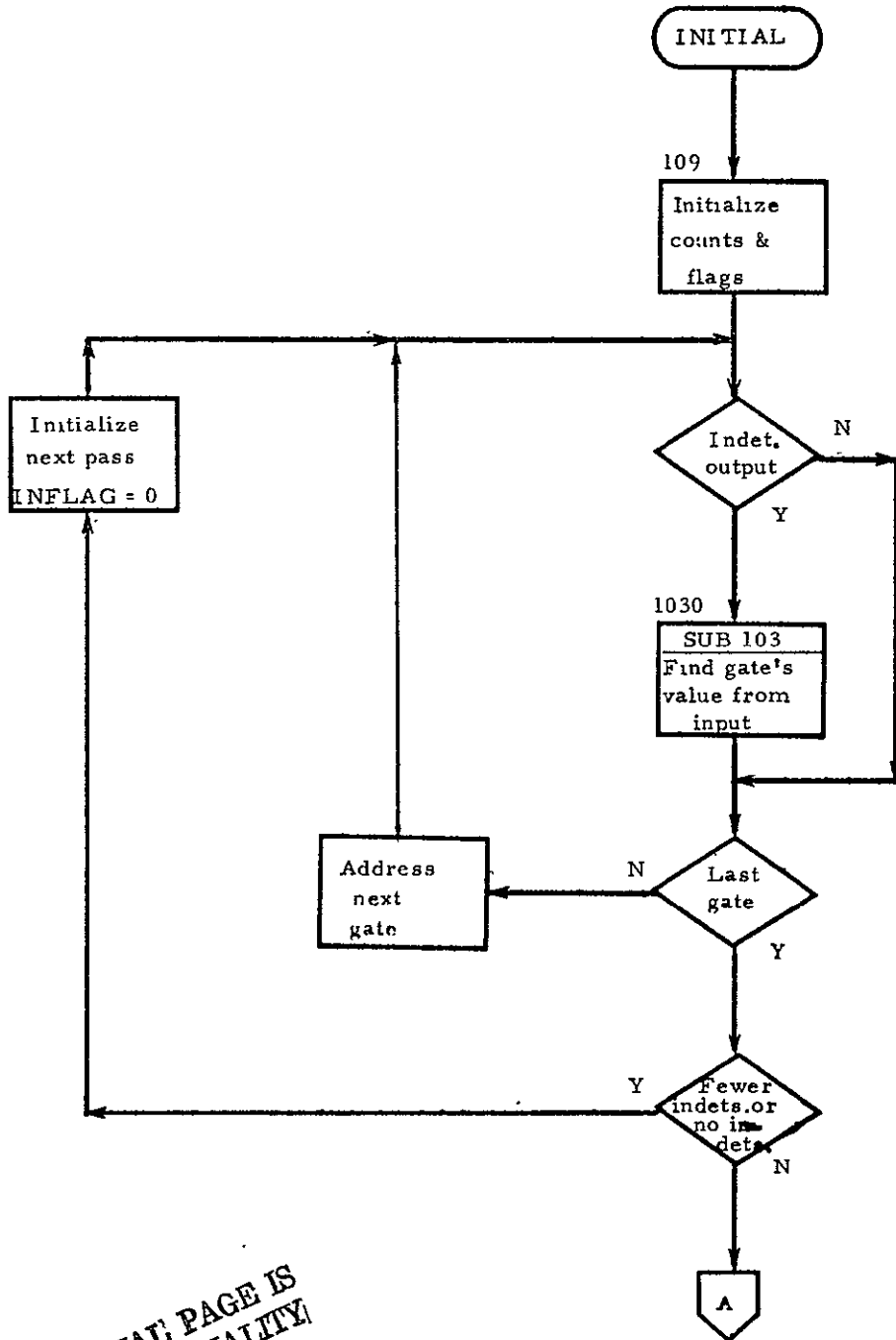


ORIGINAL PAGE IS  
OF POOR QUALITY

LOGSIM SIMULATOR

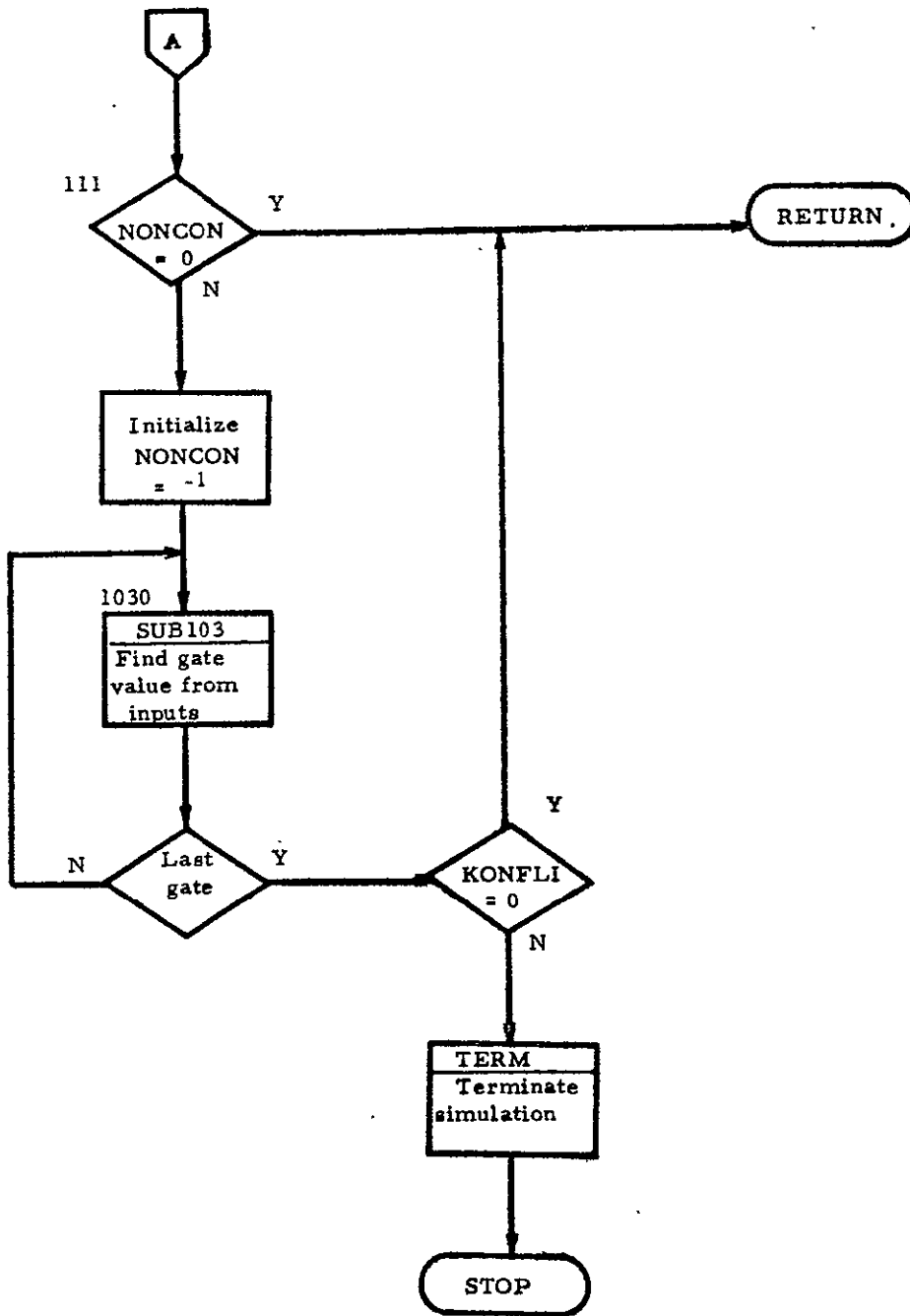


GATE INITIALIZATION SUBROUTINE



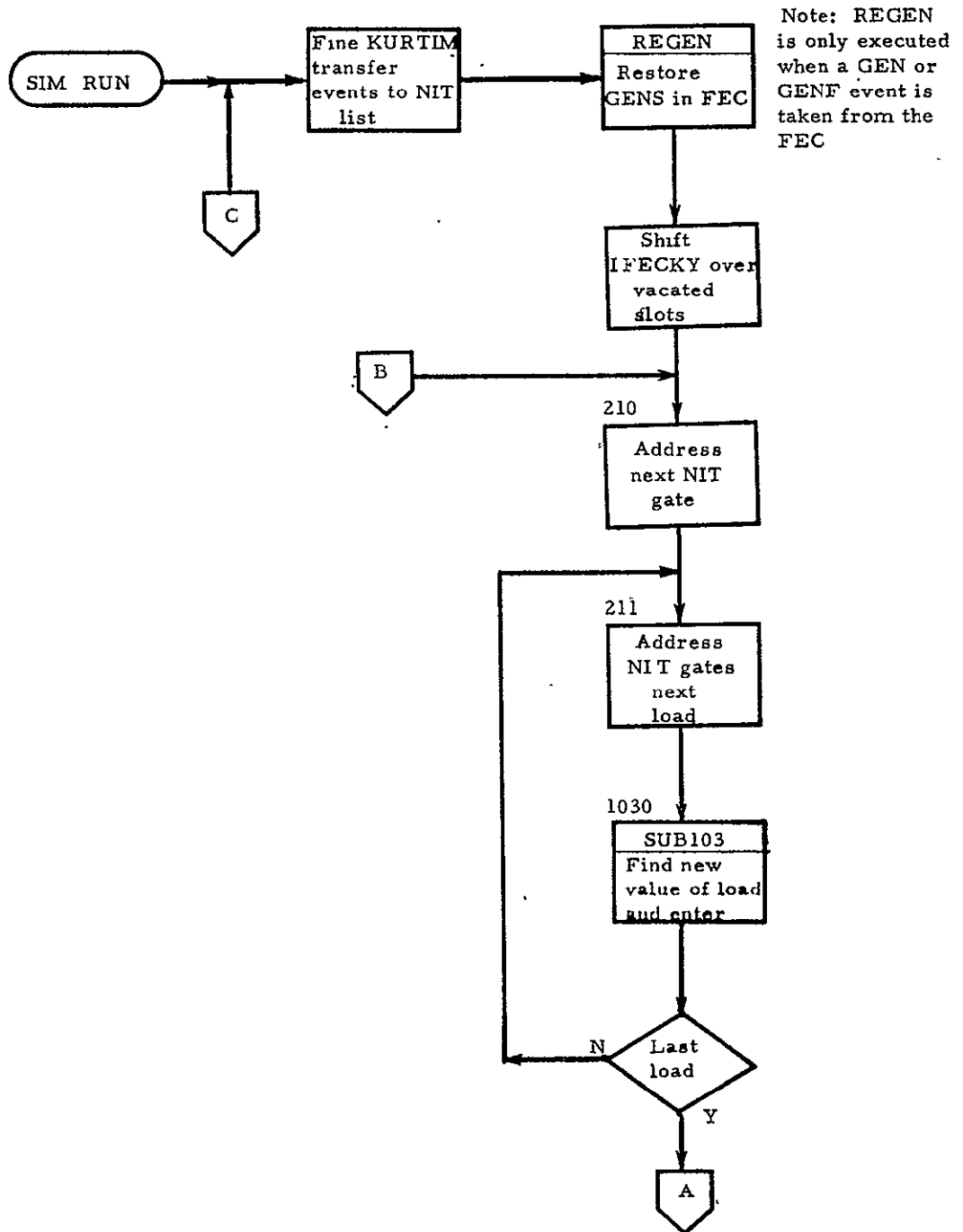
ORIGINAL PAGE IS  
OF POOR QUALITY

# GATE INITIALIZATION SUBROUTINE



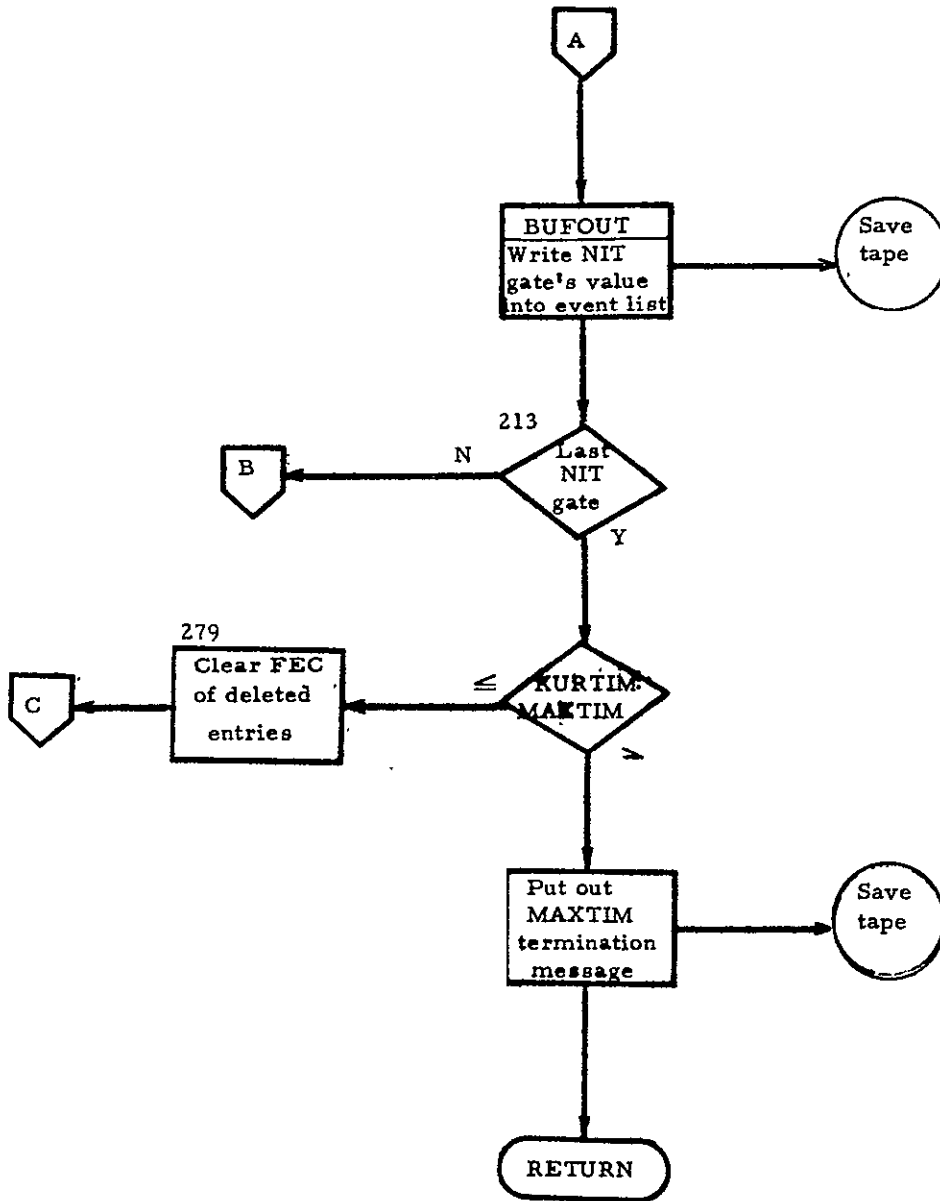


EVENT SIMULATION SUBROUTINE

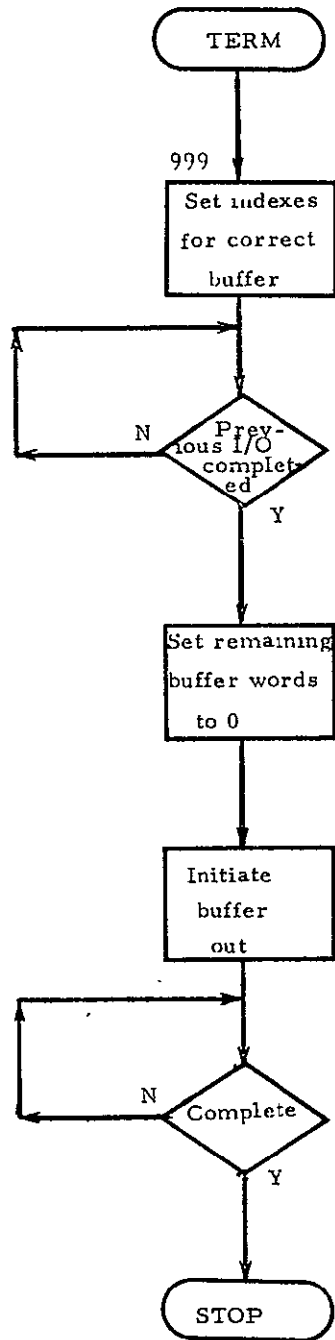


ORIGINAL PAGE IS  
OF POOR QUALITY

EVENT SIMULATION SUBROUTINE

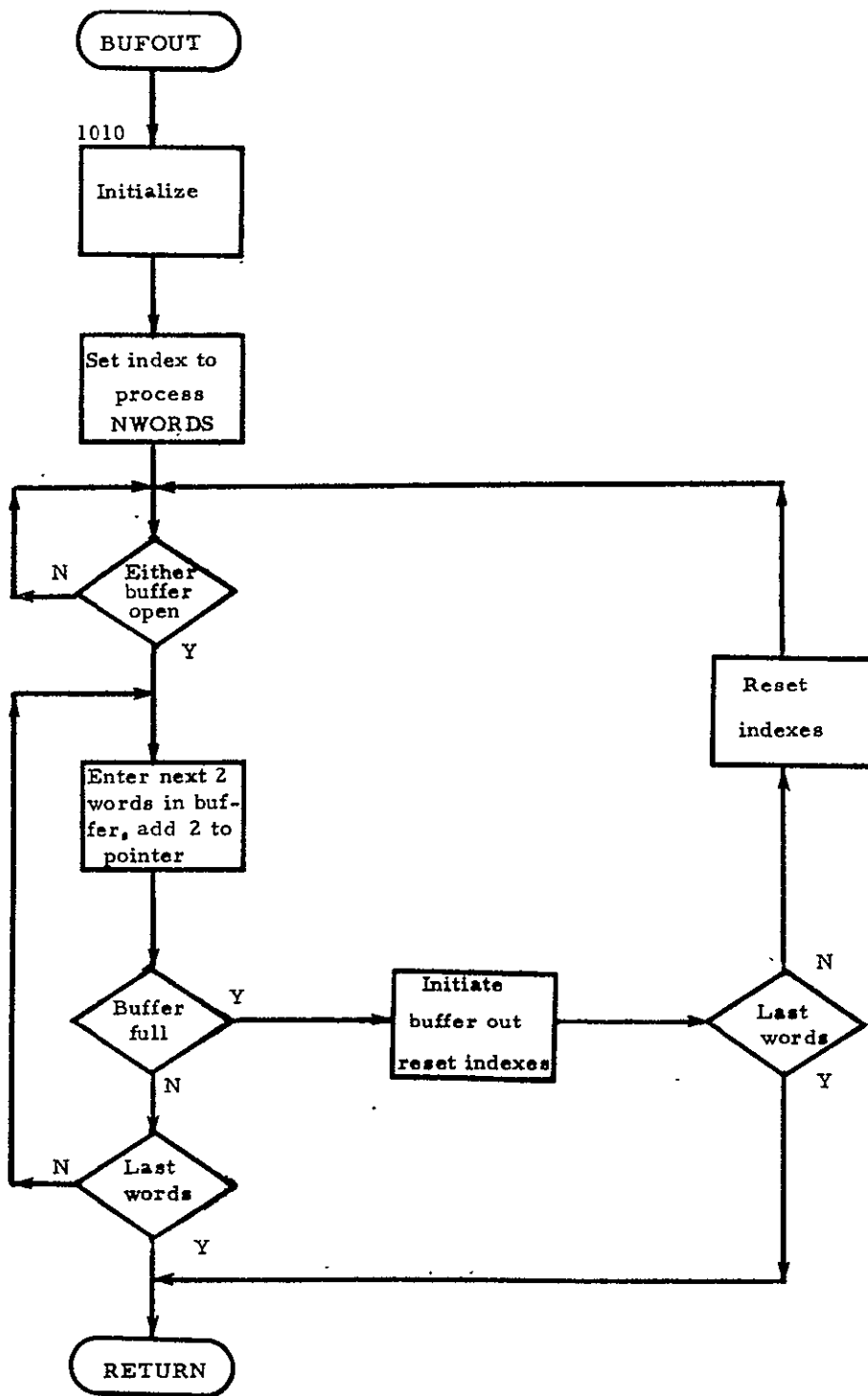


TERMINATION SUBROUTINE

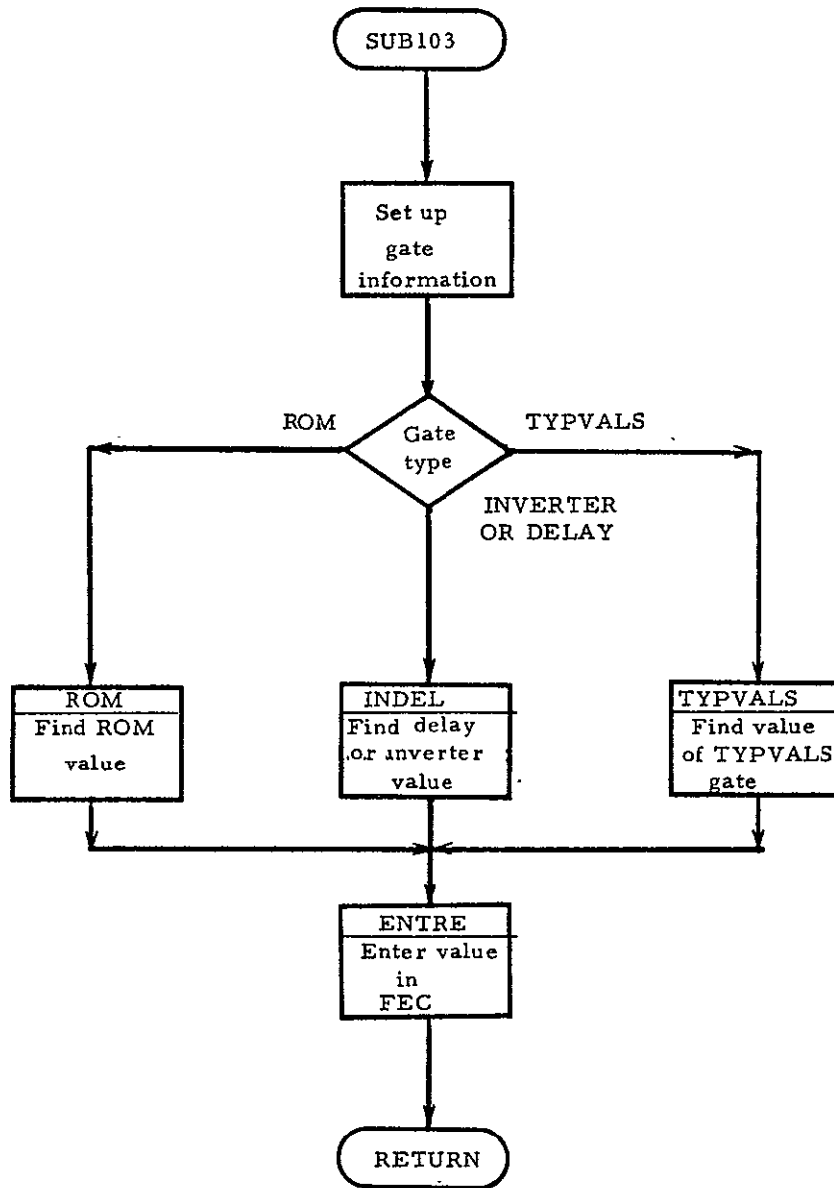


ORIGINAL PAGE IS  
OF POOR QUALITY

EVENT OUTPUT SUBROUTINE

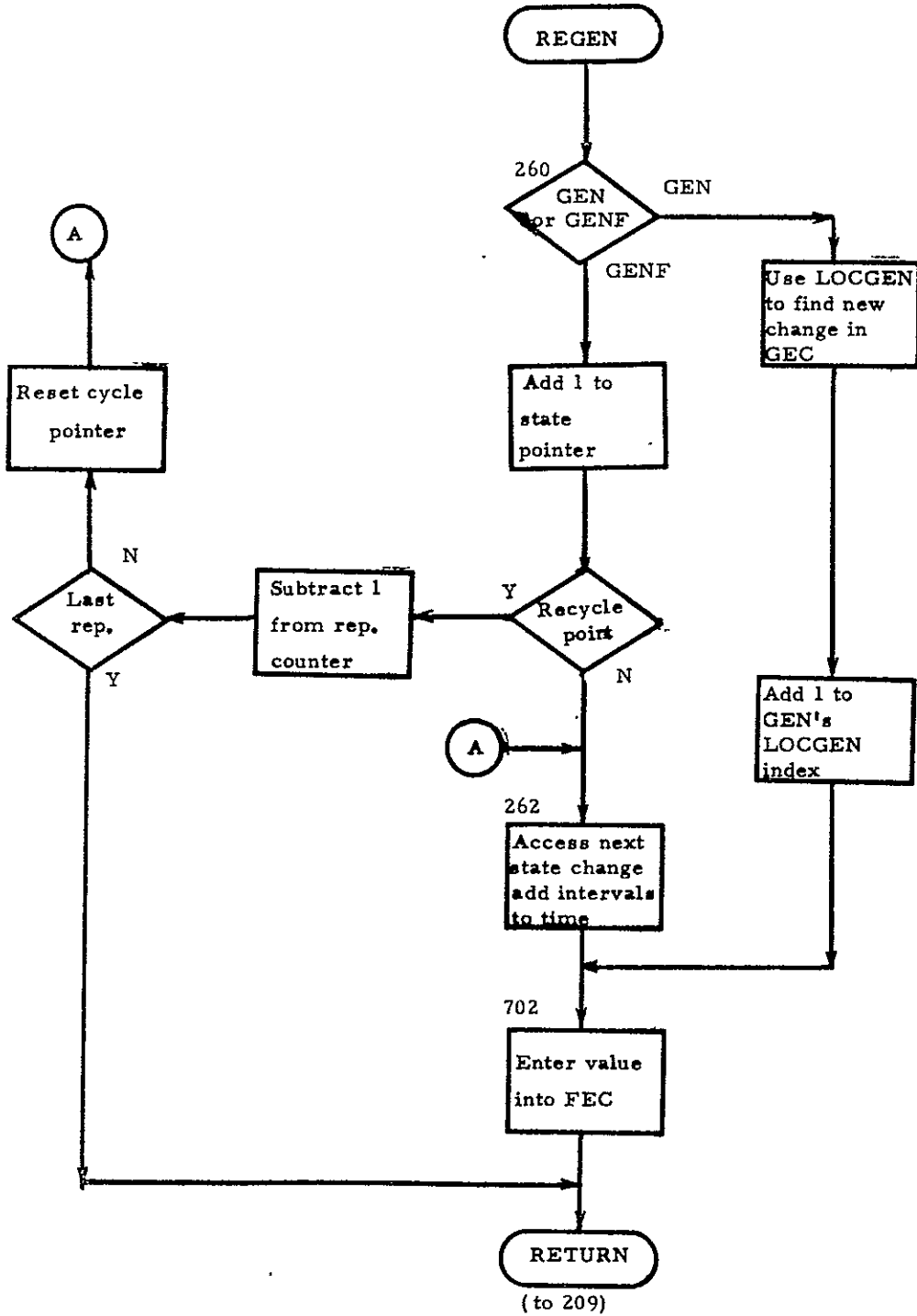


LOGIC STATE DETERMINATION SUBROUTINE

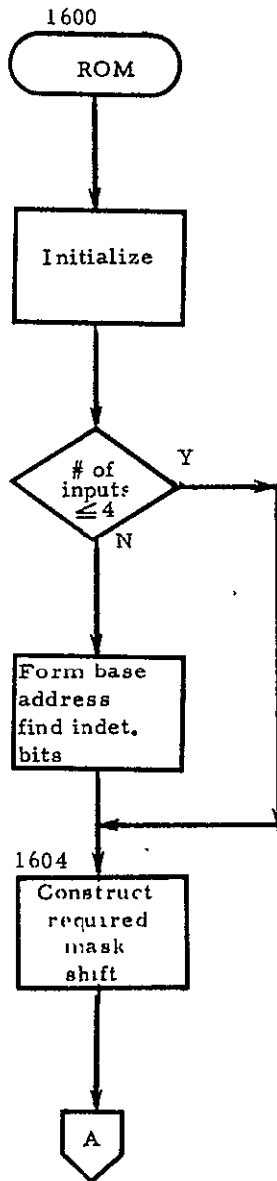


ORIGINAL PAGE IS  
OF POOR QUALITY

GENERATOR EVENTS RESTORATION SUBROUTINE

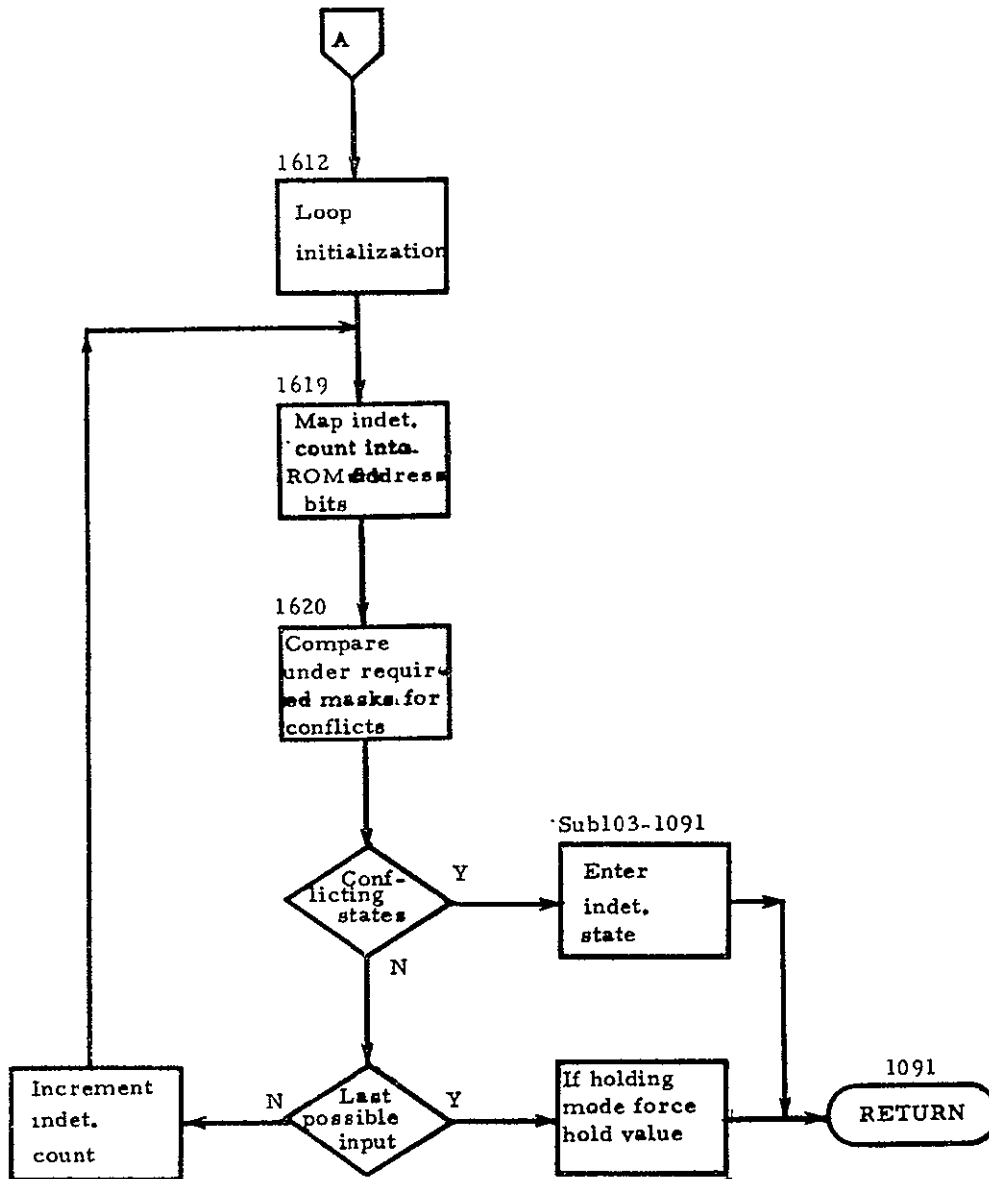


READ-ONLY-MEMORY STATE DETERMINATION SUBROUTINE



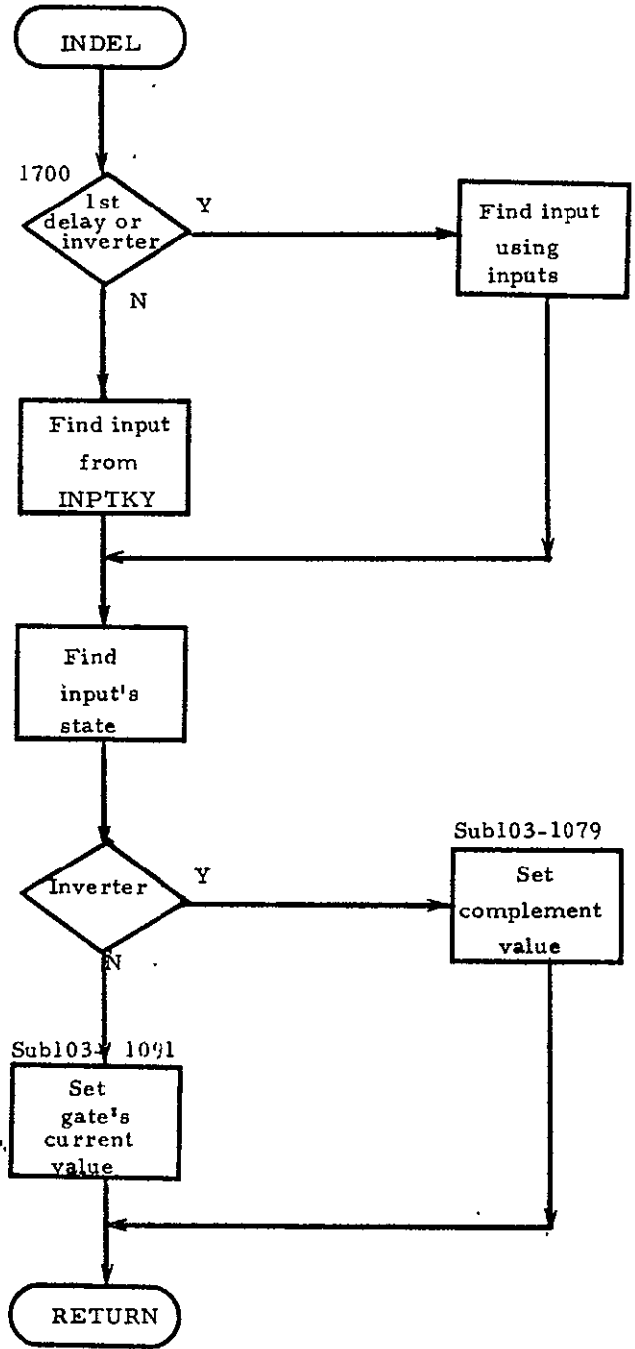
ORIGINAL PAGE IS  
OF POOR QUALITY

READ-ONLY-MEMORY STATE DETERMINATION SUBROUTINE

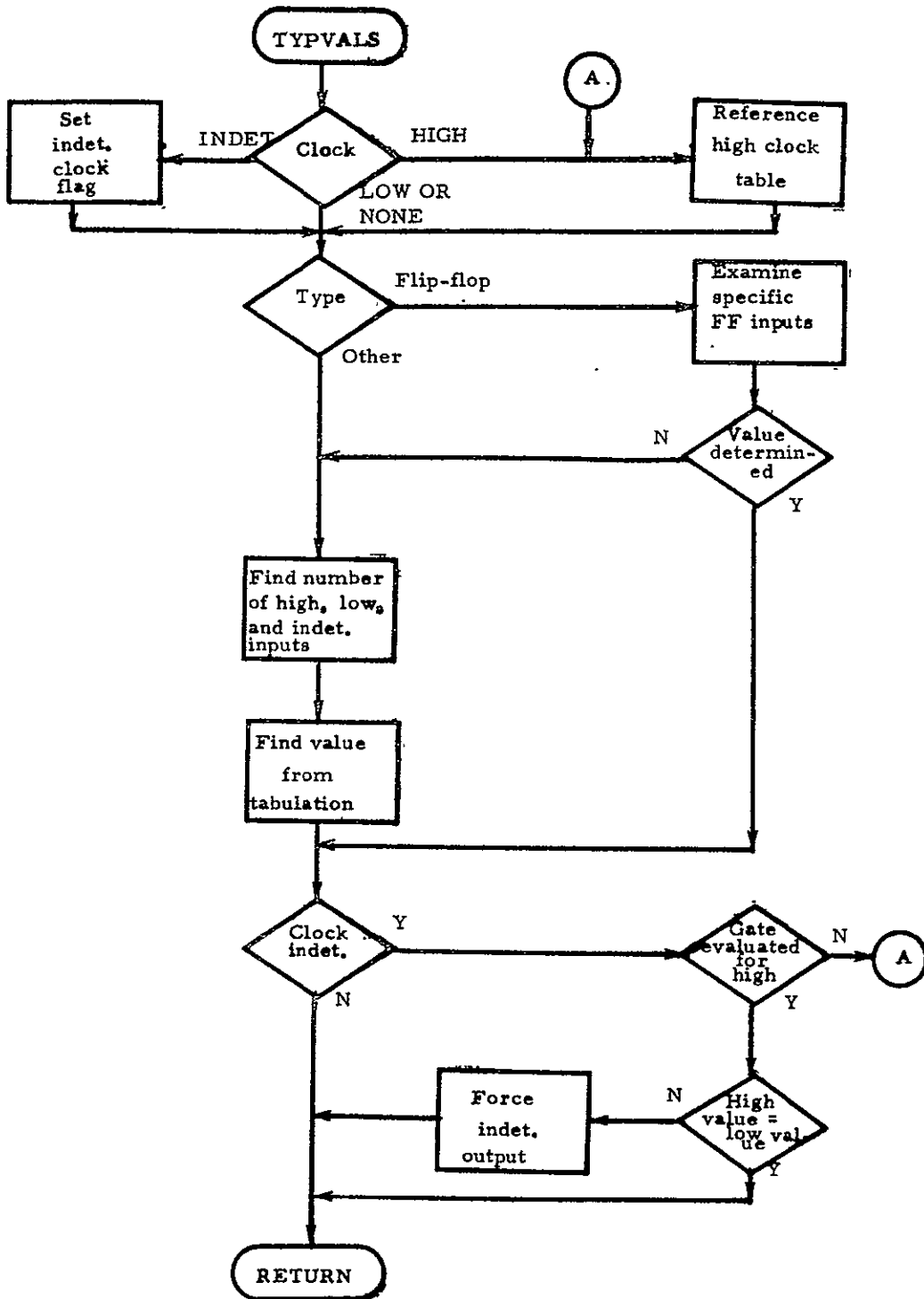




DELAY GATE AND INVERTER STATE DETERMINATION SUBROUTINE

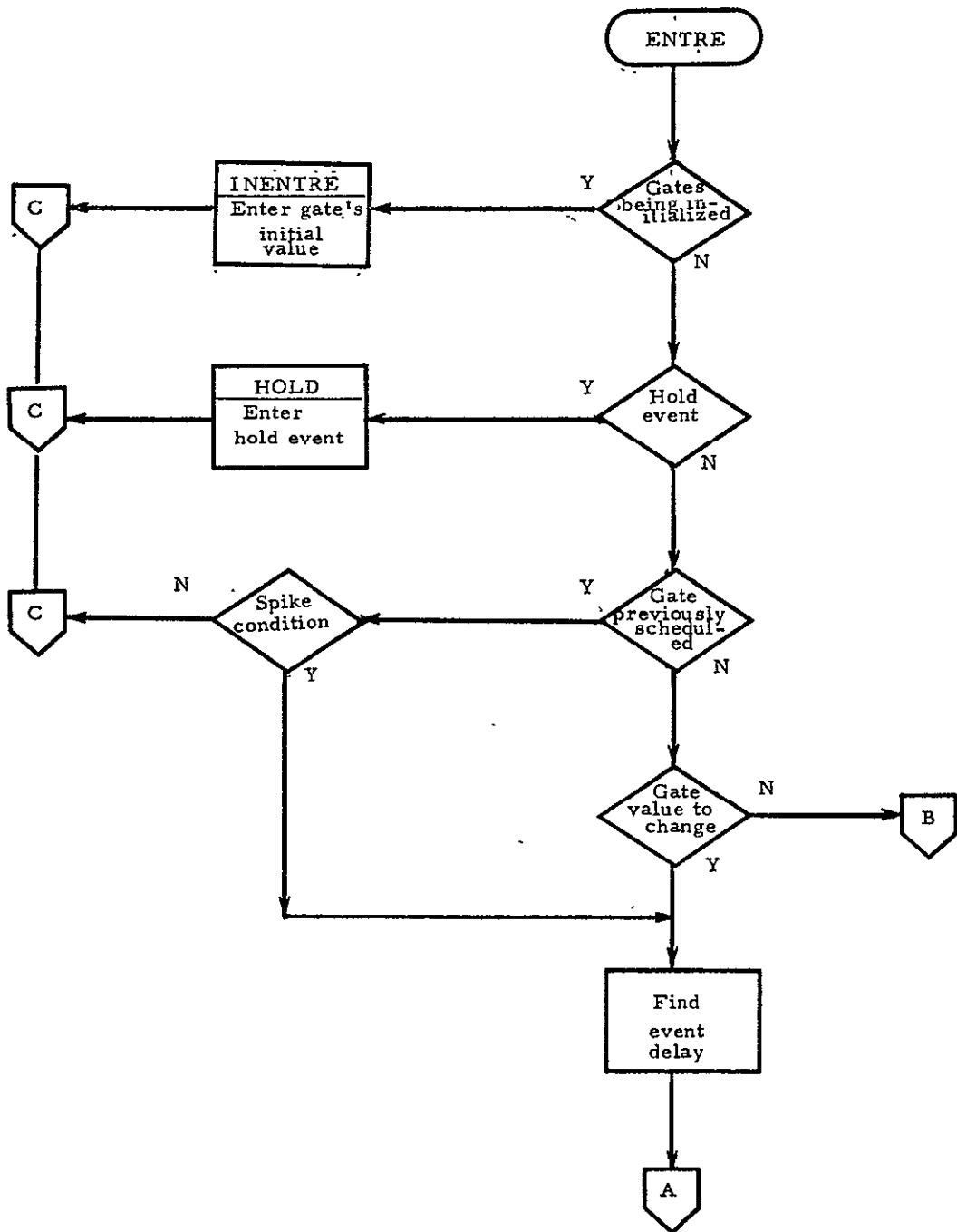


LOGIC GATE STATE DETERMINATION SUBROUTINE



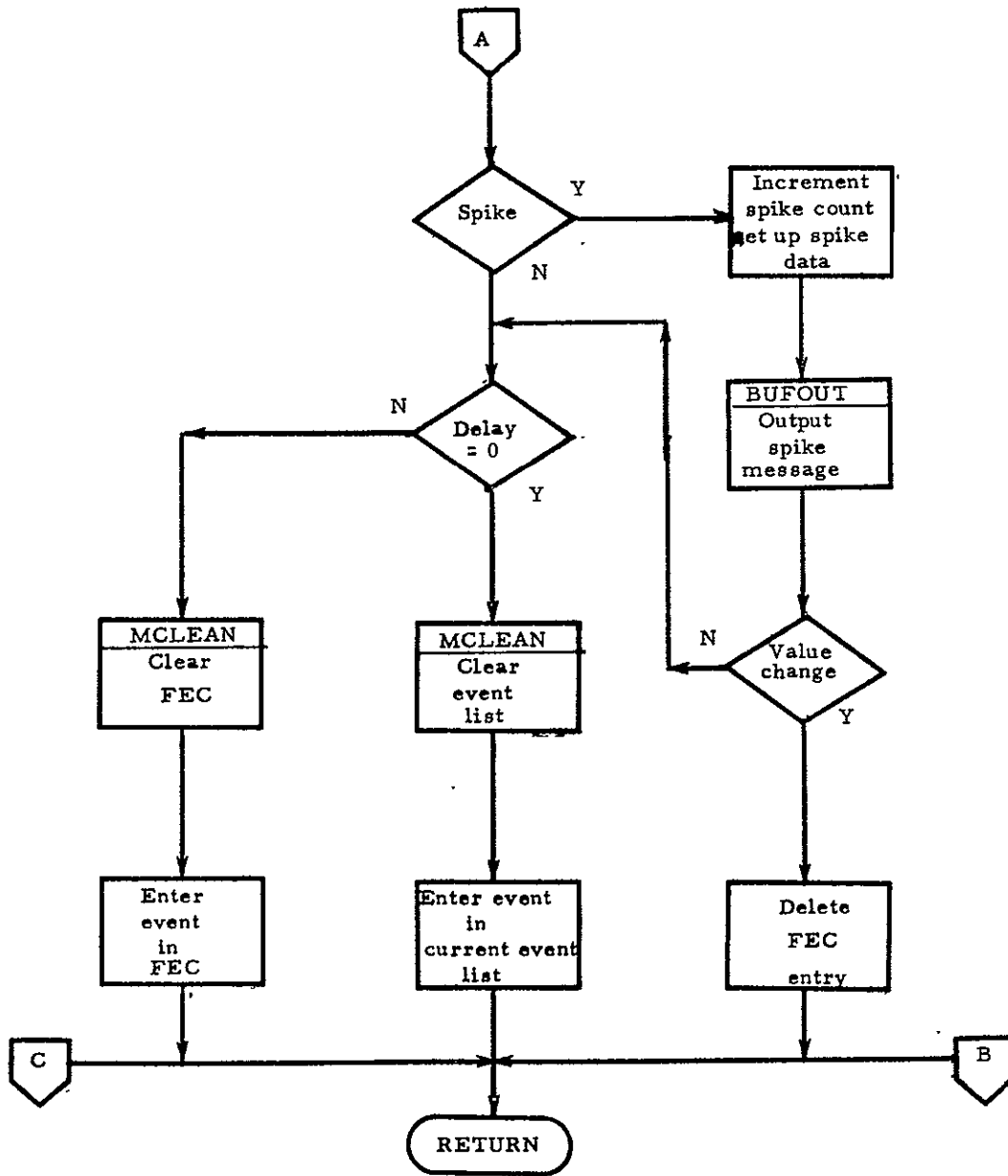
ORIGINAL PAGE IS OF POOR QUALITY

EVENT STORAGE SUBROUTINE

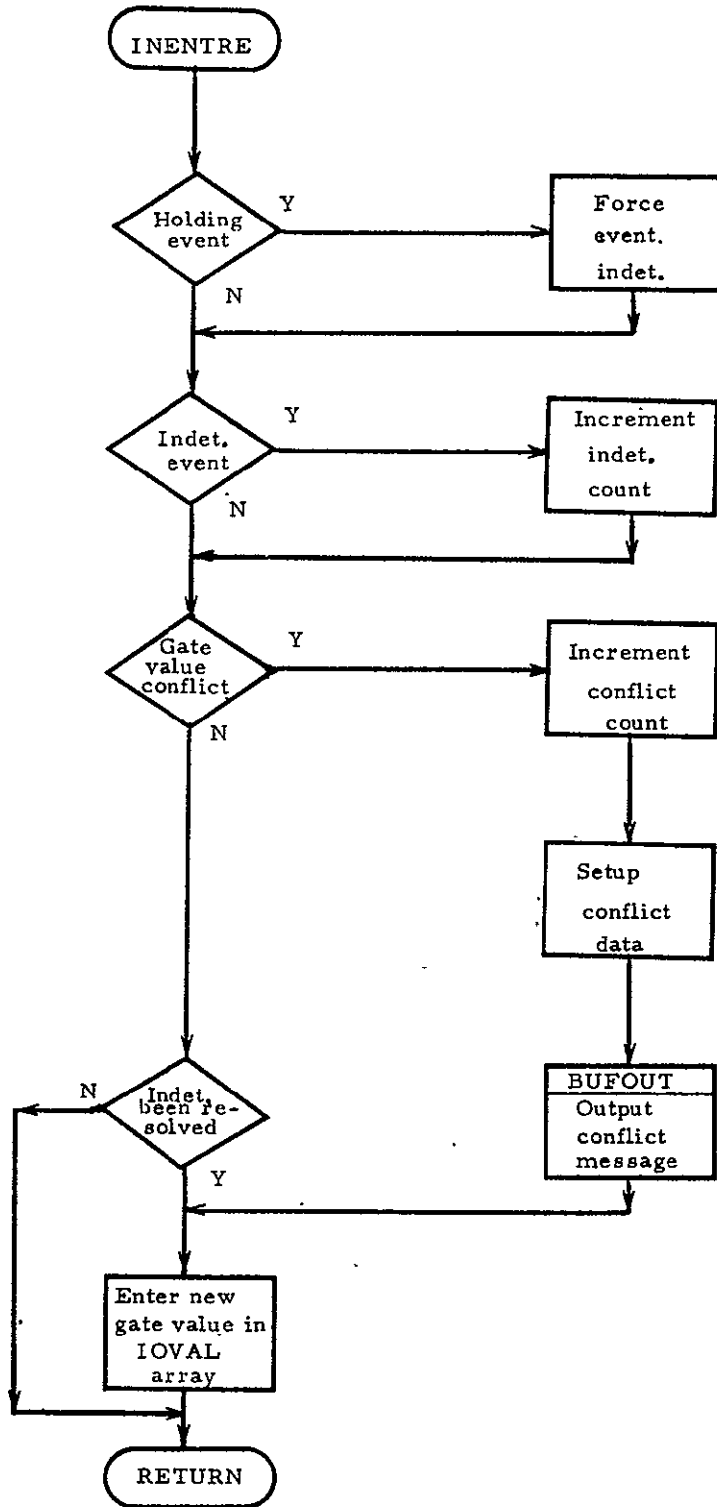


ORIGINAL PAGE IS  
OF POOR QUALITY

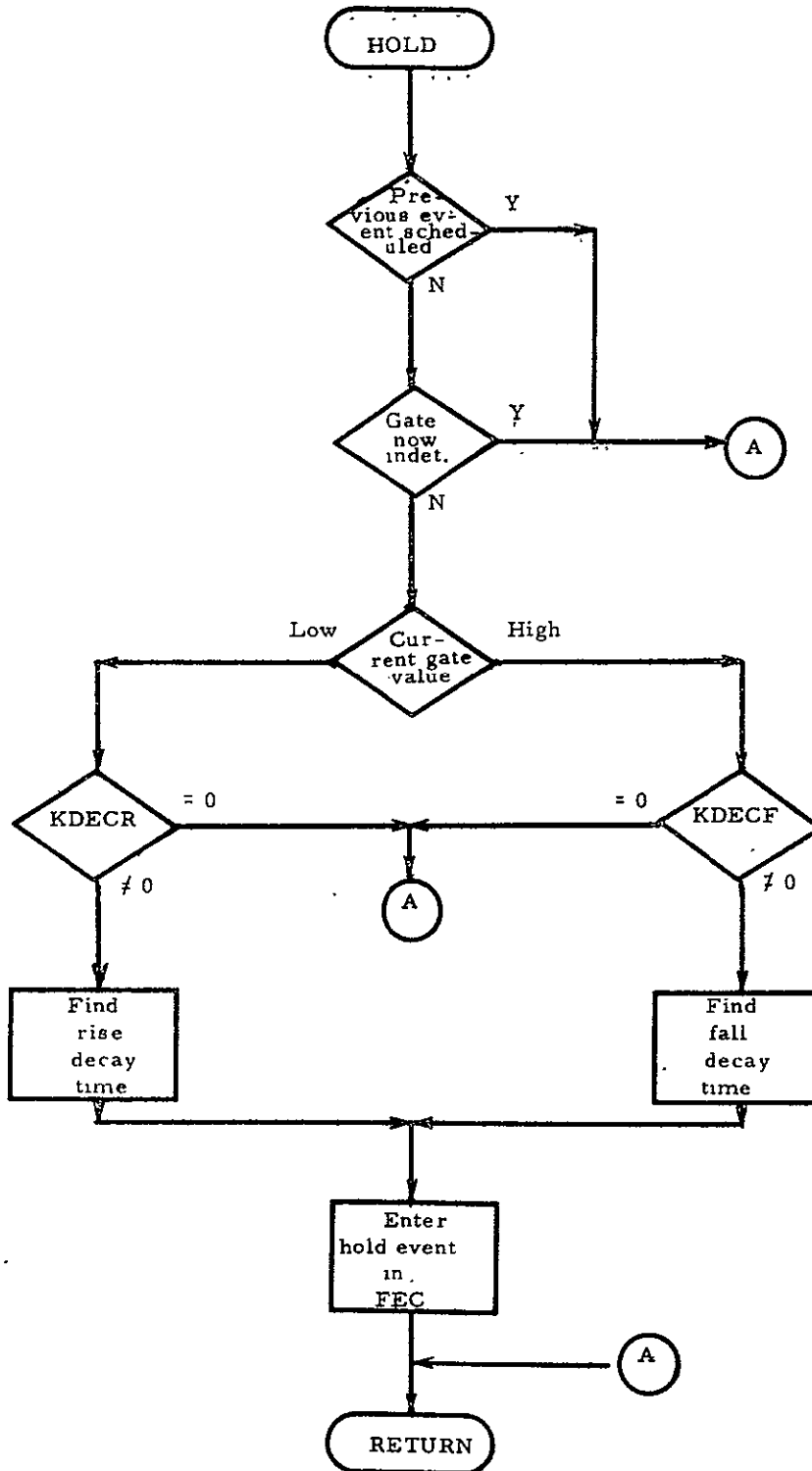
EVENT STORAGE SUBROUTINE



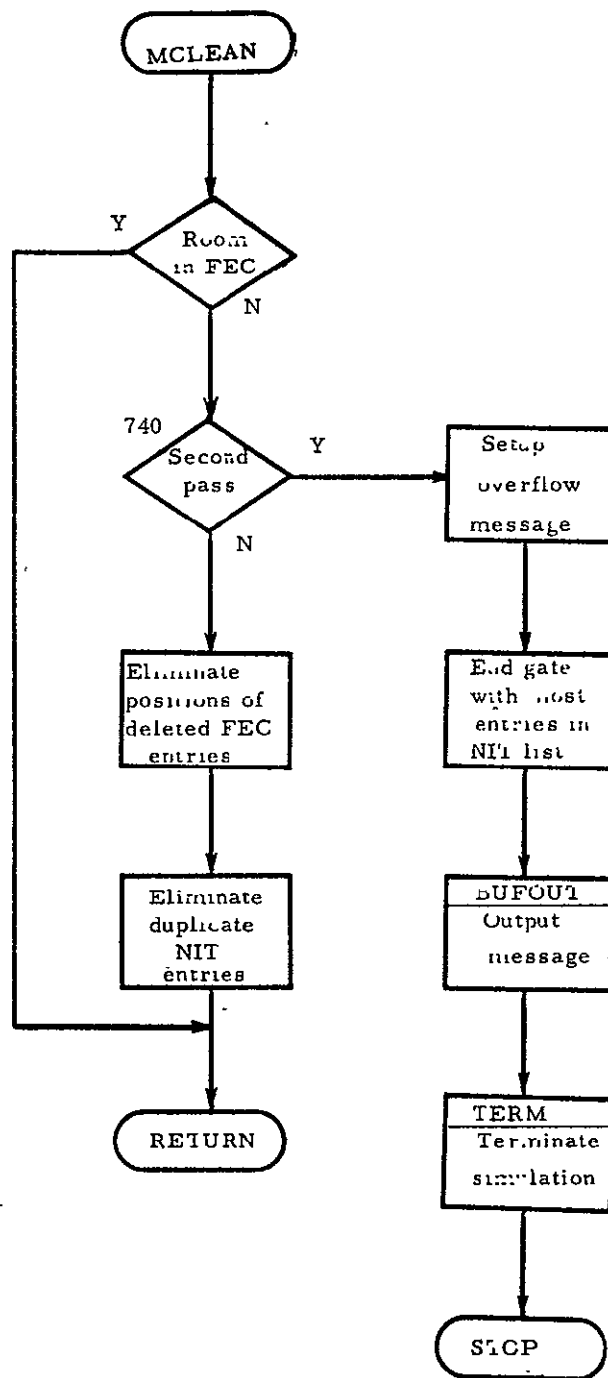
INITIAL CONDITIONS STORAGE SUBROUTINE



HOLDING MODE EVENT PROCESSING SUBROUTINE



FEC CLEARING SUBROUTINE



## APPENDIX C

### LOGSIM POSTPROCESSOR PROGRAM FLOWCHARTS

The detailed flowcharts of the LOGSIM Postprocessor are contained in this Appendix. The "Picture on a Page" technique described in Appendix A, Page A-1, and the flowchart symbol convention described in Table A2 have been used in the development of these flowcharts.

An index to the LOGSIM Postprocessor flowcharts is contained in Table C1.



# LOGSIM POSTPROCESSOR PROGRAM

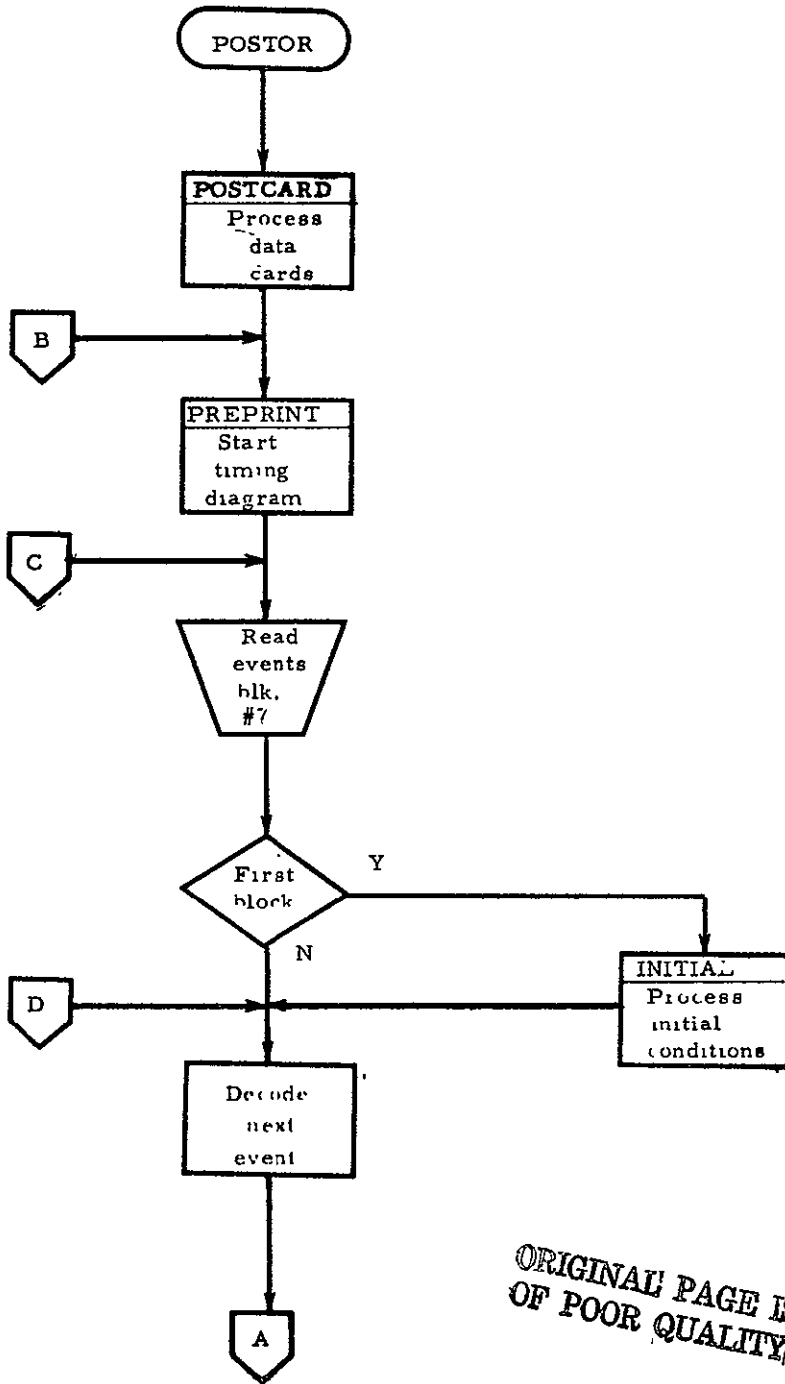
## FLOWCHART INDEX

<u>Routines</u>	<u>Page</u>
Logic Simulation Postprocessor	C-3
Card Processing Subroutine	C-4
Timing Diagram Setup Subroutine	C-5
Element Initial Conditions Processing Subroutine	C-6
Event Processing Subroutine	C-7
Control and Output Character and Shift Card Processing Subroutine	C-8
Comparison and Compare Function Card Processing Subroutine	C-9
PNT/SLOT Card Sets Processing Subroutine	C-10
Level Comparison Processing Subroutine	C-11
Spike Notice Processing Subroutine	C-12
Error Message Processing Subroutine	C-13

Table C1

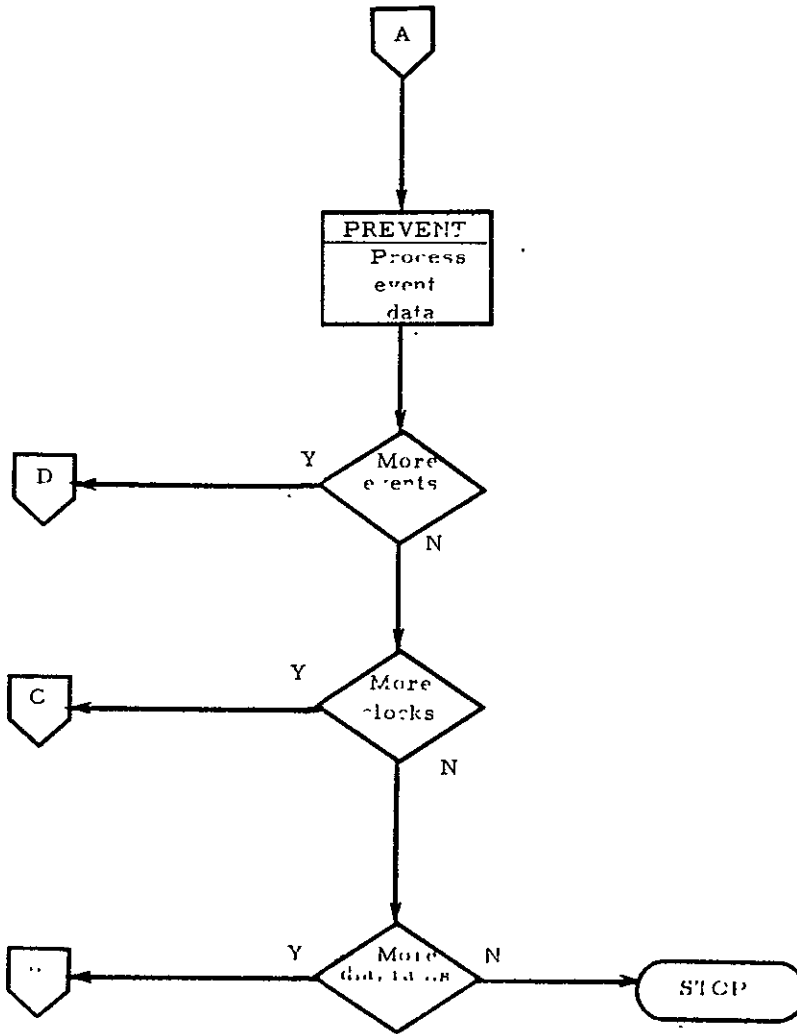
**ORIGINAL PAGE IS  
OF POOR QUALITY**

LOGIC SIMULATION POSTPROCESSOR



ORIGINAL PAGE IS  
OF POOR QUALITY

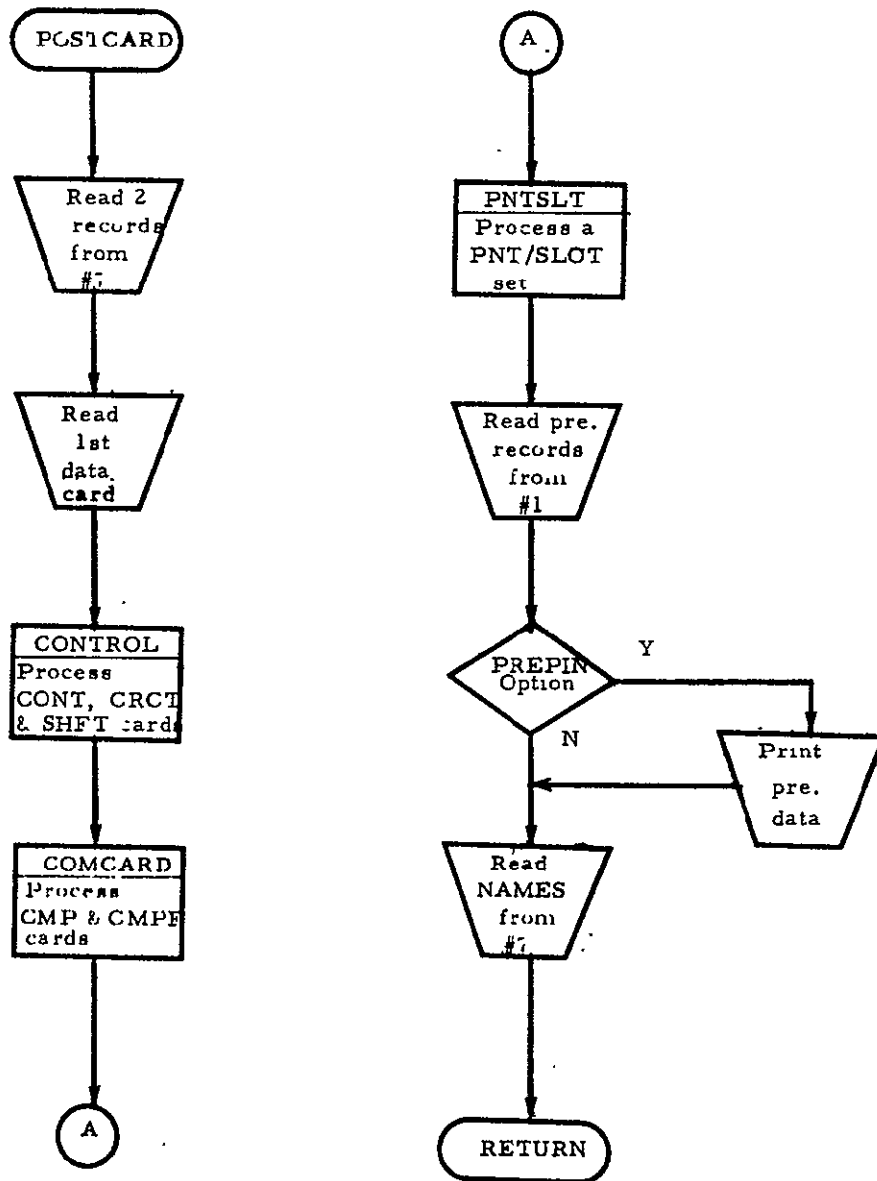
LOGIC SIMULATION POSTPROCESSOR



ORIGINAL PAGE IS  
OF POOR QUALITY

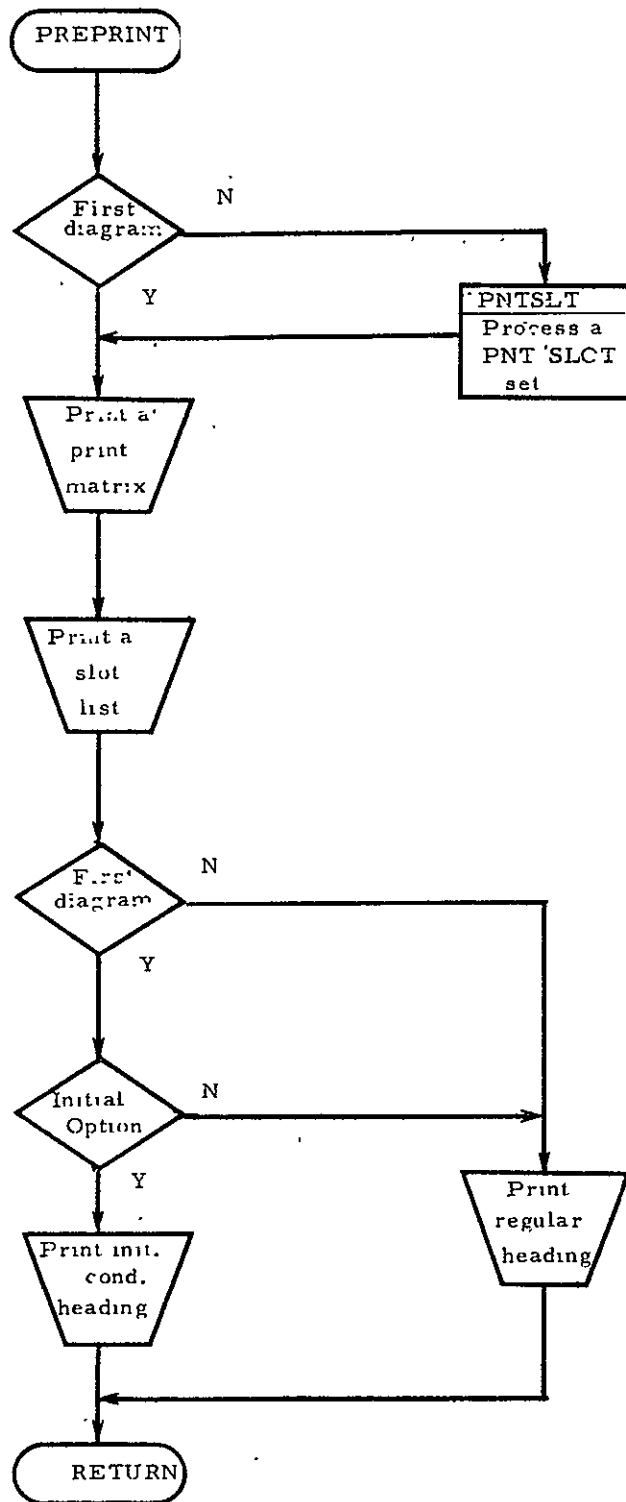
C.2

CARD PROCESSING SUBROUTINE

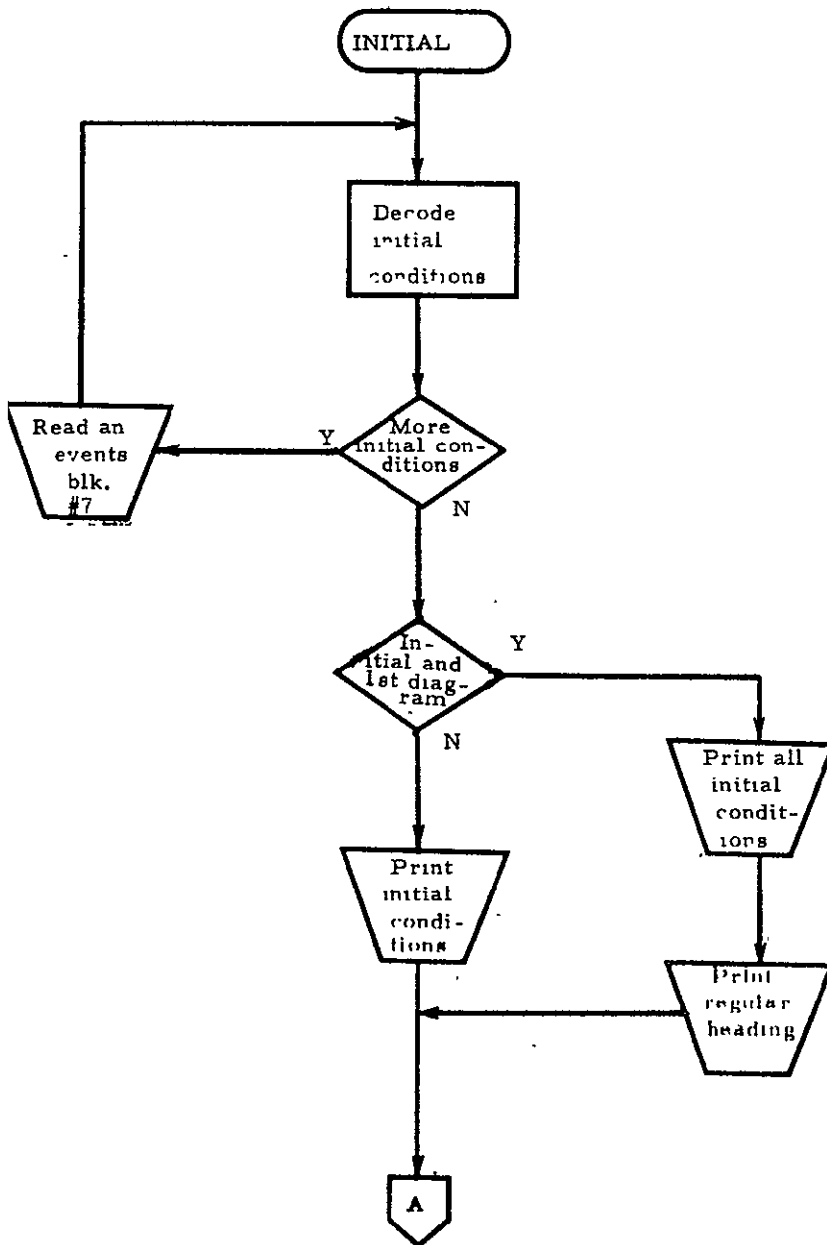


ORIGINAL PAGE IS  
OF POOR QUALITY

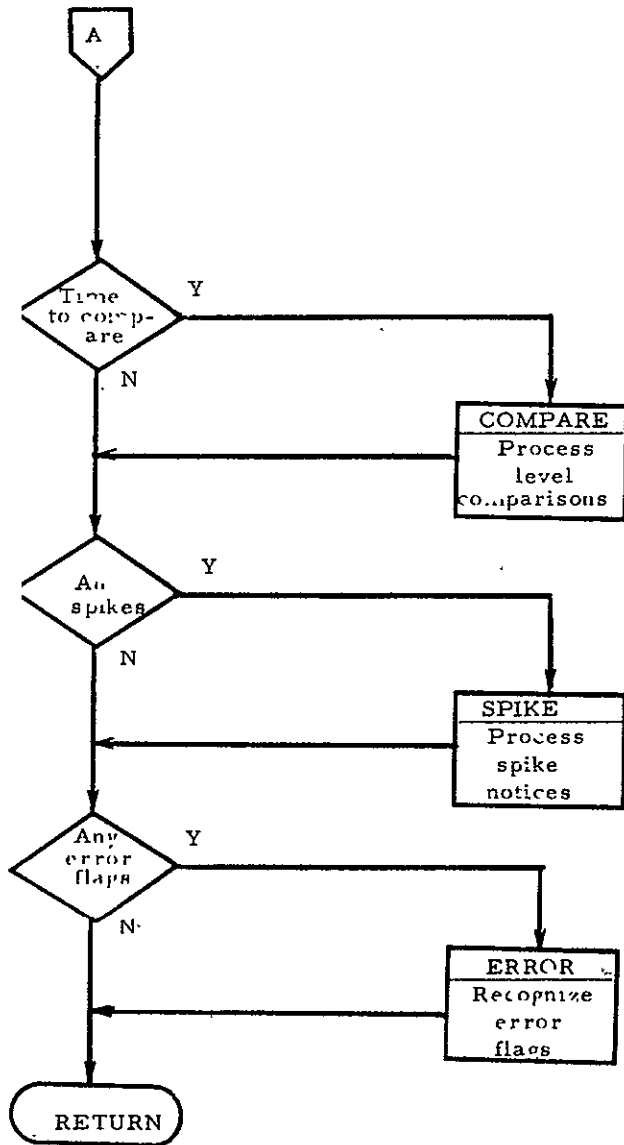
TIMING DIAGRAM SETUP SUBROUTINE



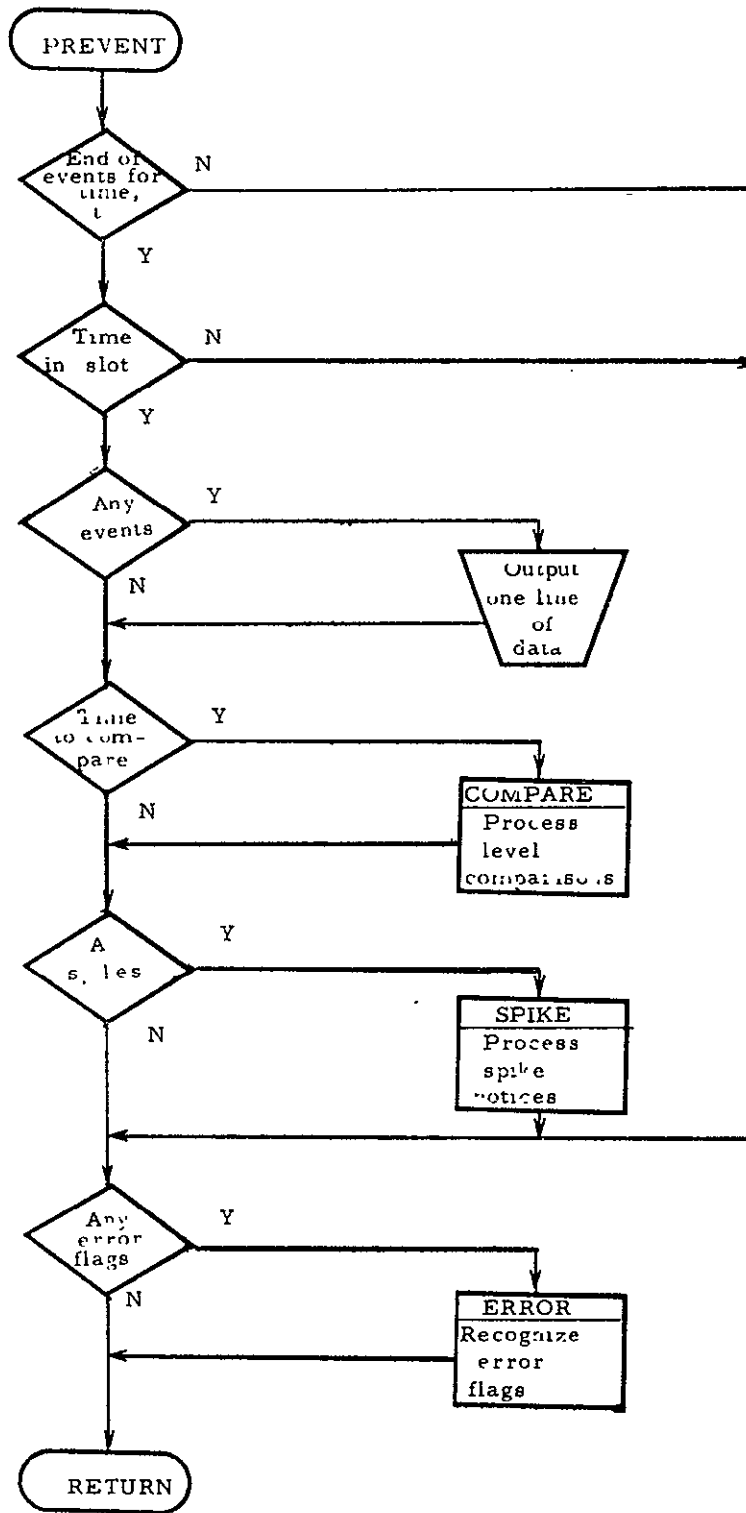
ELEMENT INITIAL CONDITIONS PROCESSING SUBROUTINE



ELEMENT INITIAL CONDITIONS PROCESSING SUBROUTINE



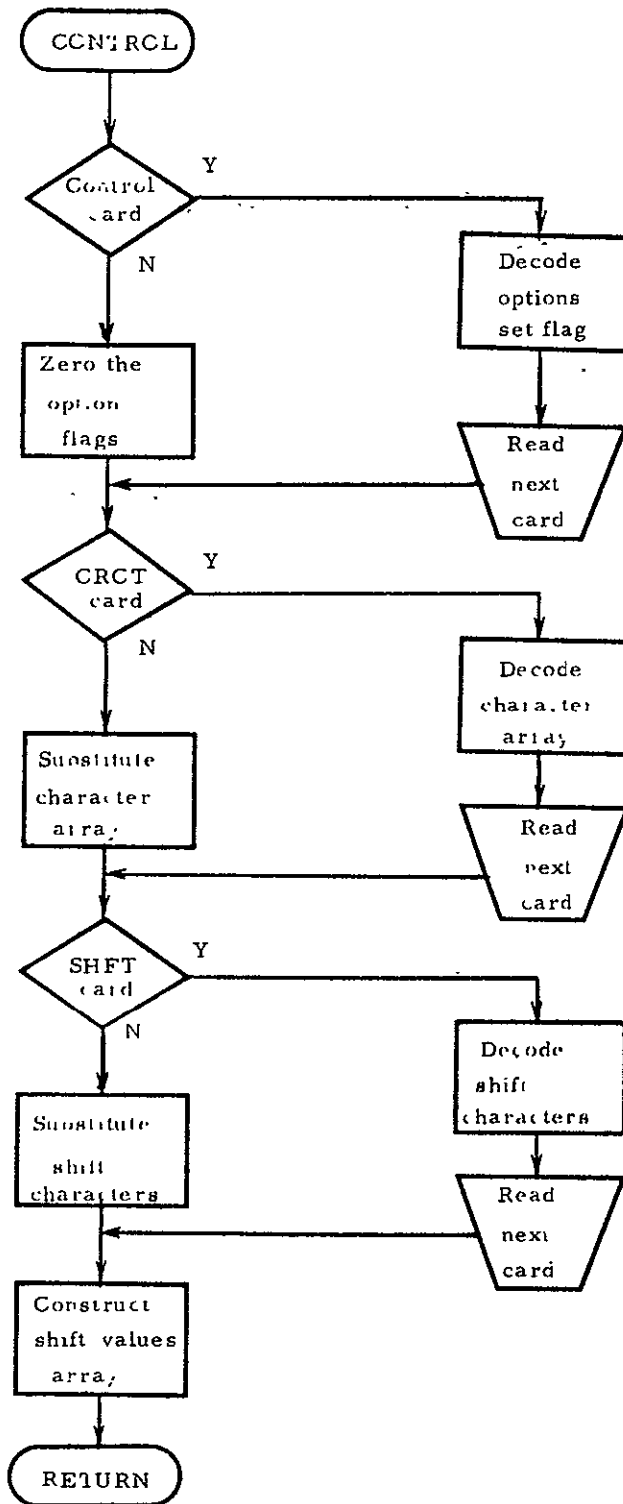
EVENT PROCESSING SUBROUTINE



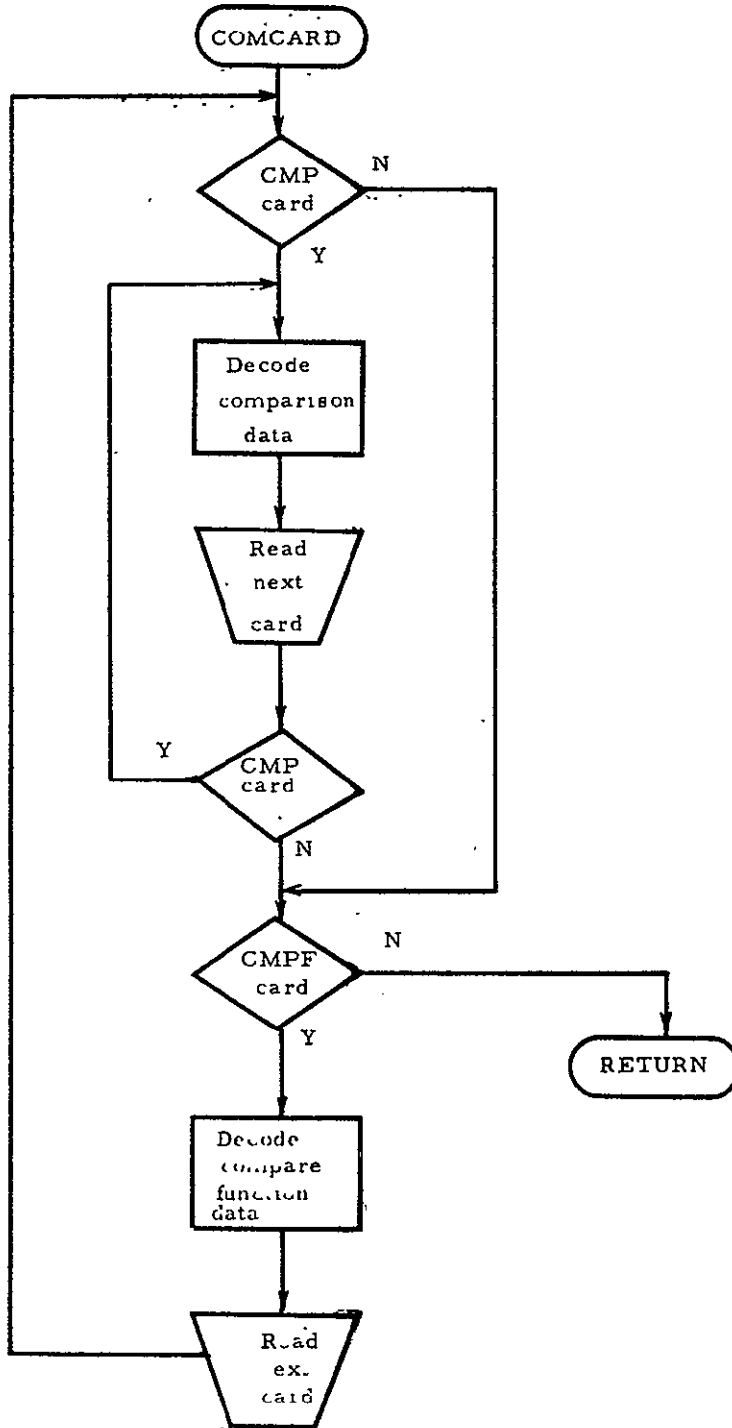
ORIGINAL PAGE IS  
OF POOR QUALITY



CONTROL AND OUTPUT CHARACTER AND SHIFT CARD PROCESSING SUBROUTINES

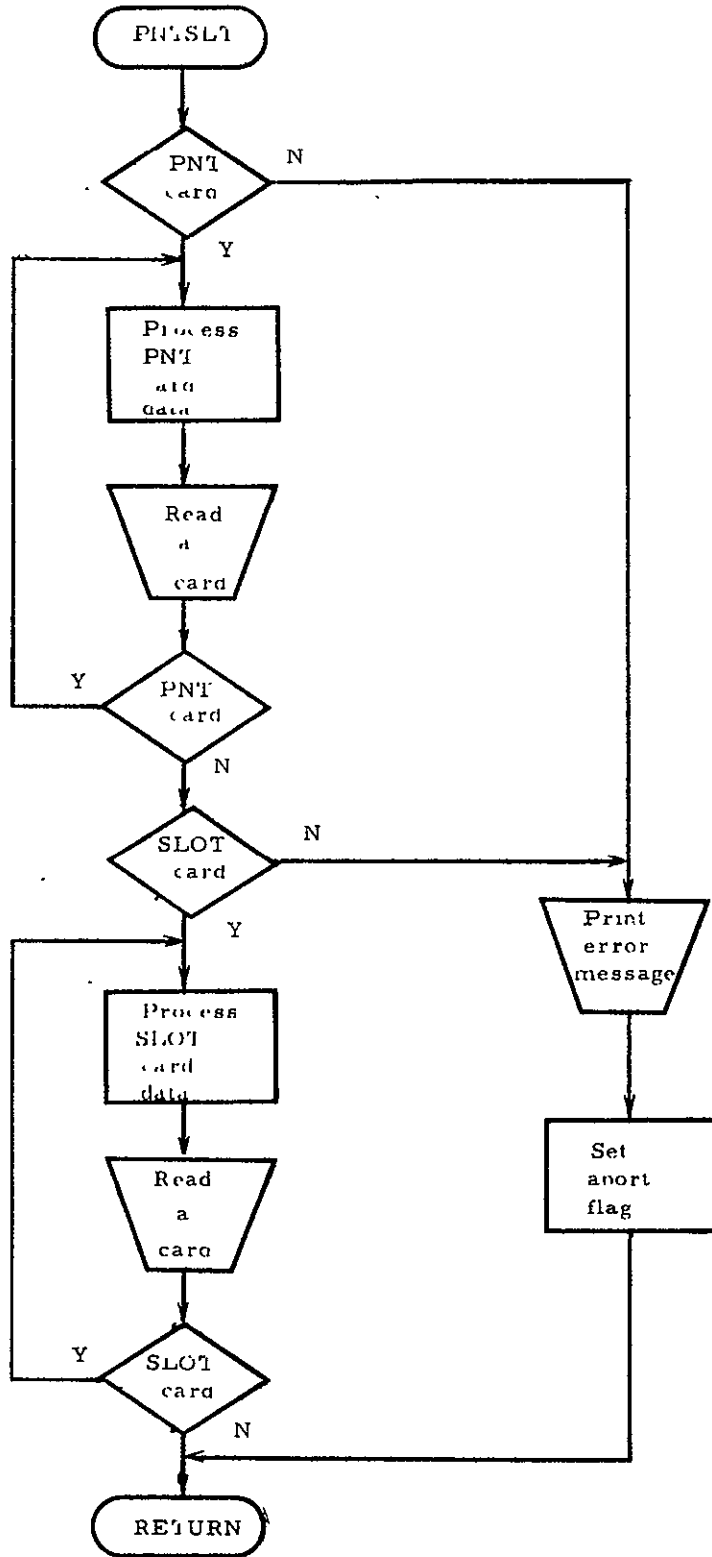


COMPARISON AND COMPARE FUNCTION CARD PROCESSING SUBROUTINE

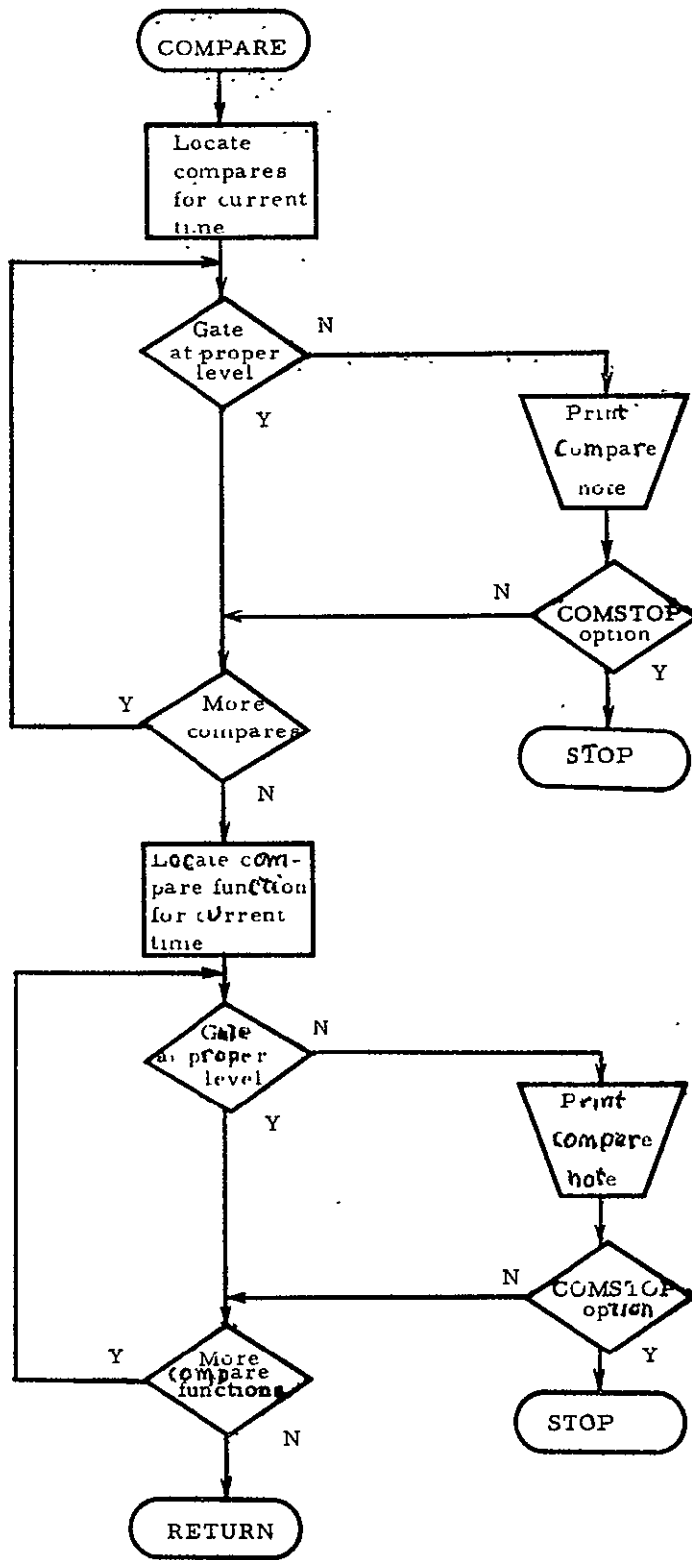


ORIGINAL PAGE IS  
OF POOR QUALITY

PNT/SLOT CARD SETS PROCESSING SUBROUTINE

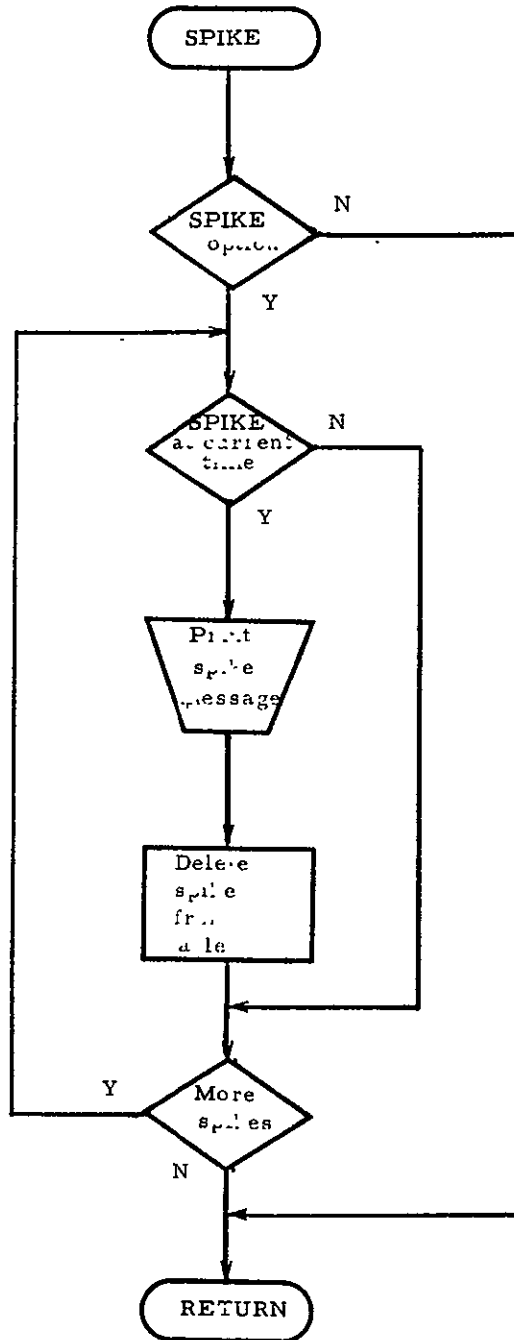


LEVEL COMPARISON PROCESSING SUBROUTINE

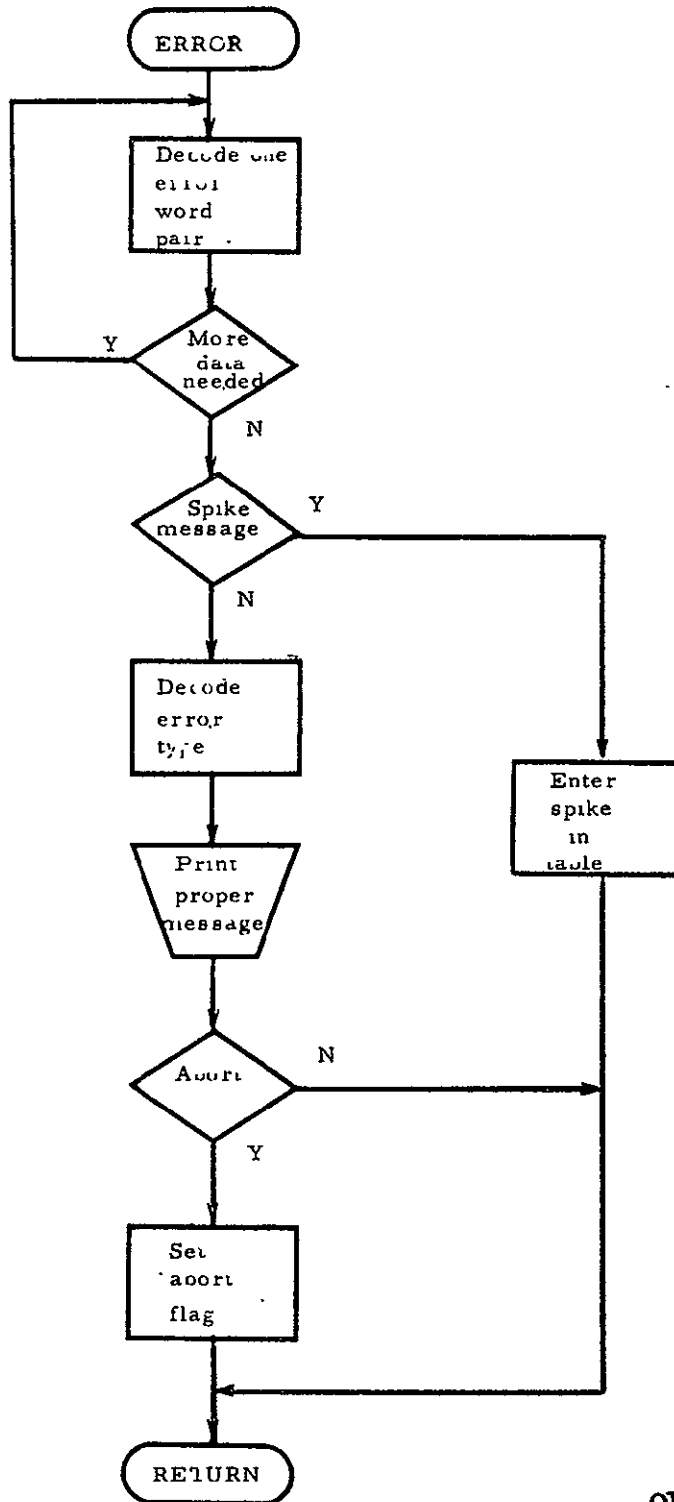


ORIGINAL PAGE IS  
OF POOR QUALITY

SPIKE NOTICE PROCESSING SUBROUTINE



ERROR MESSAGE PROCESSING SUBROUTINE



ORIGINAL PAGE IS  
OF POOR QUALITY