ESD TR-68-134
File Copy

ESD-TR-68-134

ESD RECORD COPY
RETURN TO
SCIENTIFIC & TECHNICAL INFORMATION DIVISION
(ESTI), BUILDING 1211

ESVF

MTR-564

ESD ACCESSION LIST
ESTI Call No. 63870
Copy No. 1 of 2 cys.

PROGRAM/CODING SPECIFICATION FOR THE AFICCS REPORT GENERATION (REPGEN) CAPABILITY

JANUARY 1969

N. B. Sutherland

Prepared for

DIRECTOR OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

This document has been approved for public release and sale; its distribution is unlimited.

Project 512V
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-5165

AD681840

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

ESD-TR-68-134 MTR-564

PROGRAM/CODING SPECIFICATION FOR THE AFICCS REPORT GENERATION (REPGEN) CAPABILITY

JANUARY 1969

N. B. Sutherland

Prepared for

DIRECTOR OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

This document has been approved for public release and sale; its distribution is unlimited.

Project 512V
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-5165

FOREWORD

The work reported in this document was performed by The MITRE Corporation, Bedford, Massachusetts, for the Director of Planning and Technology, Electronic Systems Division, of the Air Force Systems Command under Contract AF 19(628)-5165.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved.

WILLIAM F. HEISLER, COL, USAF
Chief, Command Systems Division
Directorate of Planning & Technology

ABSTRACT

The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 141Ø computer under the Air Force Integrated Command and Control System (AFICCS). The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, magnetic tape or on the system disk. This document contains detailed information regarding the structure and operation of the REPGEN programs, including table descriptions, core maps, program flow-charts and abstracts.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONCLUDED)

LIST OF ILLUSTRATIONS

SECTION I

INTRODUCTION

The purpose of this document is to present detailed information regarding the structure and operation of the various programs which comprise the REPGEN capability. It is assumed that the reader is familiar with the contents of the REPGEN User's Guide.

SECTION II

OVERVIEW

The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 1410 computer under the standard AFICCS system. The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, on magnetic tape or on the system disk. The output may be optionally on magnetic tape or cards. The output format is controlled by use of a quasi-English language of limited vocabulary.

There are three major activities required in order to move from source data to finished report. These are:

a. Table generation
b. File generation
c. Report production

Figure 1 shows the basic elements involved in each of these activities.

TABLE GENERATION

Before a report can be generated, certain basic information must be loaded into tables and placed on the system disk, to be available for File generation and Report production. The total REPGEN capability includes a number of support programs which are used to perform the necessary table construction.

LID (Load Input Descriptors)

The input to LID is primarily a description of the source data which will be used to make up the serial file tape (SF tape) which in turn will be the input to the output phase of REPGEN. The various source data record types must be identified, and the data attributes associated with each record type must be defined in terms of length, position within the record, data type, etc.

SAVER

In some cases, the source data may reside on the AFICCS system disk in parallel file form. In order to produce a serial file tape (SF Tape) which can be used by REPGEN output programs, a Query Language retrieval must be performed against the parallel file. The Query Language retrieval statement must specify GETTAC as the process director. GETTAC (a variant of GETB) outputs the retrieved data on an SF Tape.

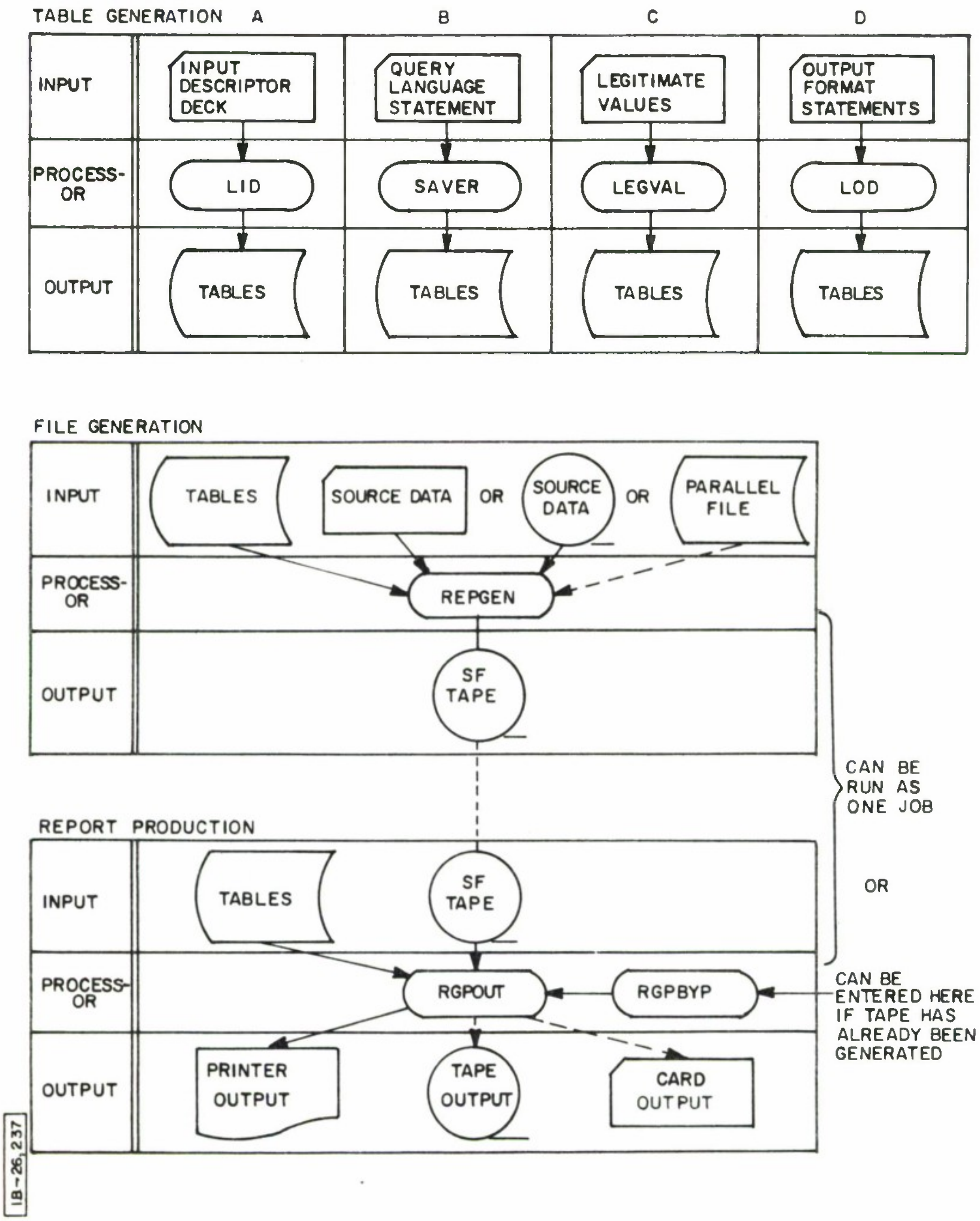


Figure 1. Overview of the Report Generation Process

SAVER's primary function is to save a Query Language retrieval statement so that it may be called up for execution when required. (SAVER is a variant of SAVE.)

LEGVAL (Legitimate Value Loader)

As information is transferred from source data to the serial file tape, it may be desirable to check certain attributes for legality. This best applies to attributes which can legally take on one of a limited number of values.

LEGVAL produces a table which contains all of the allowable values for each attribute which is to be legality checked.

LOD (Load Output Descriptors)

A quasi-English language of limited vocabulary has been implemented to give the user the capability to structure a report. Using the language provided, data can be selected from the serial file tape, arithmetically manipulated, and arranged for presentation on a printed page. (Output may optionally be to magnetic tape or punched cards.) The set of statements which direct this activity may be called an Output Program.

An output program is initially processed by LOD, which performs syntax checking, and then eliminates all redundant characters in each statement. LOD also replaces attribute names with their length and position on the serial file tape. The resulting condensed output program is placed in a table on the system disk, to be available for later execution.

FILE GENERATION

The purpose of the file generation activity is to produce a serial file tape containing the data from which the desired report(s) will be generated. The source data from which the serial file tape is made may be in punched cards, on magnetic tape, or may reside on the system disk in parallel file form.

When the source data is in cards or on tape, we are said to be operating in Mode 1; when the source data is a parallel file on disk, we are operating in Mode 2. There are internal differences in the operation of the file generation program (REPGEN), depending upon the mode in which it runs. The end result, however, is essentially the same; a serial file tape is produced which contains the data to be used in generating the desired report(s).

REPORT PRODUCTION

Once the required tables have been generated and a serial file tape has been prepared, we are ready to produce a report. This is accomplished by operating what is called the Output Phase.

The output phase can proceed as a continuation of a file generation run; at the completion of file generation the computer operator is given the option to terminate or to continue with the output phase.

A serial file tape which was produced by a previous run can be used to generate a report by operating the output phase via the Bypass Mode. In this mode, file generation is bypassed and the output phase is immediately started.

During the output phase, the condensed output program is read from disk and the statements are interpreted and executed to produce the desired output. Up to nine additional copies of a printed report may be produced automatically.

SECTION III

TABLES

REPORT DESCRIPTOR TABLE

Created By: LID

Purpose: Each entry in this table identifies by name and number an active REPGEN report.

Table Size: 1 record (expandable) of 2800 characters.

Entry Size: 100 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1-3	Report Number
4-15	Report Title
16	Input Media
17	Classification
18-47	Header Label
48-49	Size of Header Label
50-52	Low Order Position of Header Label
53-68	Blank
69	Card Indicator
70	Tape Indicator
71	Printer Indicator
72	Not Used
73	History Indicator
74-76	Save Statement Number
77-80	Low Order Position of OEDT
81-84	SDA of Selected Output Table
85-88	SDA of Record Descriptor Table
89-92	Number Records for Record Desc. Table
93-96	SDA of Output Entry Desc. Table
97-100	Number Records for OEDT

RECORD DESCRIPTOR TABLE

Created By: LID

Purpose: Each entry describes a particular type of input record which may be encountered during generation of a Serial File Tape.

Table Size: 1 record of 2800 characters allocated for each entry in the Report Descriptor Table.

Entry Size: 40 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1-5	Record Type
6-8	Low Order Position of Record Type
9	Length of Record Type
10	Input Media
11-12	Maximum Number Records per Entry
13	Number of Control Attributes
14	Number of Sub-Control Attributes
15-17	High Order Position of Remark Text
18-20	Low Order Position of Remark Text
21-32	Blank
33-36	SDA of Attribute Desc Table
37-40	Number Records for Attribute Desc Table

ATTRIBUTE DESCRIPTOR TABLE

Created By:	LID
Purpose:	Each Attribute Descriptor Table describes all of the attributes associated with a particular record type.
Table Size:	1 or more 2800 character records (sufficient to describe all attriubtes pertaining to a given record type).
Entry Size:	40 characters per entry, recorded in Move mode.
End of Table:	A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1-17	Attribute Name
18-19	Length of Attribute
20-22	Low Order Position of Attribute
23	Data Class
24	Audit Check Indicator
25	Mandatory Value Indicator
26-30	Record Type
31	Type of Attribute
32-36	Blank
37-39	Low Order Position of the Attribute Value in the Output Entry File
40	Not Used

SELECTED OUTPUT TABLE (SOT)

Created By: LID. The space is allocated by LID, but table entries are actually loaded by LOD.

Purpose: Each SOT table acts as a directory for up to seven output programs which may be associated with a given report number.

Table Size: 1 record of 245 characters (only the first 210 characters are actually used) is allocated for each entry in the Report Descriptor Table.

Entry Size: 30 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1	Entry Indicator
2-5	SDA of Condensed Output Format Statements
6	Output Media
7-10	Output Entry Size for this Report
11	History Tape Indicator
12-14	Highest Statement Number for this Output Program
15-16	Total Output Statements Fitting into Core with this Format
17-21	Core Origin of Output Statements for this Output Format
22-26	Core Origin of Input Area for this Output Format
27	Usually Blank. Set to 1 by UPSTAT to indicate that the condensed statements to which this SOT entry points have been fully condensed by UPSTAT.
28-30	Blank

OUTPUT ENTRY DESCRIPTOR TABLE (OEDT)

Created By: LID

Purpose: This table describes the format of the records on the Serial File Tape.

Table Size: 1 record of 2800 characters.

Entry Size: 25 characters per entry, written in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1-17	Attribute Name
18	"Zero" inserted by LID
19-20	Length of Attribute
21-23	Low Order Position of Attribute
24-25	Blank - Not Used

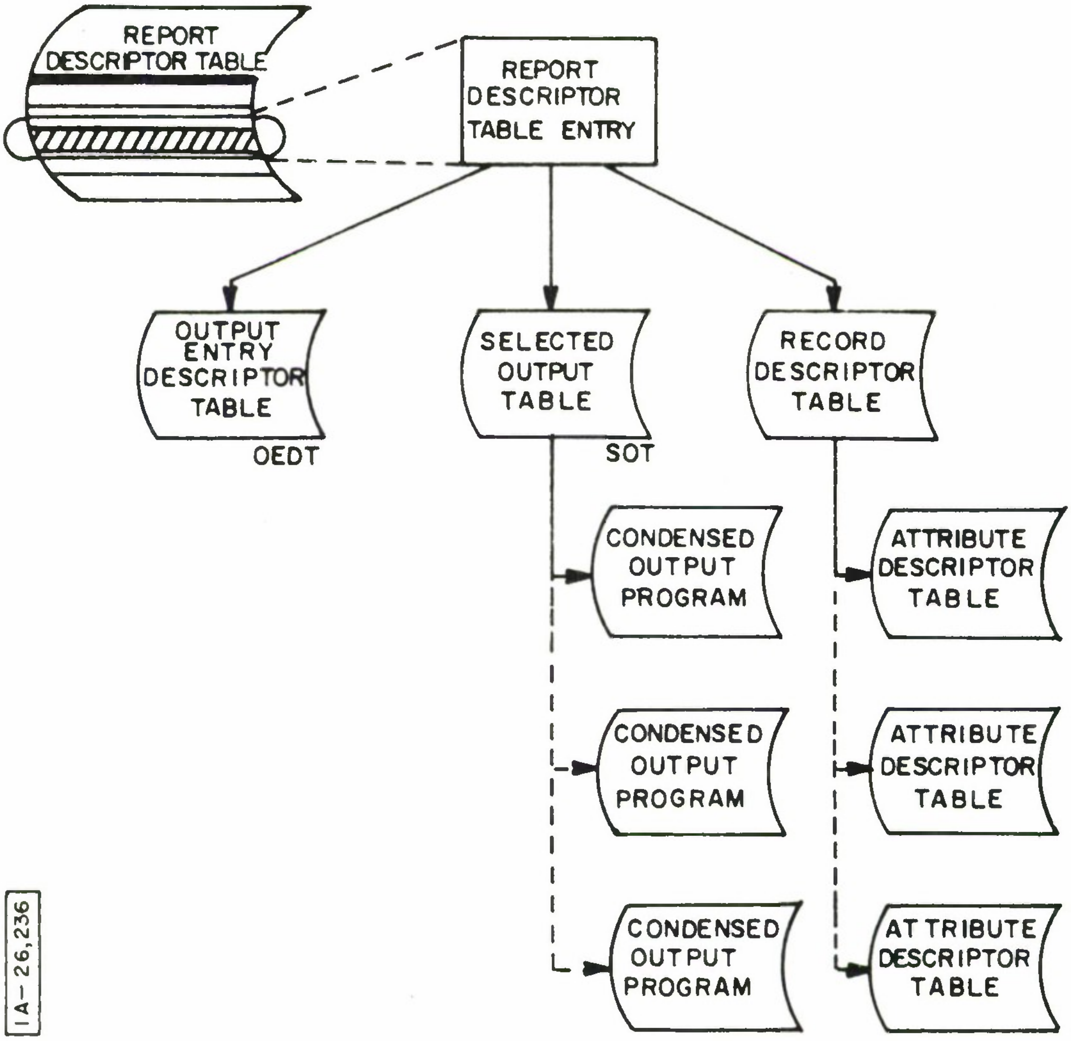


Figure 2. Tables Associated with One Report Descriptor Table Entry

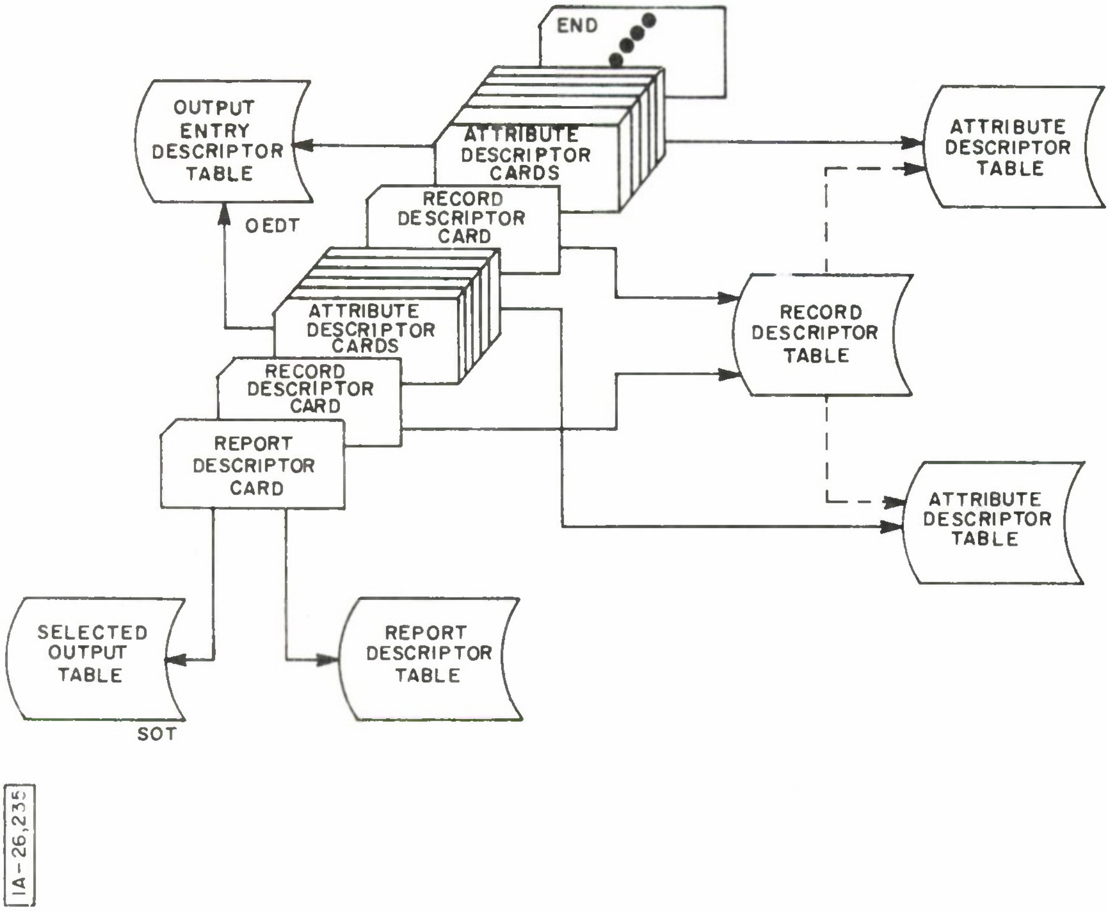


Figure 3. LID Input and Output, Mode 1

LEGITIMATE VALUE ATTRIBUTE NAME TABLE

Created By:	LEGVAL
Purpose:	This table contains the name of every attribute to be checked for legality during file generation, and acts as a directory for the lists of legal values.
Table Size:	10 records of 245 characters each.
Entry Size:	24 characters per entry, recorded in Move mode.
End of Table:	A record mark in the first position of an entry indicates the end of the table.

Entry Description:

Positions	Contents
1-17	Attribute Name
18-21	SDA of Legitimate Values for this Attribute.
22	Number of records assigned to above SDA.
23-24	Maximum length of attribute value.

LEGITIMATE VALUE TABLE

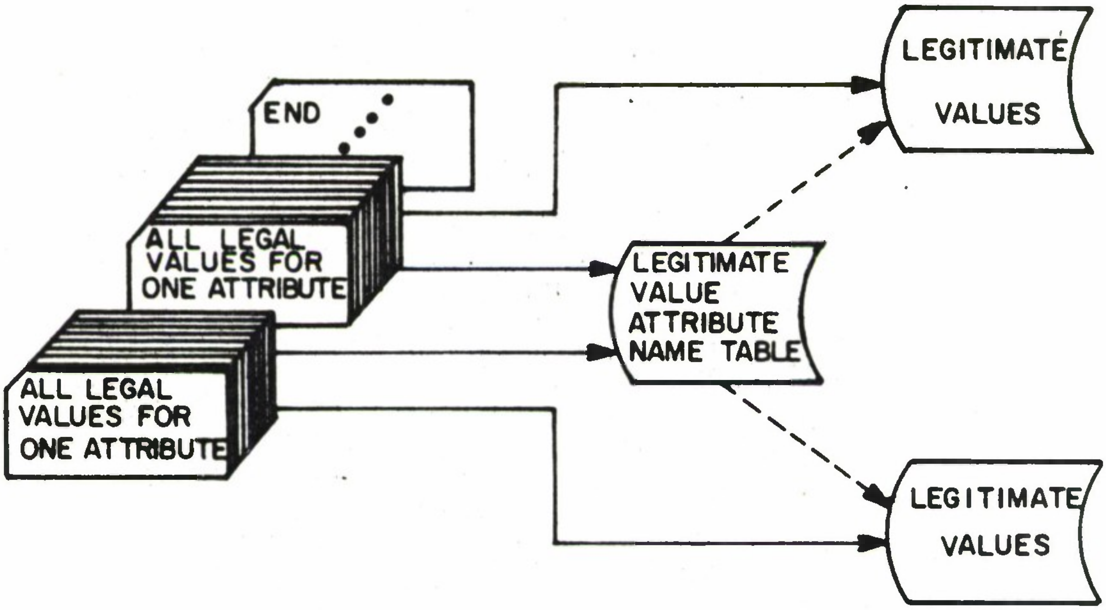
Created By: LEGVAL

Purpose: Each Legitimate Value Table contains the list of all allowable values for one particular attribute.

Table Size: 4 records of 245 characters each.

Entry Size: Variable. The entries in a given Legitimate Value Table are as long as the maximum length of that particular attribute value.

End of Table: A record mark in the first position of an entry indicates the end of the table.


END
ALL LEGAL VALUES FOR ONE ATTRIBUTE
ALL LEGAL VALUES FOR ONE ATTRIBUTE
LEGITIMATE VALUES
LEGITIMATE VALUE ATTRIBUTE NAME TABLE
LEGITIMATE VALUES

IB-26,238

Figure 4. LEGVAL Input and Output

CONDENSED OUTPUT PROGRAM

Created By: LOD

Purpose: Each table contains one Output Program, processed and condensed by LOD and available for execution.

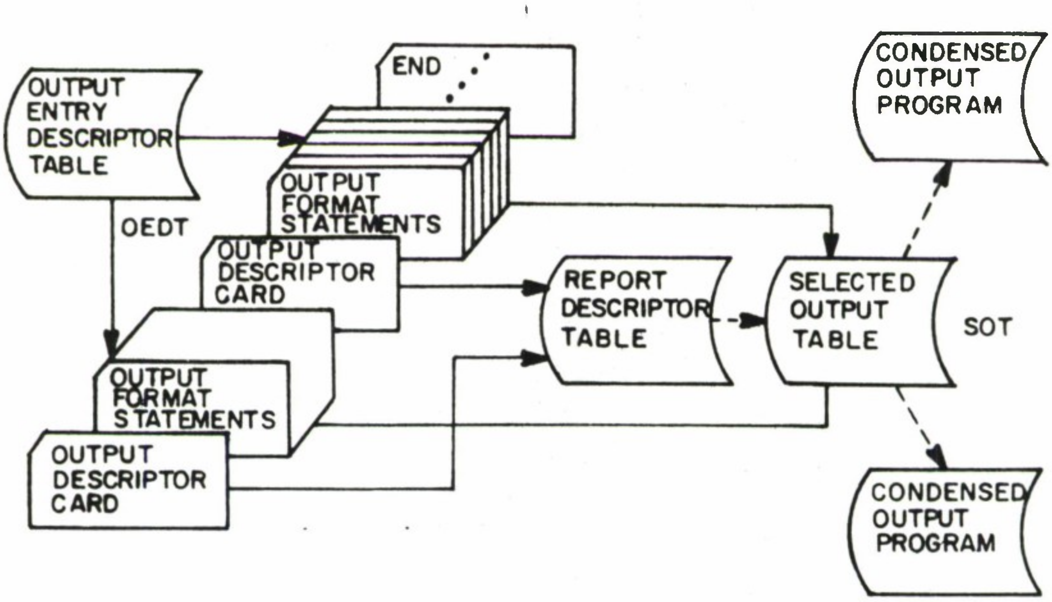
Table Size: The table contains one 245 character record for each statement of the Output Program (only the first 200 characters of each record are actually available for use).

Entry Size: Each entry (condensed statement) is contained in a 245 character record written in Move mode.

End of Table: Each condensed statement is assigned within an SDA to the relative record corresponding to the statement number. Thus, the last entry in the table is contained in the relative record corresponding to the highest statement number used in the Output Program.

Entry Description:

See program description of LOD.



IB-26,239

Figure 5. LOD Input and Output

SAVED STATEMENT TABLE DIRECTORY

Created By: SAVER

Purpose: This table serves as a directory for locating the Query Language statements which have been stored on the system disk by SAVER.

Table Size: 1 record of 2800 characters (only the first 2000 are used).

Entry Size: Each entry is 4 characters in length.

End of Table: The end of the table is indicated by the 500th entry or a blank entry, whichever occurs first.

Entry Description:

Each entry is an SDA of a saved Query Language statement. The relative position of the entry in the table corresponds to the statement number previously assigned to the saved statement.

SAVED STATEMENT TABLE

Created By: SAVER

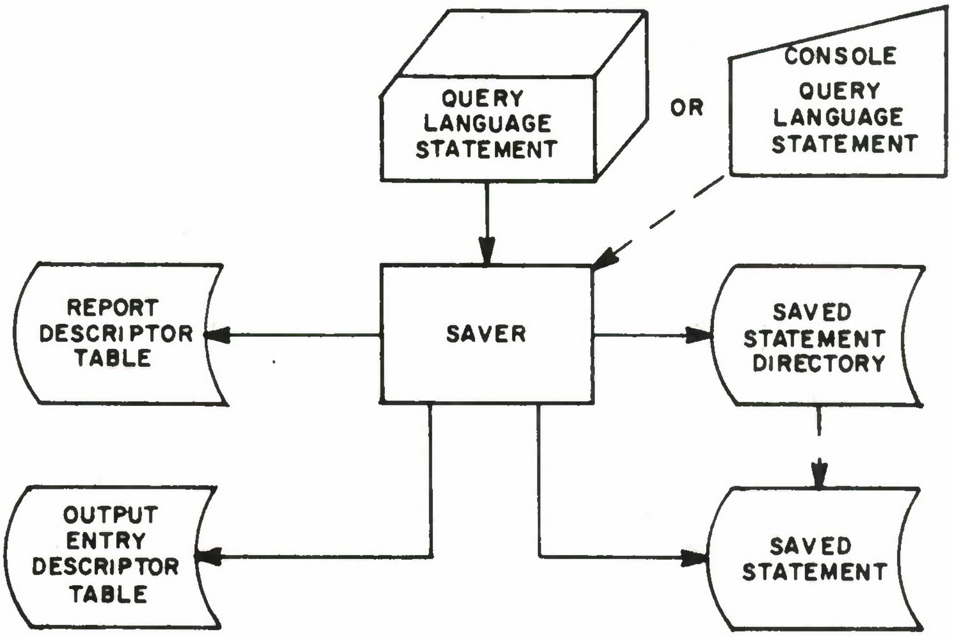
Purpose: Query Language retrieval statements are stored on the system disk for subsequent recall and execution.

Table Size: A maximum of 500 2800 character records.

Entry Size: Each saved statement is contained in one 2800 character record.

Entry Description:

See program description for SAVER.


QUERY LANGUAGE STATEMENT
OR
CONSOLE QUERY LANGUAGE STATEMENT
REPORT DESCRIPTOR TABLE
SAVER
SAVED STATEMENT DIRECTORY
OUTPUT ENTRY DESCRIPTOR TABLE
SAVED STATEMENT

1A-26,234

Figure 6. SAVER Input and Output

I/O COMMUNICATION RECORD

Created By: REPGEN (or by RGPBYP when using bypass mode).

Purpose: This record contains information which must be passed from the input phase to the output phase.

Table Size: 1 record of 245 characters.

Entry Size: 1 entry of 126 characters.

Entry Description:

Positions	Contents
1-100	Report Descriptor Table entry for report being processed.
101-117	AS OF TIME entered on console
118-120	High order position of Remarks text
121-123	Low order position of Remarks text
124	Format number of requested Output Program
125	Classification code of requested report
126	Number of additional copies requested

SECTION IV

OUTPUT PHASE COMMUNICATION ZONE

Name	Loc	Length	Tag	Meaning
WAREA	25ØØ	2ØØ	High	Area from which a condensed statement is executed.
ENTIND	27ØØ	1	Low	Indicator for entry from SOT to be used. A or C = active; D or E or ḇ = inactive.
ENDSDA	27Ø4	4	Low	SDA of condensed statements for this Output Program.
ENTMED	27Ø5	1	Low	Output device for this Output Program. C = card punch, T = tape, P = printer.
ENTSIZ	27Ø9	4	Low	Size of the output image for this Output Program.
ENTHIS	271Ø	1	Low	History tape indicator. Y = yes, ḇ = no.
ENTHIH	2713	3	Low	Highest statement number for this condensed Output Program.
ENTAMT	2715	2	Low	Number of condensed statements that can fit in core for this Output Program.
ENTSTA	272Ø	5	Low	Core Origin of the condensed statements for this Output Program.
ENTINP	2725	5	Low	Core origin of the input area for this Output Program.
ENTRY	2729	4	Low	The low-order character of the SOT entry in use.
SOT	273Ø	245	High	Selected Output Table for this report.
OVFLOW	2976	2	Low	The line after which overflow is to occur.
COUNTR	2977	51 x 17	High	The 5Ø Counters available to an Output Program, plus the REMAINDER Counter (Counter Ø).

Name	Loc	Length	Tag	Meaning
ASFDAT	386Ø	17	Low	Location where AS OF TIME from console is stored.
DATE	3877	17	Low	Location where system date is stored (referred to by system attribute name TODAY).
CLASS	3894	17	Low	Report classification.
SWITCH	3895	1Ø	High	The 9 Switches available to an Output Program.
LNCTER	39Ø6	2	Low	Line counter.
COM1NO	39Ø9	3	Low	The statement number of the first level statement in operation.
COM1PO	3914	5	Low	The position in the first level statement in operation.
COM1AC	3915	1	Low	First level active indicator; 1 = active; Ø = inactive.
COM2NO	3918	3	Low	The statement number of the second level statement in operation.
COM2PO	3923	5	Low	The position in the second level phrase in operation.
COM2AC	3924	1	Low	Second level active indicator; 1 = active; Ø = inactive.
COMREP	3927	3	Low	The statement number of the REPEAT statement in operation.
COMSOR	3932	5	Low	The core origin of the condensed statements.
COMSNO	3934	2	Low	The number of statements in core.
COMIOR	3939	5	Low	The core origin of the input area.
COMIEN	3944	5	Low	The end of the input area in core.
COMOOR	3949	5	Low	The core origin of the output area.

Name	Loc	Length	Tag	Meaning
COMOEN	3954	5	Low	The end of the output area in core.
COMTES	3955	1	Low	The result of the last IF test; 1 = true; Ø = false.
COMPUN	3956	1	Low	The punctuation from the last phrase passed.
COMDON	3957	1	Low	Finished indicator; 1 = no more input data.
COMSTK	3958	1	Low	The stacker into which the next card to be punched will be placed.
COMQL	3959	1	Low	Query Language indicator. Non blank = Query Language.
COMTYP	396Ø	1	Low	The statement type of the last statement operated.
COMDR	3961	1	Low	Type of first-level statement last executed; D = DO; R = REPEAT.
COMOVF	3962	1	Low	Overflow indicator.
HILO	3968	6	Low	The high and low order addresses of remarks data on remarks card.
COMNEW	3969	1	Low	Indicator that a new statement has become active.
COPIES	397Ø	1	Low	Number of additional copies of the report requested.
NEWPAG	3971	1	Low	Indicator that a SKIP TO NEW PAGE was executed.
SPACES	3973	2	Low	Number of blank lines printed by the last SKIP statement.
WRTSCT	3975	2	Low	Not used.

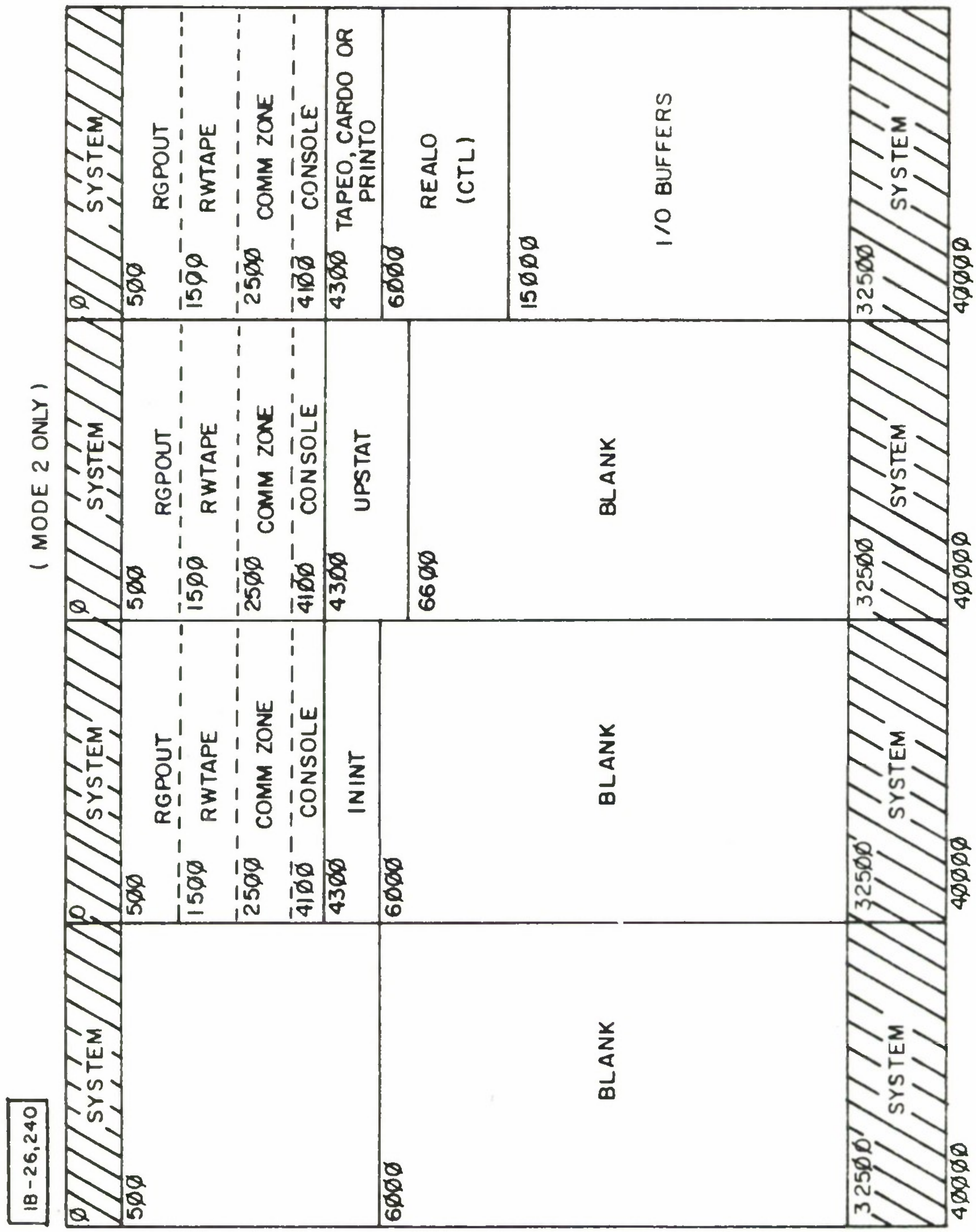


Figure 7. REPGEN Output Phase Core Map

SECTION V

SYMBOLIC DISK ADDRESSES

The REPGEN programs make frequent use of the system disk. All disk references are made through the use of DSKACC, using Symbolic Disk Addresses (SDA's). Several programs in REPGEN use the system allocation routine DSKALL to dynamically obtain needed SDA's. However, there are a number of SDA's which must be obtained prior to installing the REPGEN capability. These SDA's are assembled directly into some of the REPGEN programs.

Table 1 which follows describes the characteristics of the disk areas for which SDA's must be obtained prior to installing REPGEN.

TABLE 1

SDA'S REQUIRED BY REPGEN PROGRAMS

SDA Explanation	Mode Code / # of Records	# Char. Per Rec.	USED BY																			
			LID	LOD	LEGVAL	LTID	TAPEFM	REPGEN	RGPBYP	RGPOUT	REALO	CARDO	TAPEO	PRINTO	ININT	UPSTAT	REMARK	CHGLOD	SAVER	REKALL	DELEET	REPEET
General Tape Descriptor Table	3/01	2165				X	X															
Typewriter Input Area	4/1440	2800					X	X	X	X								X	X		X	X
Current Date	8/01	245					X	X							X							
Scratch Area	4/8429	2800					X	X									X					
I/O Communication Record	8/01	245						X	X						X	X						
LEGVAL Attribute Name Table	8/10	245			X			X														
Report Descriptor Table	4/01	2800	X	X				X	X									X	X			
Communications Zone	5/04	400								X					X							
CARDO Routine	3/01	2165								X												
TAPEO Routine	5/04	400								X												
PRINTO Routine	5/05	400								X												
REALO Routine	3/04	2165								X												
ININT Routine	5/04	400								X												
UPSTAT Routine	5/06	400								X												
REMARK Routine	3/02	2165								X												
Temporary Header Storage	6/02	529											X									
History Tape Overlay Area	5/05	400										X	X									
Table of Saved Statement SDA's	4/01	2800																	X	X	X	X

SECTION VI

REPGEN FLOWCHARTS AND ABSTRACTS

6.1 LID

Purpose

LID reads a deck of input descriptor cards and generates a set of tables which are required by subsequent REPGEN programs.

Calling Sequence

Enter LID on the console typewriter. Return will always be to IDLOOP.

Description

There are four acceptable input card types to LID. They are the Report Descriptor Card, the Record Descriptor Card, the Attribute Descriptor Card and the END Card.

Report Descriptor Card -- When LID reads a Report Descriptor Card it does the following:

a. Checks the Report Descriptor Table to make sure there is an open slot or that this is not a duplicate of another report number.

b. If the input medium declared on this card is not disk, an SDA is allocated for a Record Descriptor Table.

c. An SDA is allocated for an SOT.

d. An SDA is allocated for an OEDT.

Record Descriptor Card -- When LID encounters a Record Descriptor Card it:

a. Allocates an SDA for an Attribute Descriptor Table.

Attribute Descriptor Card -- An Attribute Descriptor Card causes LID to:

a. Make an entry in the current Attribute Descriptor Table.

b. Read the next Attribute Descriptor Card and repeat the process until all attributes for this record type have been treated.

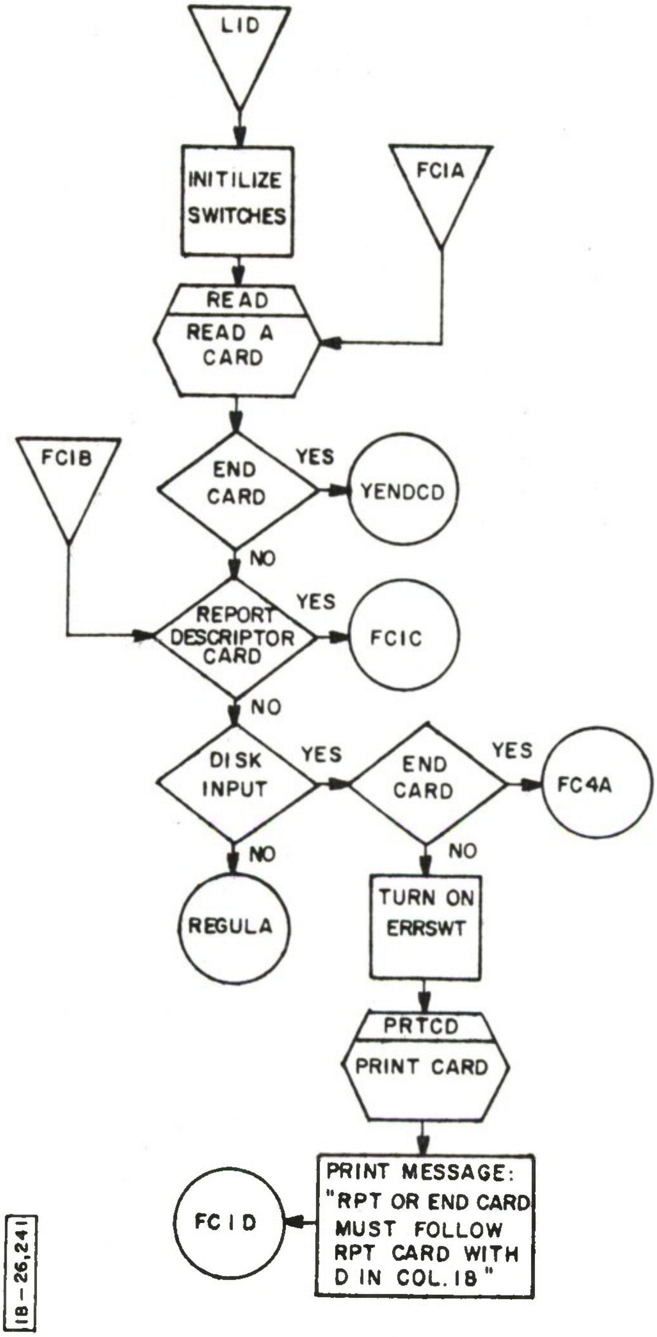
c. Make an entry in the Record Descriptor Table.

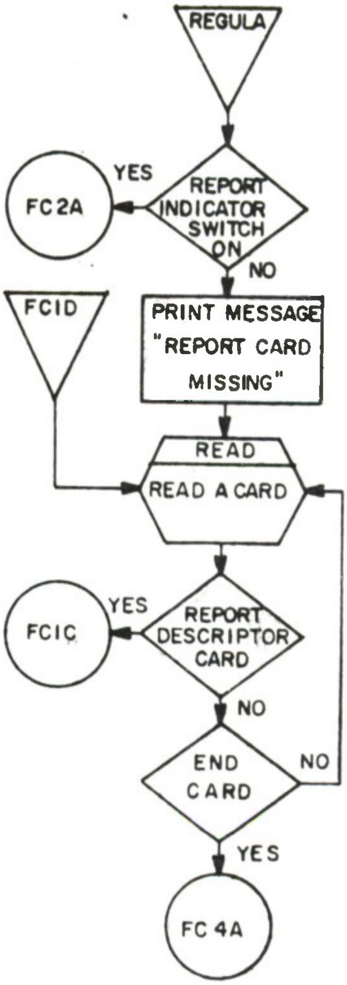
END Card (or next Report Descriptor Card)-- When an END Card is read, LID will:

a. Construct an Output Entry Descriptor Table (OEDT) to describe the format of each record as it will later appear on the Serial File Tape.

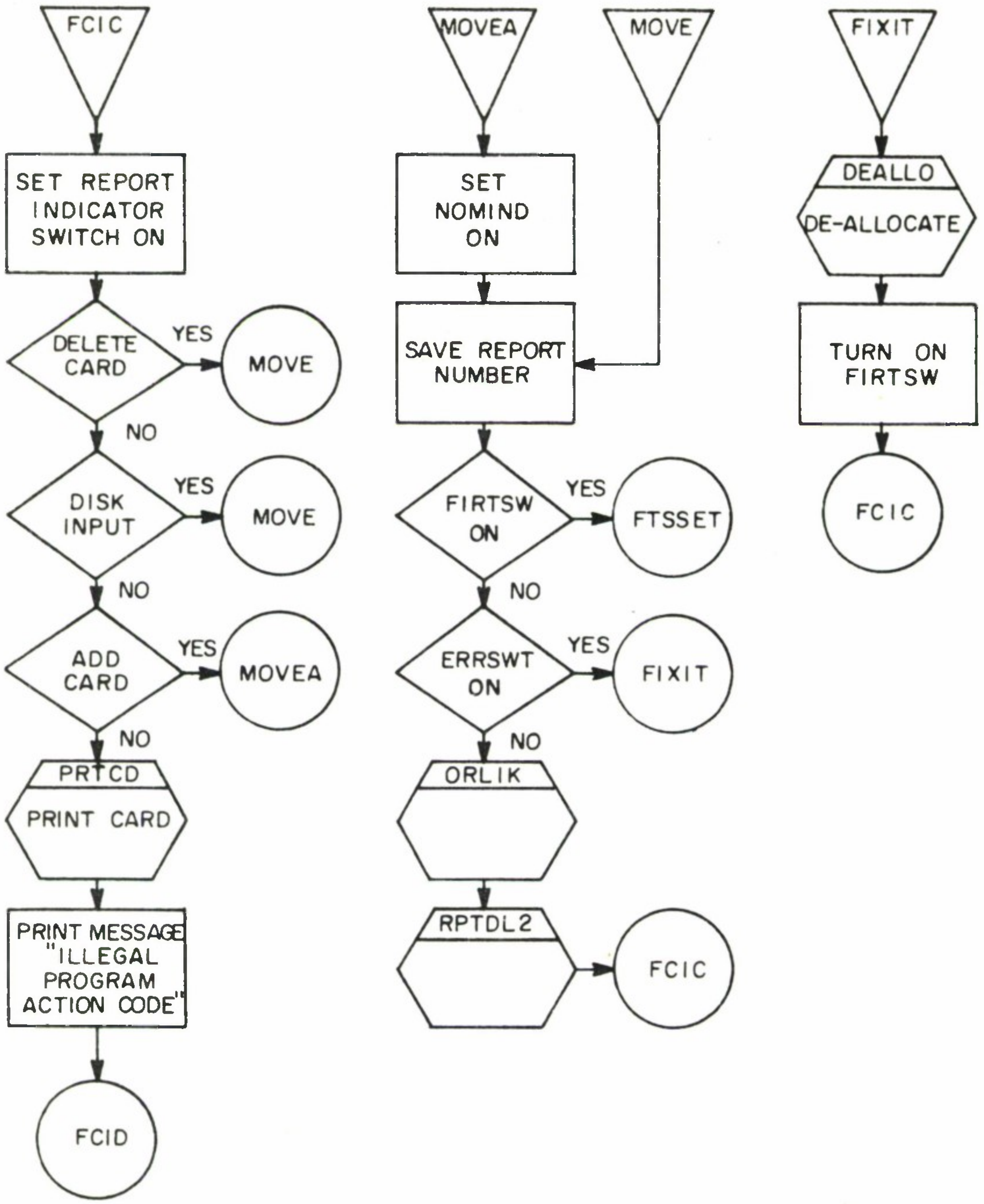
b. Modify each Attribute Descriptor Table entry to include data identifying the attribute's assigned position in the Serial File Tape records.

c. The modified Report Descriptor Table (and Record Descriptor Table if input is not disk) is written onto the system disk.

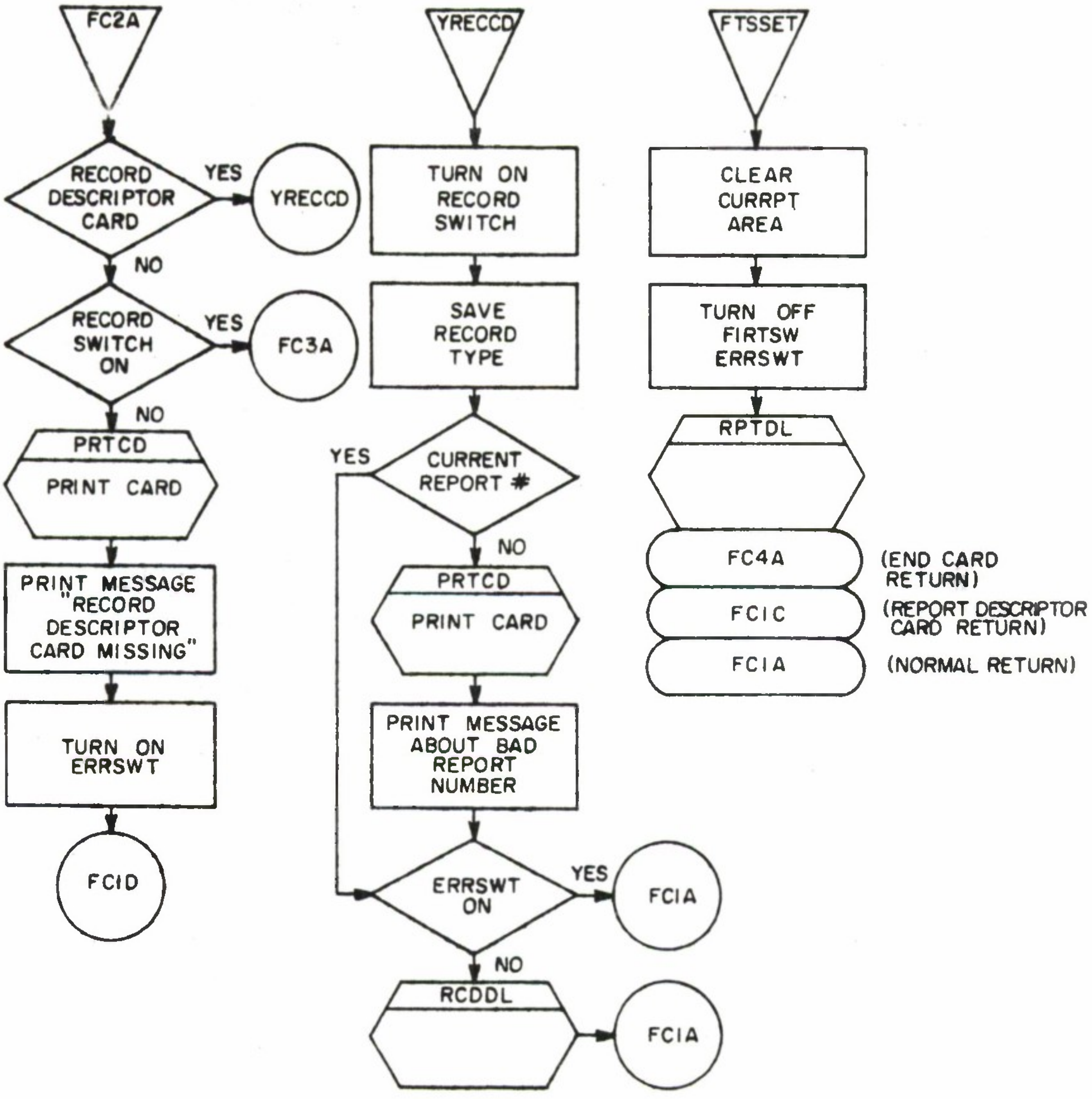

LID
INITILIZE SWITCHES
FCIA
READ
READ A CARD
FCIB
END CARD
YES
YENDCD
NO
REPORT DESCRIPTOR CARD
YES
FCIC
NO
DISK INPUT
YES
END CARD
YES
FC4A
NO
REGULA
NO
TURN ON ERRSWT
PRTCD
PRINT CARD
PRINT MESSAGE: "RPT OR END CARD MUST FOLLOW RPT CARD WITH D IN COL. 18"
FC I D


REGULA
YES
FC2A
REPORT INDICATOR SWITCH ON
NO
FCID
PRINT MESSAGE "REPORT CARD MISSING"
READ
READ A CARD
YES
FCIC
REPORT DESCRIPTOR CARD
NO
END CARD
NO
YES
FC 4A

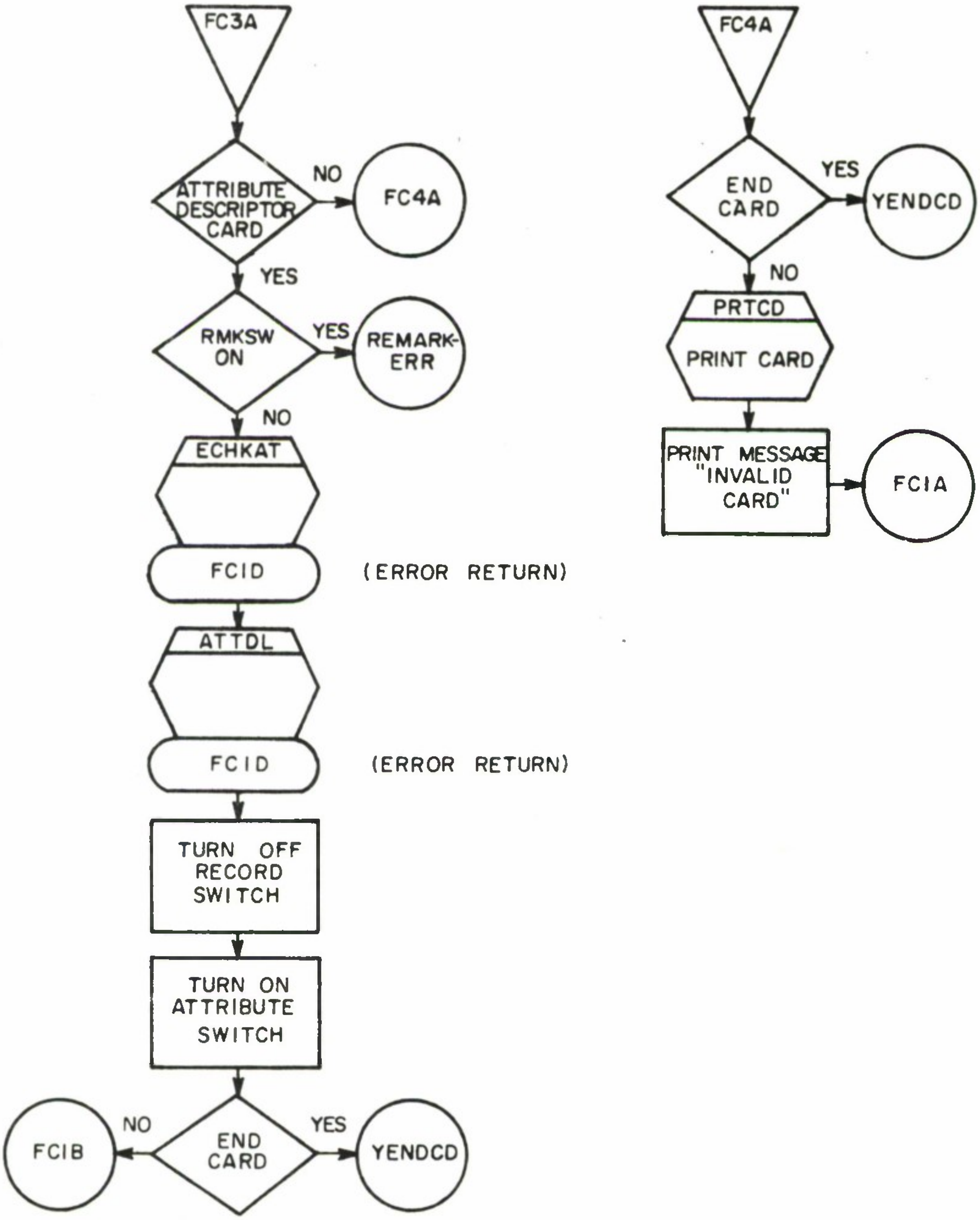
18-26,241


FCIC
SET REPORT INDICATOR SWITCH ON
DELETE CARD
YES
MOVE
NO
DISK INPUT
YES
MOVE
NO
ADD CARD
YES
MOVEA
NO
PRTCD
PRINT CARD
PRINT MESSAGE "ILLEGAL PROGRAM ACTION CODE"
FCID
MOVEA
MOVE
SET NOMIND ON
SAVE REPORT NUMBER
FIRTSW ON
YES
FTSSET
NO
ERRSWT ON
YES
FIXIT
NO
ORLIK
RPTDL2
FCIC
FIXIT
DEALLO
DE-ALLOCATE
TURN ON FIRTSW
FCIC

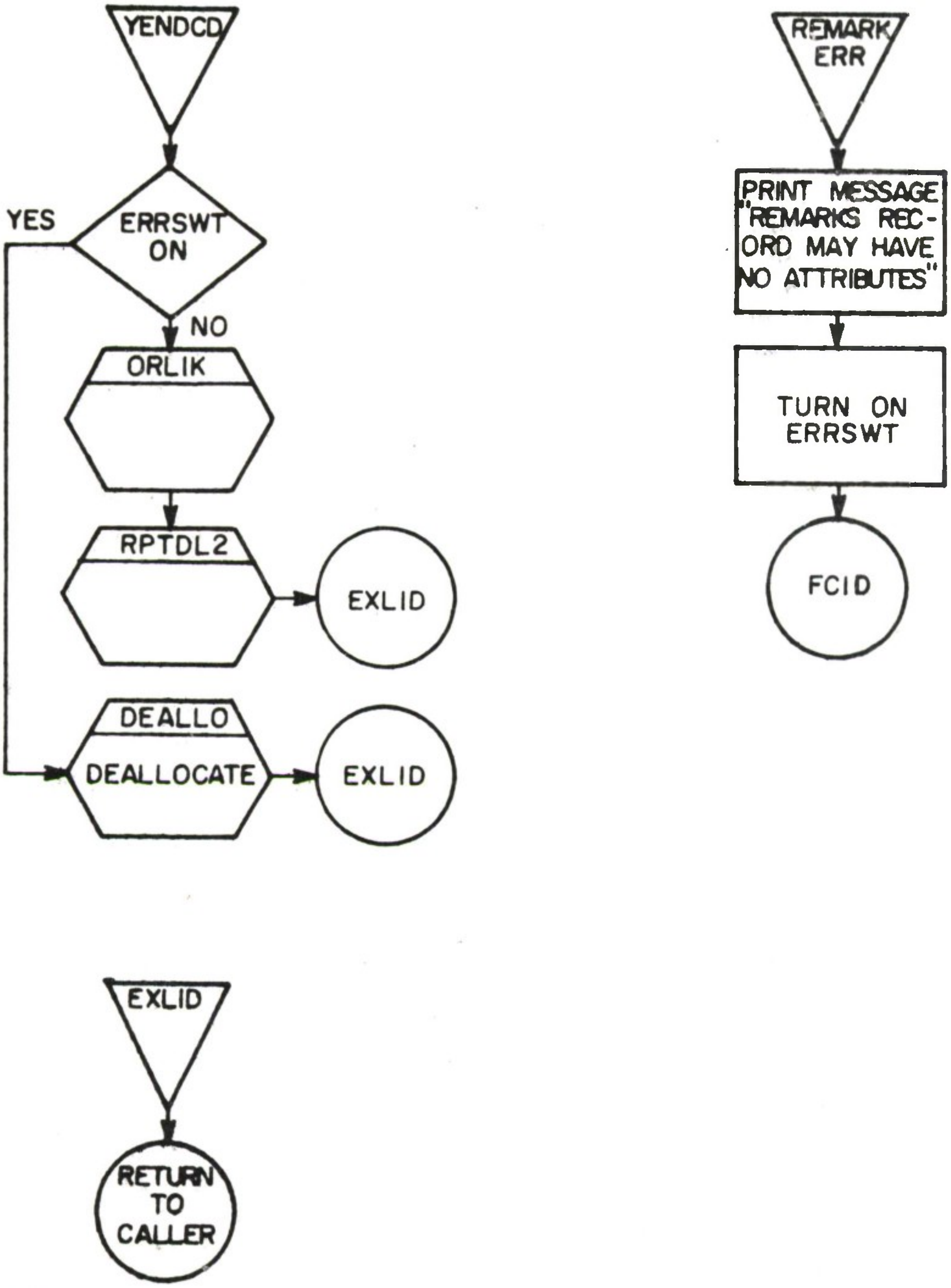
18-26,229


FC2A
RECORD DESCRIPTOR CARD
YES
YRECCD
NO
RECORD SWITCH ON
YES
FC3A
NO
PRTCD
PRINT CARD
PRINT MESSAGE "RECORD DESCRIPTOR CARD MISSING"
TURN ON ERRSWT
FCID
YRECCD
TURN ON RECORD SWITCH
SAVE RECORD TYPE
YES
CURRENT REPORT #
NO
PRTCD
PRINT CARD
PRINT MESSAGE ABOUT BAD REPORT NUMBER
ERRSWT ON
YES
FCIA
NO
RCDDL
FCIA
FTSSET
CLEAR CURRPT AREA
TURN OFF FIRTSW ERRSWT
RPTDL
FC4A
(END CARD RETURN)
FCIC
(REPORT DESCRIPTOR CARD RETURN)
FCIA
(NORMAL RETURN)

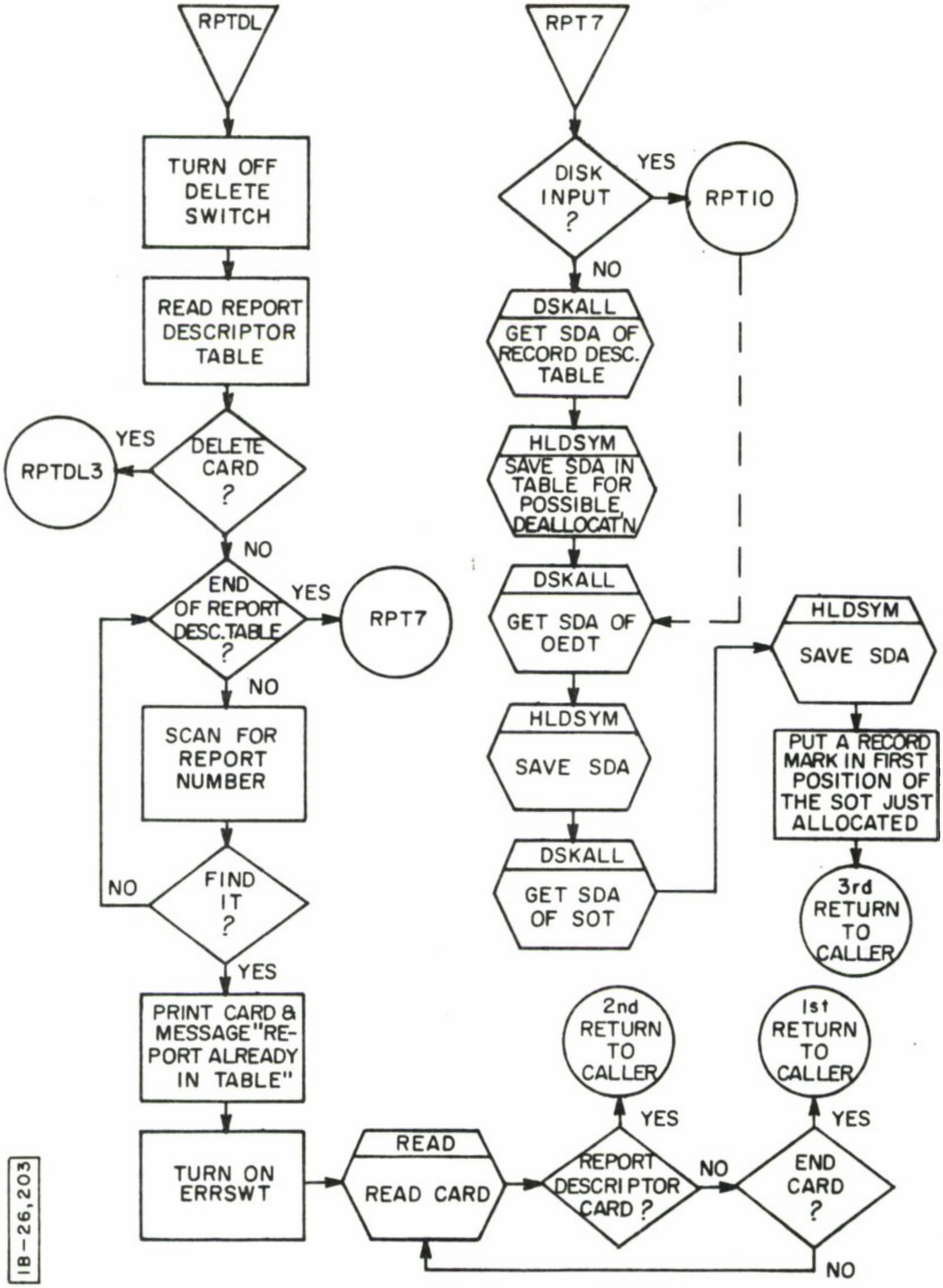
IB-26,230

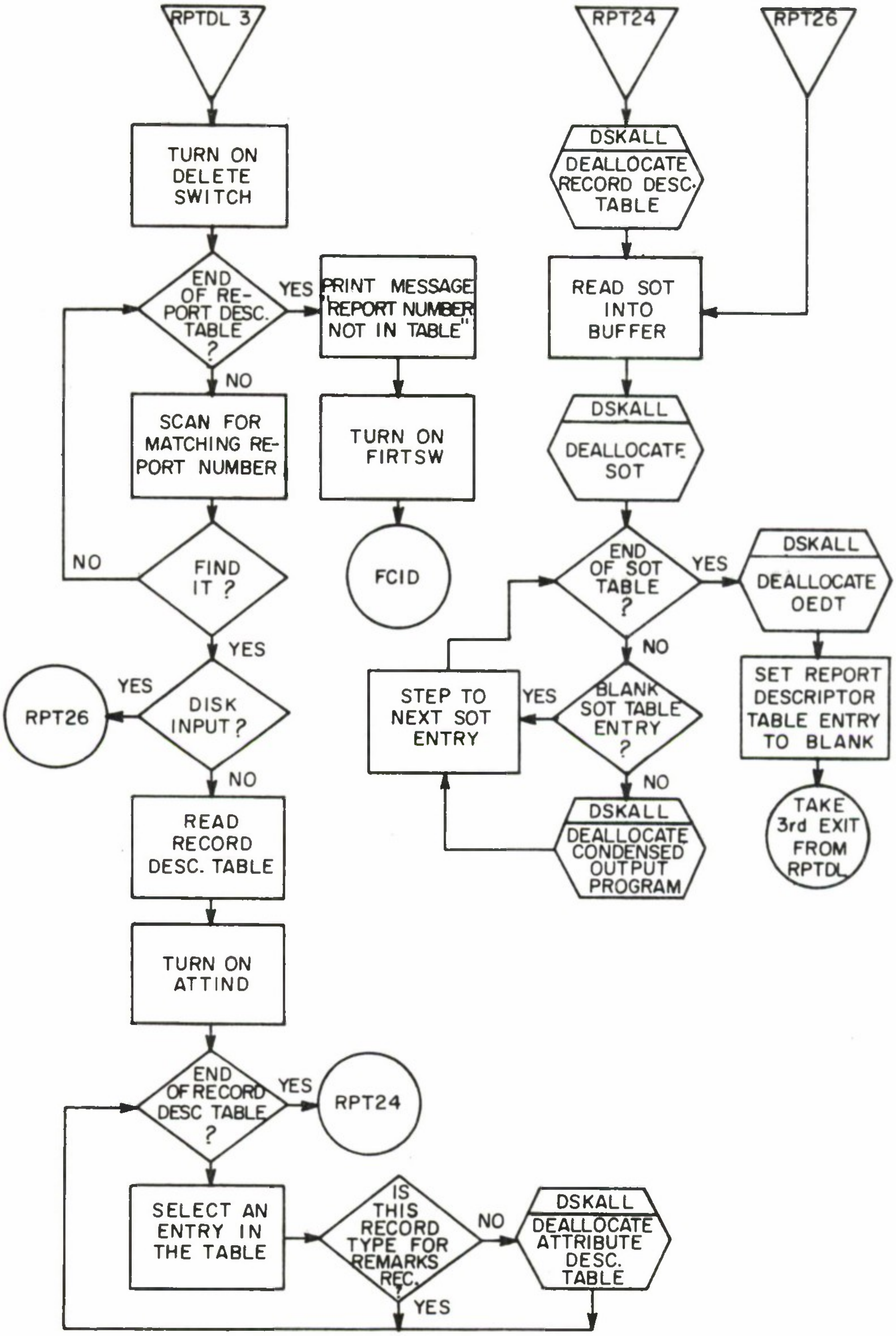

FC3A
ATTRIBUTE DESCRIPTOR CARD
NO
FC4A
YES
RMKSW ON
YES
REMARK-ERR
NO
ECHKAT
FCID
(ERROR RETURN)
ATTDL
FCID
(ERROR RETURN)
TURN OFF RECORD SWITCH
TURN ON ATTRIBUTE SWITCH
NO
FCIB
END CARD
YES
YENDCD
FC4A
END CARD
YES
YENDCD
NO
PRTCD
PRINT CARD
PRINT MESSAGE "INVALID CARD"
FCIA

1B-26,209

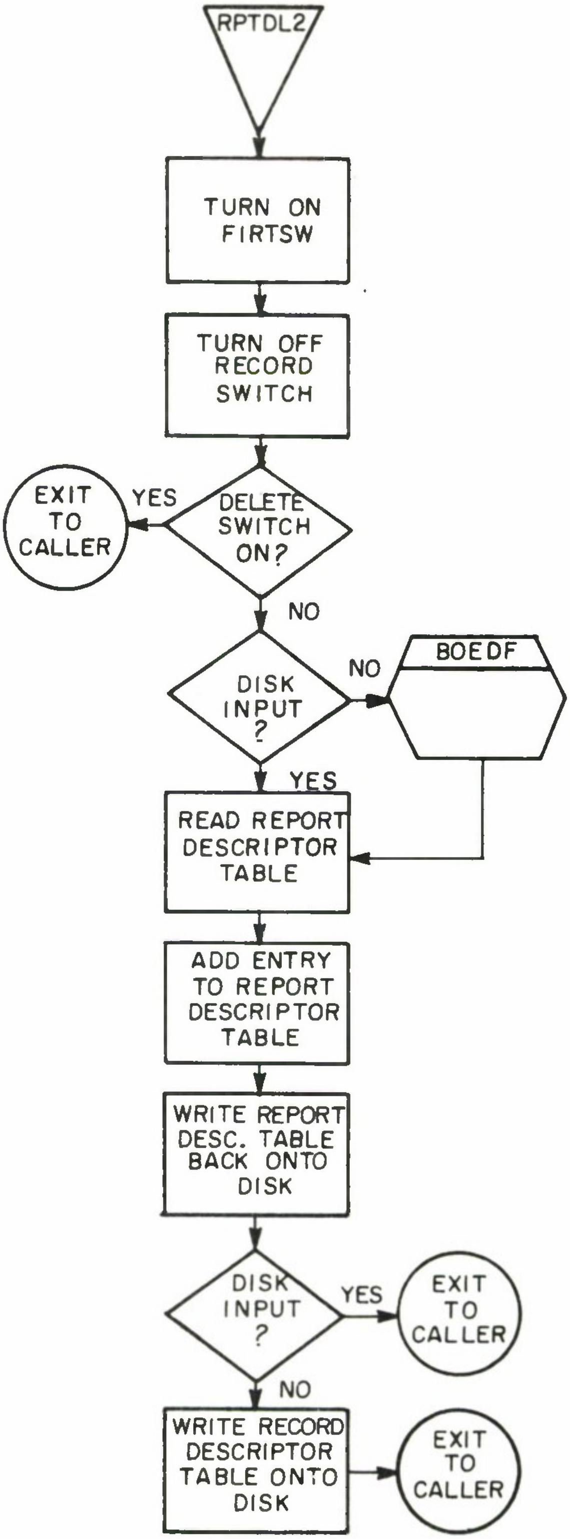

YENDCD
ERRSWT ON
YES
NO
ORLIK
RPTDL2
EXLID
DEALLO
DEALLOCATE
EXLID
REMARK ERR
PRINT MESSAGE "REMARKS RECORD MAY HAVE NO ATTRIBUTES"
TURN ON ERRSWT
FCID
EXLID
RETURN TO CALLER

18-26,208

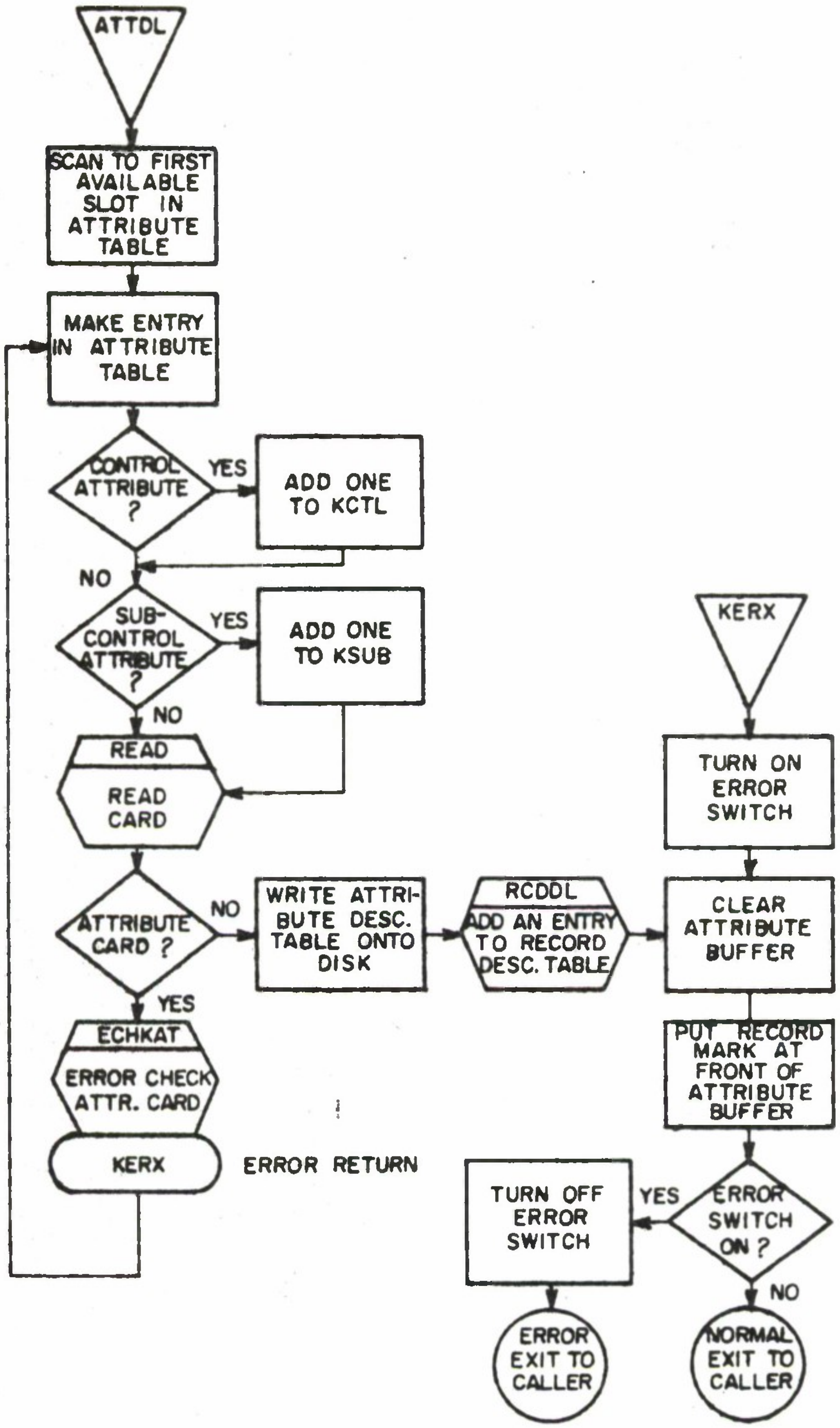

RPTDL
TURN OFF DELETE SWITCH
READ REPORT DESCRIPTOR TABLE
DELETE CARD ?
YES
RPTDL3
NO
END OF REPORT DESC. TABLE ?
YES
RPT7
NO
SCAN FOR REPORT NUMBER
FIND IT ?
NO
YES
PRINT CARD & MESSAGE "REPORT ALREADY IN TABLE"
TURN ON ERRSWT
READ
READ CARD
REPORT DESCRIPTOR CARD ?
YES
2nd RETURN TO CALLER
NO
END CARD ?
YES
1st RETURN TO CALLER
NO
RPT7
DISK INPUT ?
YES
RPT10
NO
DSKALL
GET SDA OF RECORD DESC. TABLE
HLDSYM
SAVE SDA IN TABLE FOR POSSIBLE DEALLOCAT'N
DSKALL
GET SDA OF OEDT
HLDSYM
SAVE SDA
DSKALL
GET SDA OF SOT
HLDSYM
SAVE SDA
PUT A RECORD MARK IN FIRST POSITION OF THE SOT JUST ALLOCATED
3rd RETURN TO CALLER
IB-26,203


RPTDL 3
TURN ON DELETE SWITCH
END OF REPORT DESC. TABLE ?
YES
PRINT MESSAGE "REPORT NUMBER NOT IN TABLE"
TURN ON FIRTSW
FCID
NO
SCAN FOR MATCHING REPORT NUMBER
NO
FIND IT ?
YES
DISK INPUT ?
YES
RPT26
NO
READ RECORD DESC. TABLE
TURN ON ATTIND
END OF RECORD DESC TABLE ?
YES
RPT24
SELECT AN ENTRY IN THE TABLE
IS THIS RECORD TYPE FOR REMARKS REC. ?
NO
YES
DSKALL
DEALLOCATE ATTRIBUTE DESC. TABLE
RPT24
RPT26
DSKALL
DEALLOCATE RECORD DESC. TABLE
READ SOT INTO BUFFER
DSKALL
DEALLOCATE SOT
END OF SOT TABLE ?
YES
DSKALL
DEALLOCATE OEDT
SET REPORT DESCRIPTOR TABLE ENTRY TO BLANK
TAKE 3rd EXIT FROM RPTDL
NO
BLANK SOT TABLE ENTRY ?
YES
STEP TO NEXT SOT ENTRY
NO
DSKALL
DEALLOCATE CONDENSED OUTPUT PROGRAM

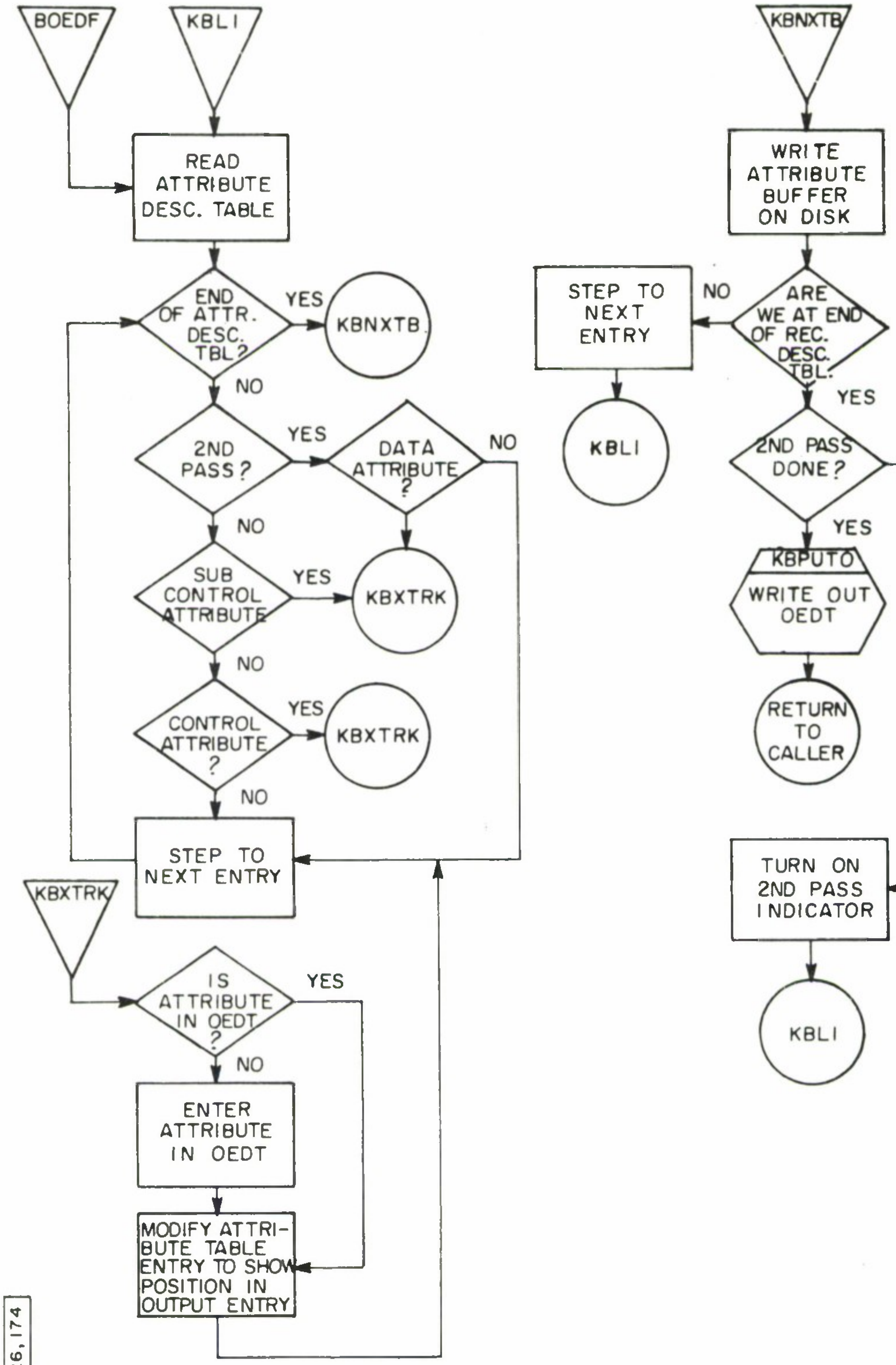
IB-26,204


RPTDL2
TURN ON FIRTSW
TURN OFF RECORD SWITCH
DELETE SWITCH ON?
YES
EXIT TO CALLER
NO
DISK INPUT ?
NO
BOEDF
YES
READ REPORT DESCRIPTOR TABLE
ADD ENTRY TO REPORT DESCRIPTOR TABLE
WRITE REPORT DESC. TABLE BACK ONTO DISK
DISK INPUT ?
YES
EXIT TO CALLER
NO
WRITE RECORD DESCRIPTOR TABLE ONTO DISK
EXIT TO CALLER

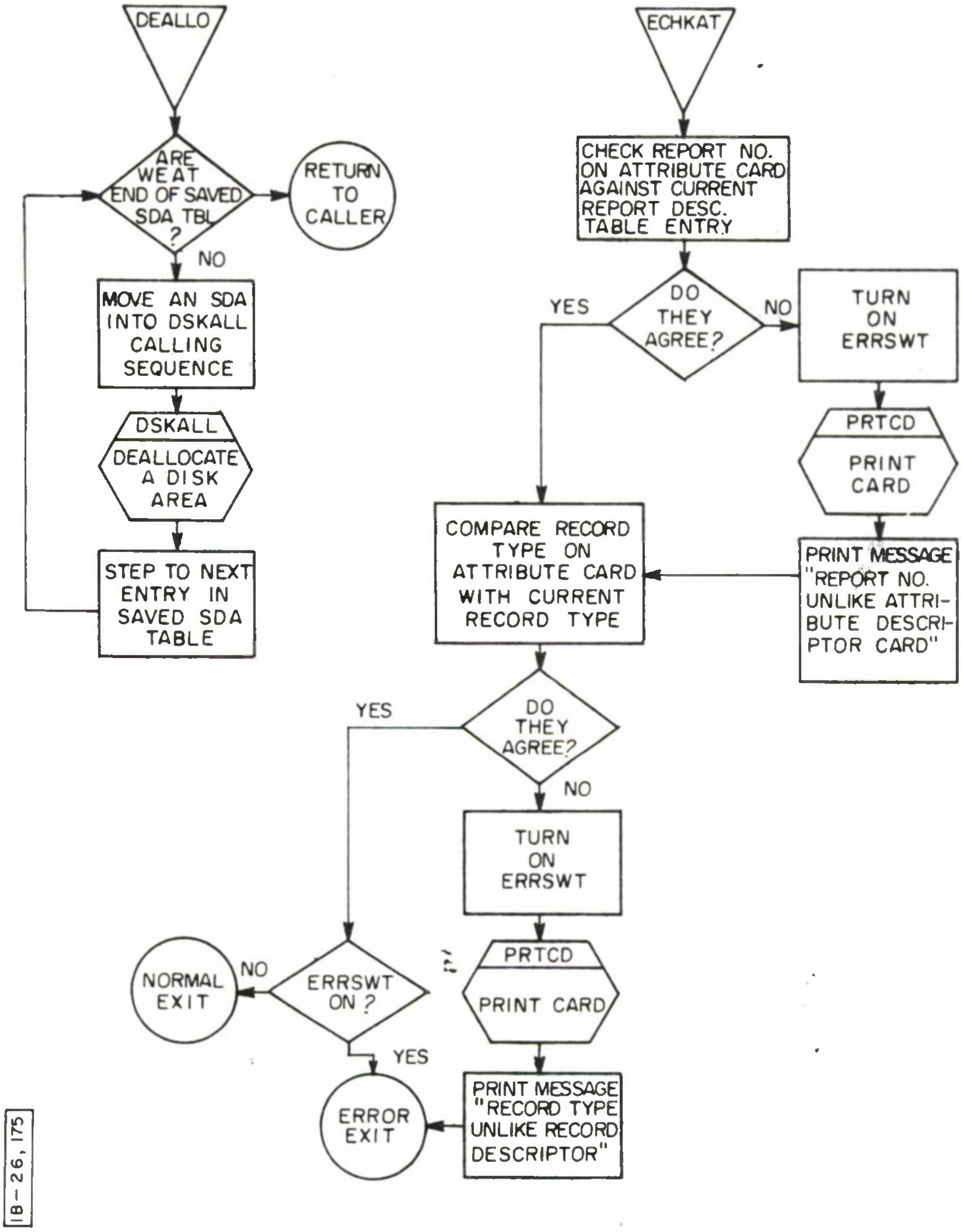
18-26,206


ATTDL
SCAN TO FIRST AVAILABLE SLOT IN ATTRIBUTE TABLE
MAKE ENTRY IN ATTRIBUTE TABLE
CONTROL ATTRIBUTE ?
YES
ADD ONE TO KCTL
NO
SUB-CONTROL ATTRIBUTE ?
YES
ADD ONE TO KSUB
NO
READ
READ CARD
ATTRIBUTE CARD ?
NO
WRITE ATTRIBUTE DESC. TABLE ONTO DISK
RCDDL
ADD AN ENTRY TO RECORD DESC. TABLE
YES
ECHKAT
ERROR CHECK ATTR. CARD
KERX
ERROR RETURN
KERX
TURN ON ERROR SWITCH
CLEAR ATTRIBUTE BUFFER
PUT RECORD MARK AT FRONT OF ATTRIBUTE BUFFER
ERROR SWITCH ON ?
YES
TURN OFF ERROR SWITCH
ERROR EXIT TO CALLER
NO
NORMAL EXIT TO CALLER

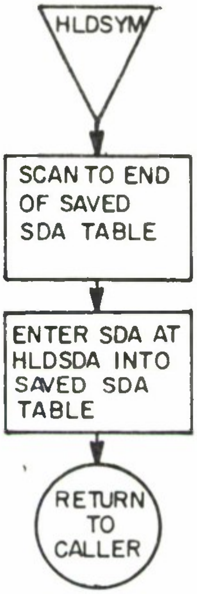
18-26,207

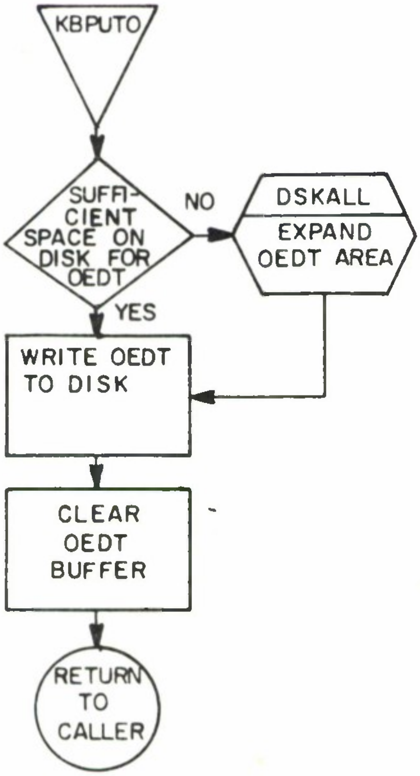

BOEDF
KBLI
READ ATTRIBUTE DESC. TABLE
END OF ATTR. DESC. TBL?
YES
KBNXTB
NO
2ND PASS?
YES
DATA ATTRIBUTE ?
NO
NO
SUB CONTROL ATTRIBUTE
YES
KBXTRK
NO
CONTROL ATTRIBUTE ?
YES
KBXTRK
NO
STEP TO NEXT ENTRY
KBXTRK
IS ATTRIBUTE IN OEDT ?
YES
NO
ENTER ATTRIBUTE IN OEDT
MODIFY ATTRIBUTE TABLE ENTRY TO SHOW POSITION IN OUTPUT ENTRY
KBNXTB
WRITE ATTRIBUTE BUFFER ON DISK
ARE WE AT END OF REC. DESC. TBL.
NO
STEP TO NEXT ENTRY
KBLI
YES
2ND PASS DONE?
NO
YES
KBPUTO
WRITE OUT OEDT
RETURN TO CALLER
TURN ON 2ND PASS INDICATOR
KBLI

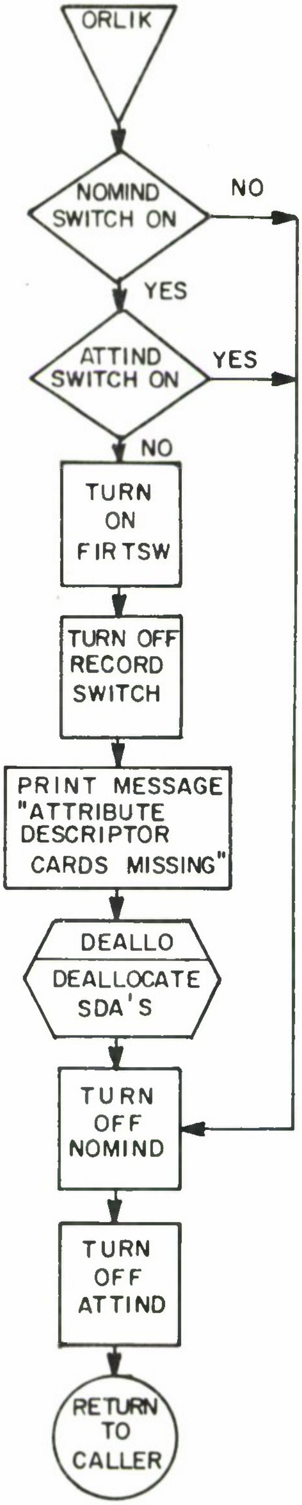
18-26,174


DEALLO
ARE WE AT END OF SAVED SDA TBL ?
RETURN TO CALLER
NO
MOVE AN SDA INTO DSKALL CALLING SEQUENCE
DSKALL
DEALLOCATE A DISK AREA
STEP TO NEXT ENTRY IN SAVED SDA TABLE
ECHKAT
CHECK REPORT NO. ON ATTRIBUTE CARD AGAINST CURRENT REPORT DESC. TABLE ENTRY
YES
DO THEY AGREE?
NO
TURN ON ERRSWT
PRTCD
PRINT CARD
PRINT MESSAGE "REPORT NO. UNLIKE ATTRIBUTE DESCRIPTOR CARD"
COMPARE RECORD TYPE ON ATTRIBUTE CARD WITH CURRENT RECORD TYPE
YES
DO THEY AGREE?
NO
TURN ON ERRSWT
PRTCD
PRINT CARD
PRINT MESSAGE "RECORD TYPE UNLIKE RECORD DESCRIPTOR"
NORMAL EXIT
NO
ERRSWT ON ?
YES
ERROR EXIT

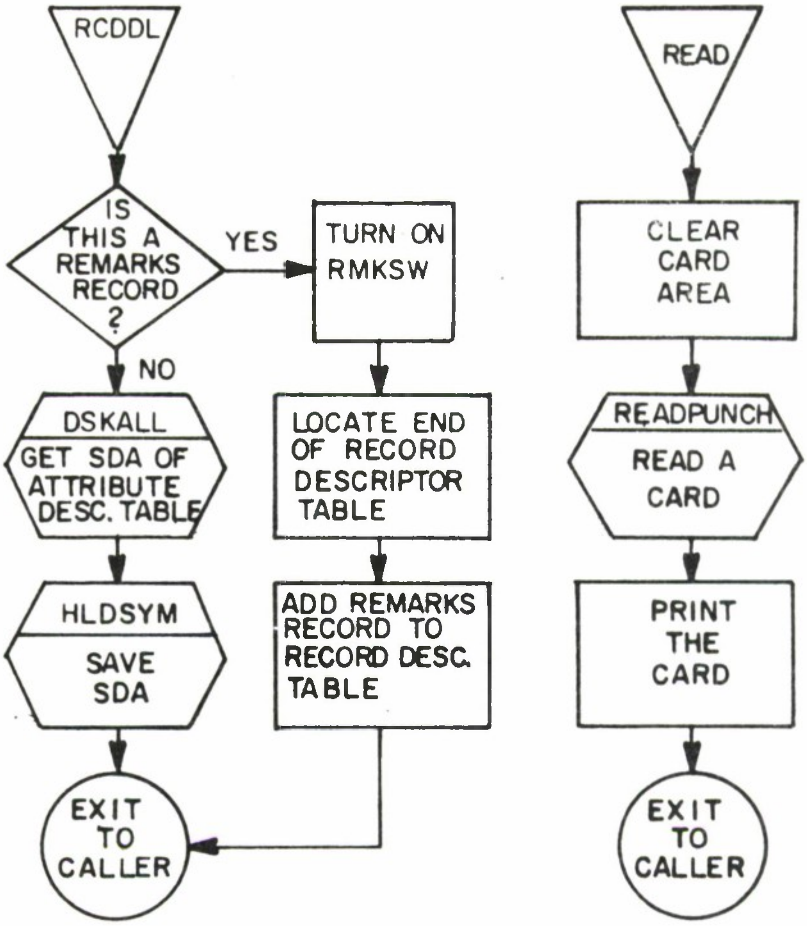
IB-26,175


HLDSYM
SCAN TO END OF SAVED SDA TABLE
ENTER SDA AT HLDSDA INTO SAVED SDA TABLE
RETURN TO CALLER


KBPUTO
SUFFI-CIENT SPACE ON DISK FOR OEDT
NO
DSKALL
EXPAND OEDT AREA
YES
WRITE OEDT TO DISK
CLEAR OEDT BUFFER
RETURN TO CALLER


ORLIK
NOMIND SWITCH ON
NO
YES
ATTIND SWITCH ON
YES
NO
TURN ON FIRTSW
TURN OFF RECORD SWITCH
PRINT MESSAGE "ATTRIBUTE DESCRIPTOR CARDS MISSING"
DEALLO
DEALLOCATE SDA'S
TURN OFF NOMIND
TURN OFF ATTIND
RETURN TO CALLER

IB-26,242


RCDDL
IS THIS A REMARKS RECORD ?
YES
NO
TURN ON RMKSW
DSKALL
GET SDA OF ATTRIBUTE DESC. TABLE
LOCATE END OF RECORD DESCRIPTOR TABLE
HLDSYM
SAVE SDA
ADD REMARKS RECORD TO RECORD DESC. TABLE
EXIT TO CALLER
READ
CLEAR CARD AREA
READPUNCH
READ A CARD
PRINT THE CARD
EXIT TO CALLER

IB-26,243

LEGVAL

Purpose

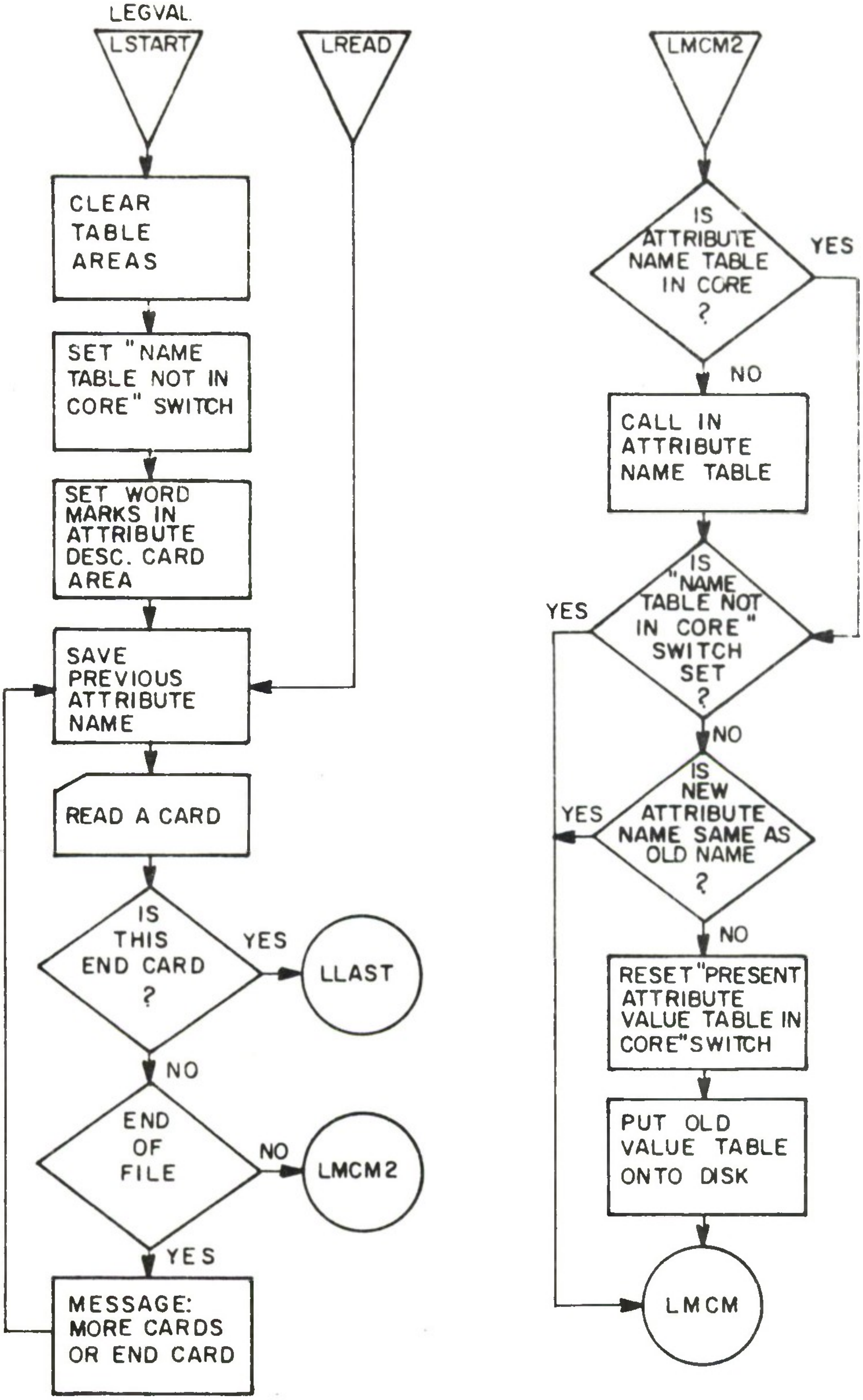
To generate a disk table containing all allowable values for one or more attributes. This table is to be used during the file generation process which creates a Serial File Tape from source input data.

Calling Sequence

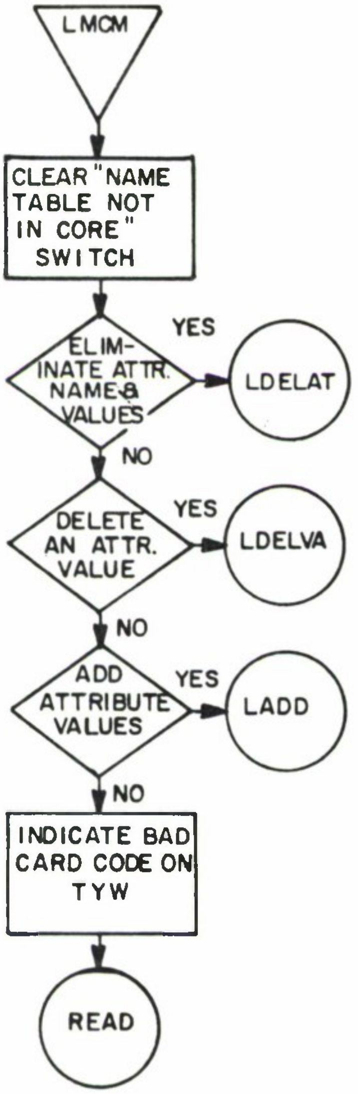
Enter LEGVAL on the console typewriter.

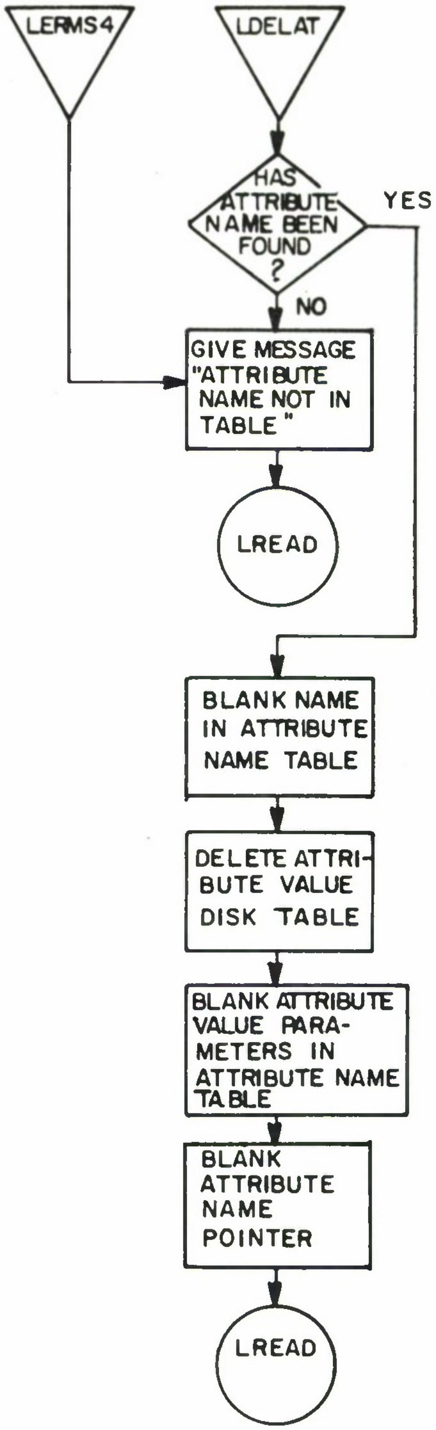
Description

Each input card to LEGVAL contains the name of an attribute and one of the allowable values for that attribute. LEGVAL checks to see if that attribute name has been previously encountered. If not, the name is added to a list of attributes which are to be legality checked. After the name has been added to the name list, an SDA is allocated to be used for storing the list of legal values pertaining to the attribute. When all allowable values for a given attribute have been read, the value table is written on the system disk. When an END card is read, the name list is written on the system disk.

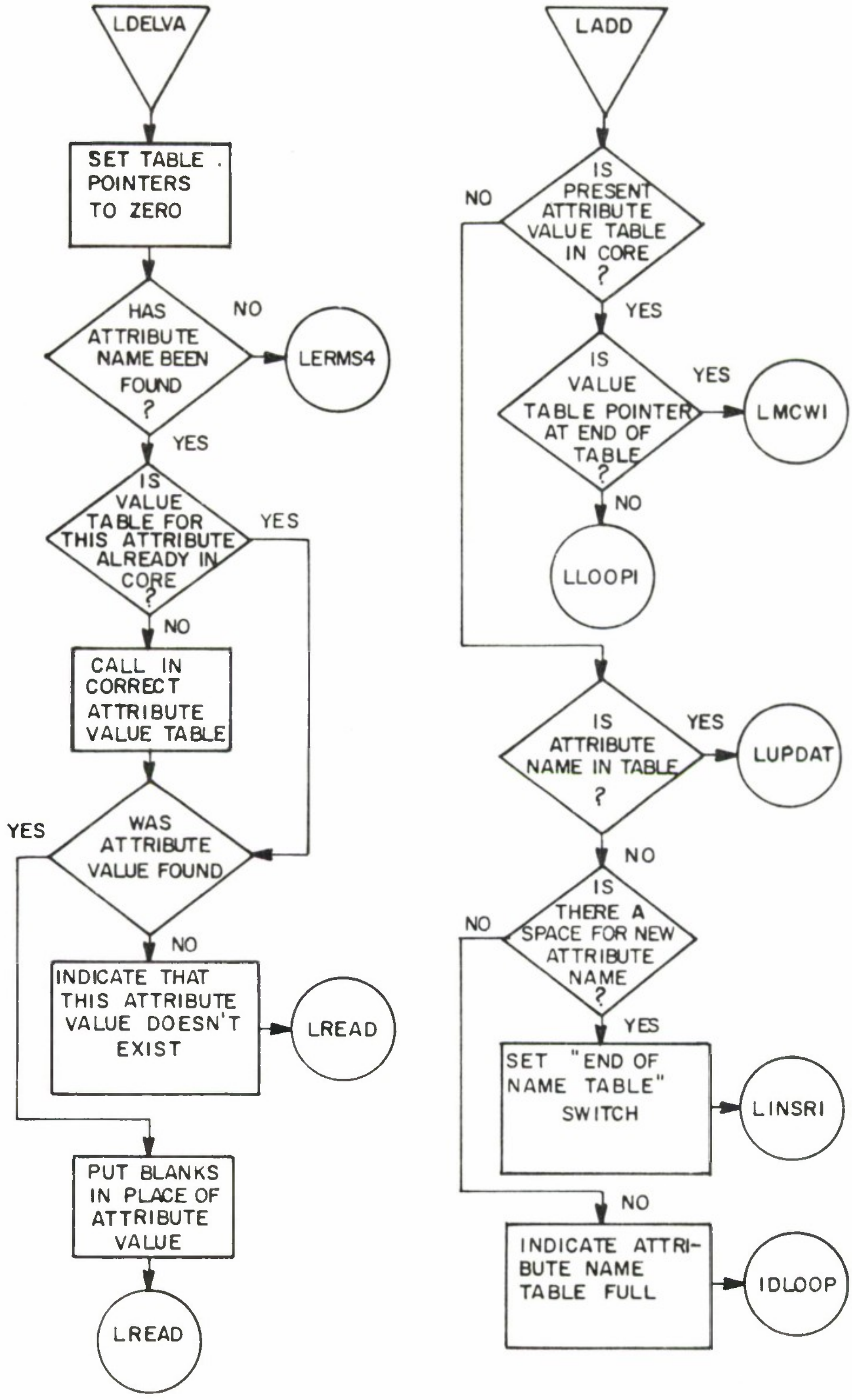

LEGVAL.
LSTART
LREAD
LMCM2
CLEAR TABLE AREAS
SET "NAME TABLE NOT IN CORE" SWITCH
SET WORD MARKS IN ATTRIBUTE DESC. CARD AREA
SAVE PREVIOUS ATTRIBUTE NAME
READ A CARD
IS THIS END CARD ?
YES
LLAST
NO
END OF FILE
NO
LMCM2
YES
MESSAGE: MORE CARDS OR END CARD
IS ATTRIBUTE NAME TABLE IN CORE ?
YES
NO
CALL IN ATTRIBUTE NAME TABLE
IS "NAME TABLE NOT IN CORE" SWITCH SET ?
YES
NO
IS NEW ATTRIBUTE NAME SAME AS OLD NAME ?
YES
NO
RESET "PRESENT ATTRIBUTE VALUE TABLE IN CORE" SWITCH
PUT OLD VALUE TABLE ONTO DISK
LMCM

IB-26,244

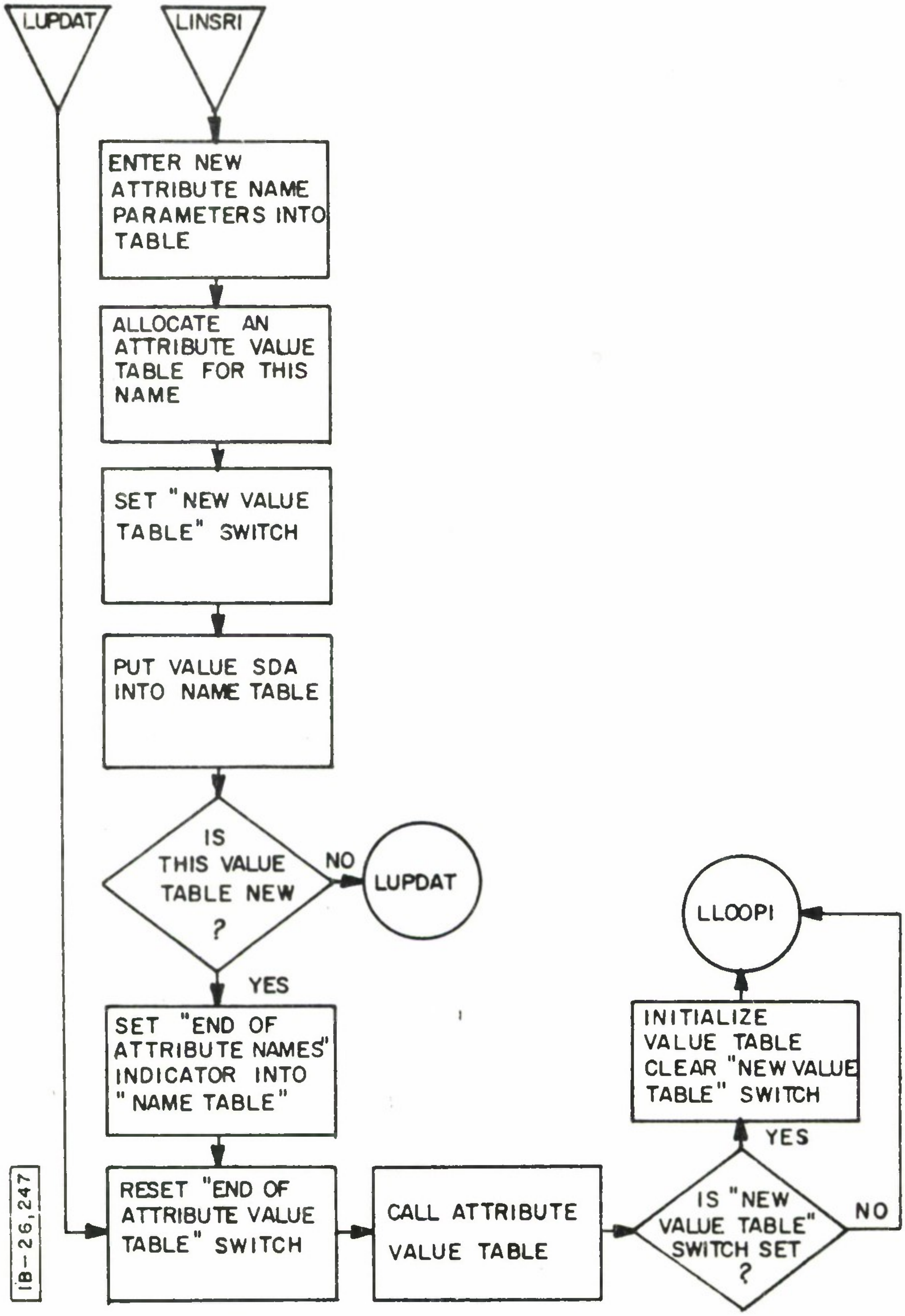

LMCM
CLEAR "NAME TABLE NOT IN CORE" SWITCH
ELIMINATE ATTR. NAME & VALUES
YES
LDELAT
NO
DELETE AN ATTR. VALUE
YES
LDELVA
NO
ADD ATTRIBUTE VALUES
YES
LADD
NO
INDICATE BAD CARD CODE ON TYW
READ

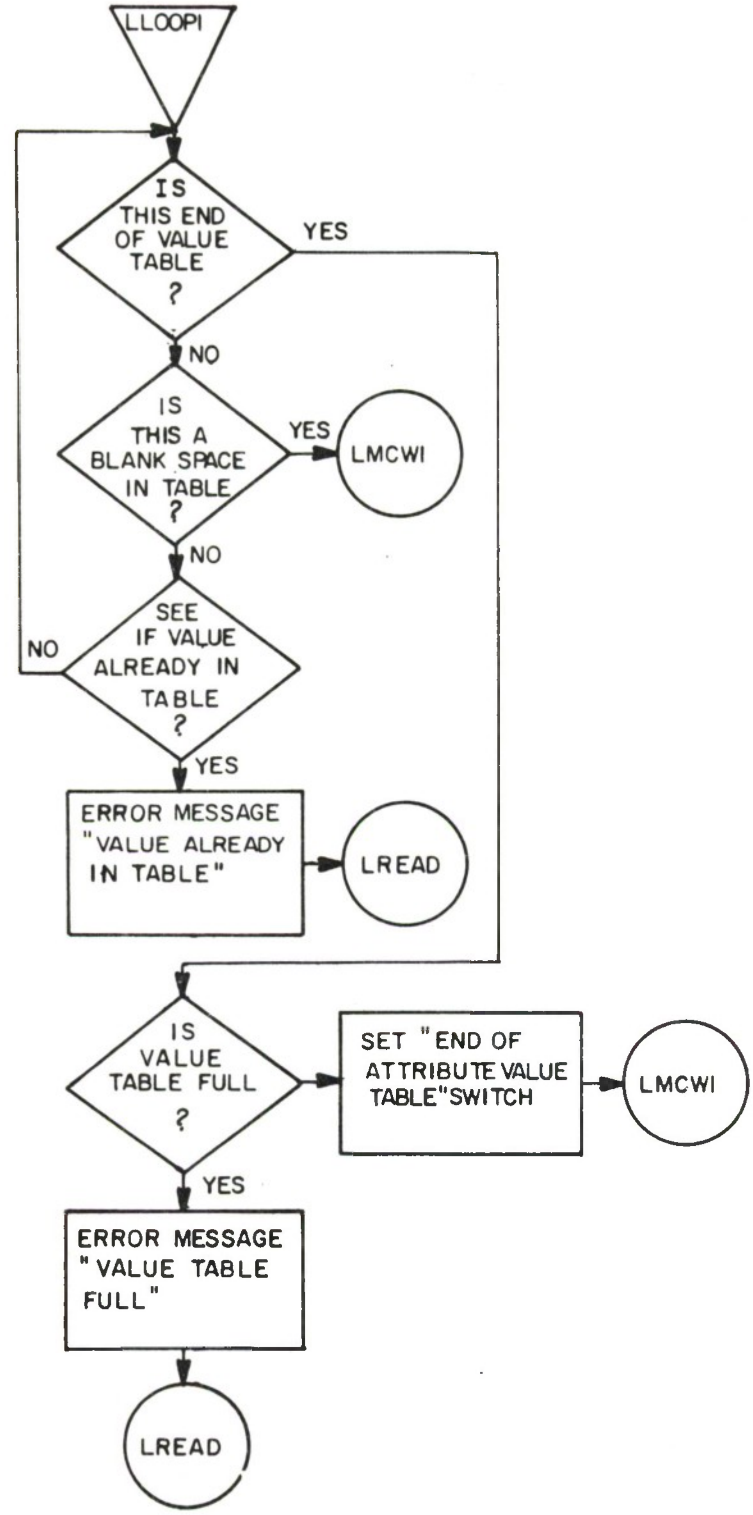

LERMS4
LDELAT
HAS ATTRIBUTE NAME BEEN FOUND ?
YES
NO
GIVE MESSAGE "ATTRIBUTE NAME NOT IN TABLE"
LREAD
BLANK NAME IN ATTRIBUTE NAME TABLE
DELETE ATTRIBUTE VALUE DISK TABLE
BLANK ATTRIBUTE VALUE PARAMETERS IN ATTRIBUTE NAME TABLE
BLANK ATTRIBUTE NAME POINTER
LREAD

18-26,245

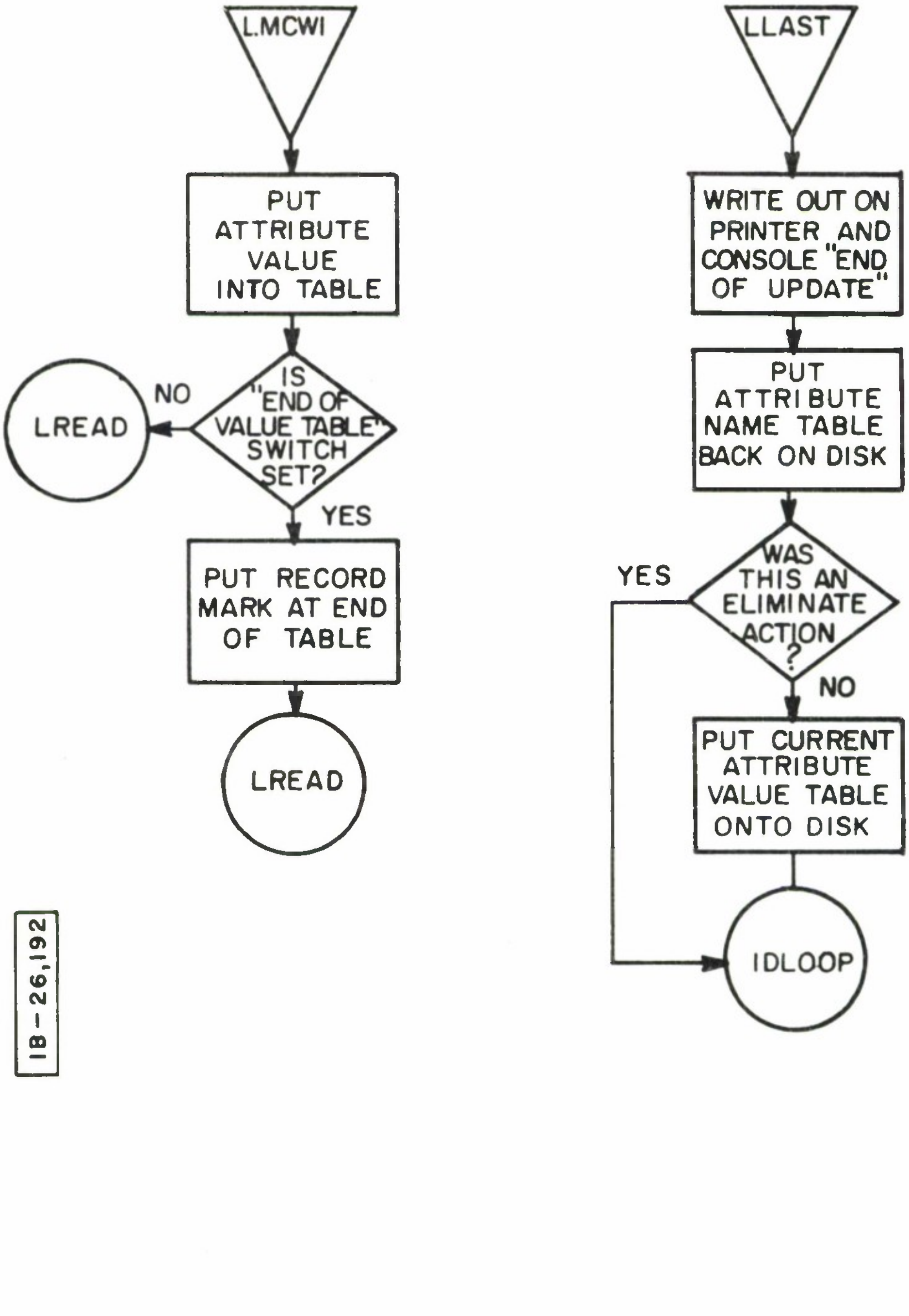

LDELVA
SET TABLE POINTERS TO ZERO
HAS ATTRIBUTE NAME BEEN FOUND ?
NO
LERMS4
YES
IS VALUE TABLE FOR THIS ATTRIBUTE ALREADY IN CORE ?
YES
NO
CALL IN CORRECT ATTRIBUTE VALUE TABLE
WAS ATTRIBUTE VALUE FOUND
YES
NO
INDICATE THAT THIS ATTRIBUTE VALUE DOESN'T EXIST
LREAD
PUT BLANKS IN PLACE OF ATTRIBUTE VALUE
LREAD
LADD
IS PRESENT ATTRIBUTE VALUE TABLE IN CORE ?
NO
YES
IS VALUE TABLE POINTER AT END OF TABLE ?
YES
LMCWI
NO
LLOOPI
IS ATTRIBUTE NAME IN TABLE ?
YES
LUPDAT
NO
IS THERE A SPACE FOR NEW ATTRIBUTE NAME ?
NO
YES
SET "END OF NAME TABLE" SWITCH
LINSRI
NO
INDICATE ATTRIBUTE NAME TABLE FULL
IDLOOP

1B-26,246


LUPDAT
LINSRI
ENTER NEW ATTRIBUTE NAME PARAMETERS INTO TABLE
ALLOCATE AN ATTRIBUTE VALUE TABLE FOR THIS NAME
SET "NEW VALUE TABLE" SWITCH
PUT VALUE SDA INTO NAME TABLE
IS THIS VALUE TABLE NEW ?
NO
LUPDAT
YES
SET "END OF ATTRIBUTE NAMES" INDICATOR INTO "NAME TABLE"
IB-26,247
RESET "END OF ATTRIBUTE VALUE TABLE" SWITCH
CALL ATTRIBUTE VALUE TABLE
IS "NEW VALUE TABLE" SWITCH SET ?
NO
YES
INITIALIZE VALUE TABLE CLEAR "NEW VALUE TABLE" SWITCH
LLOOPI


LLOOPI
IS THIS END OF VALUE TABLE ?
YES
NO
IS THIS A BLANK SPACE IN TABLE ?
YES
LMCWI
NO
SEE IF VALUE ALREADY IN TABLE ?
NO
YES
ERROR MESSAGE "VALUE ALREADY IN TABLE"
LREAD
IS VALUE TABLE FULL ?
SET "END OF ATTRIBUTE VALUE TABLE" SWITCH
LMCWI
YES
ERROR MESSAGE "VALUE TABLE FULL"
LREAD

1B-26,248


L.MCWI
PUT ATTRIBUTE VALUE INTO TABLE
IS "END OF VALUE TABLE" SWITCH SET?
NO
LREAD
YES
PUT RECORD MARK AT END OF TABLE
LREAD
LLAST
WRITE OUT ON PRINTER AND CONSOLE "END OF UPDATE"
PUT ATTRIBUTE NAME TABLE BACK ON DISK
WAS THIS AN ELIMINATE ACTION ?
YES
NO
PUT CURRENT ATTRIBUTE VALUE TABLE ONTO DISK
IDLOOP
IB-26,192

SAVER

Purpose

A query language statement containing the GETTAC process director and report number(s) is stored on disk for utilization by the query portion of the report generation program.

Calling Sequence

From I/O typewriter, or cards with sense switch B on, starting in column 1.

```
    SAVERb̸GETTACb̸AAAA'''AA[BBB/BBB''']
  XXX+XXX+'''+XXX
```

AA''' = qualifying attributes

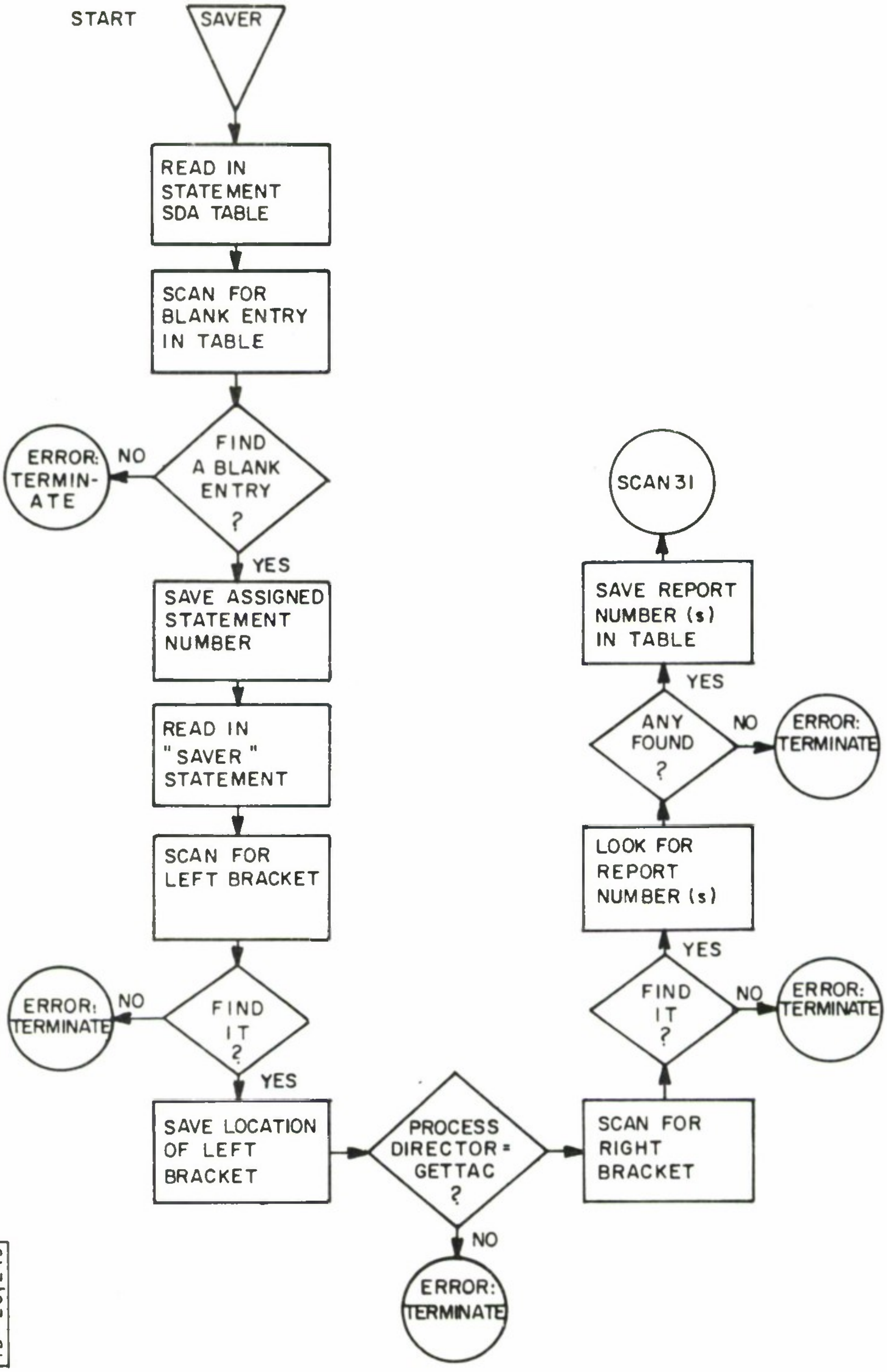
BB''' = output selector

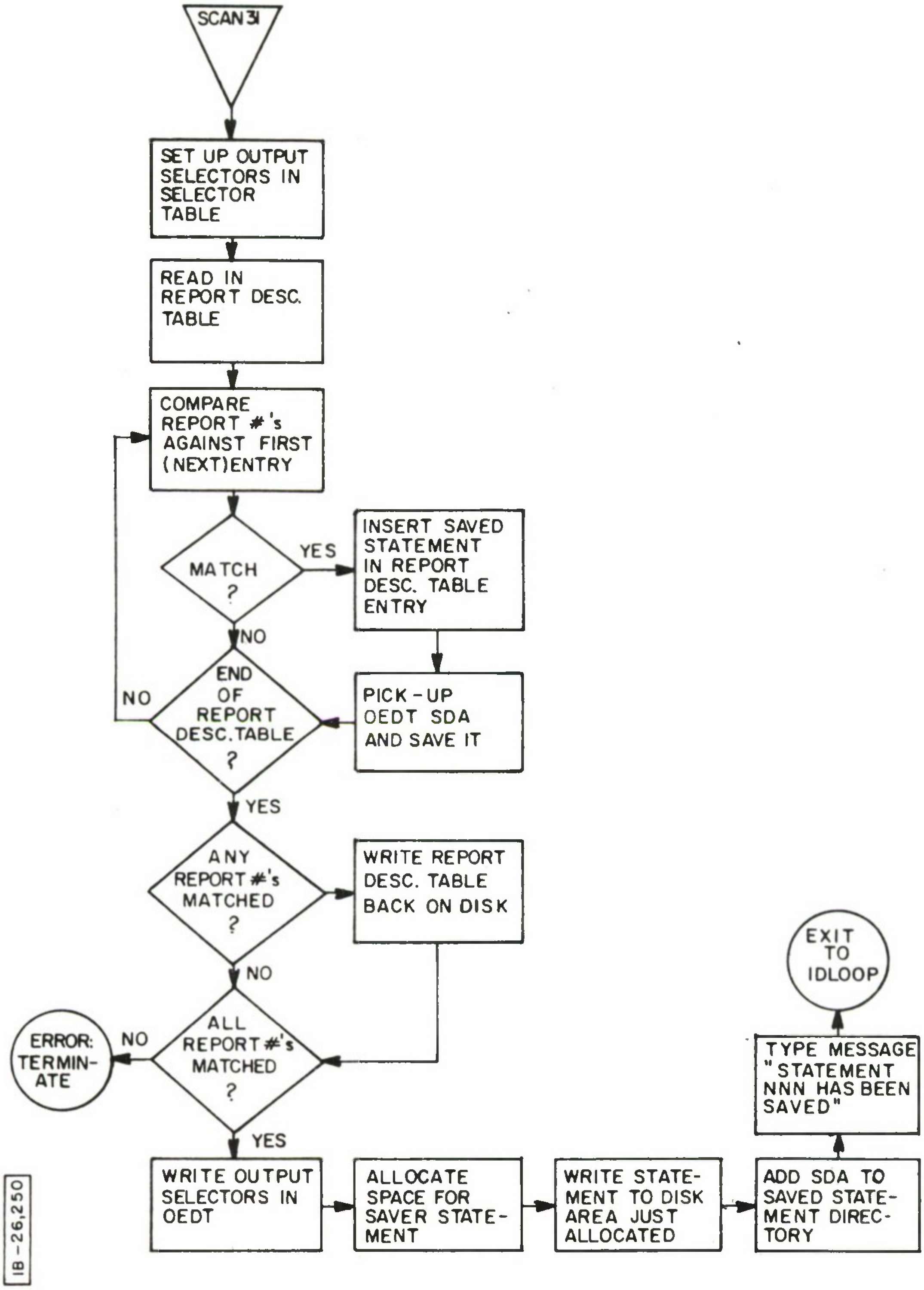
XXX = report numbers (maximum of 10)

Description

The saved statement SDA table is read from disk. The first blank entry is located and noted. The statement to be saved is accessed from the buffer SDA 'ØØØ3'. The statement is scanned for correctness, such as appearance of a left and right bracket and report numbers. The report numbers are stored in a small table area. The output selectors are stored in a table area 50 characters to each entry. The report descriptor file is now accessed. Report numbers are now matched from the report descriptor file with those of the saved statement. If a match is found the saved statement number is inserted in the appropriate slot of the report descriptor file. The output entry descriptor table SDA and number of records referenced by the report descriptor is saved in a table area. The new report descriptor is written back on system disk. The process is continued until all report numbers from the saved statement are exhausted.

The output selector table is now written on disk at the SDA's indicated by the matching reports discovered. Disk space is allocated for the saved statement and the statement is saved at the SDA allocated. The actual SDA for the statement is inserted in the SDA statement buffer and rewritten on the system disk. The program will terminate with a printout of the statement number assigned to the 'SAVER' statement.


START
SAVER
READ IN STATEMENT SDA TABLE
SCAN FOR BLANK ENTRY IN TABLE
FIND A BLANK ENTRY ?
NO
ERROR: TERMIN-ATE
YES
SAVE ASSIGNED STATEMENT NUMBER
READ IN "SAVER" STATEMENT
SCAN FOR LEFT BRACKET
FIND IT ?
NO
ERROR: TERMINATE
YES
SAVE LOCATION OF LEFT BRACKET
PROCESS DIRECTOR = GETTAC ?
NO
ERROR: TERMINATE
SCAN FOR RIGHT BRACKET
FIND IT ?
NO
ERROR: TERMINATE
YES
LOOK FOR REPORT NUMBER (s)
ANY FOUND ?
NO
ERROR: TERMINATE
YES
SAVE REPORT NUMBER (s) IN TABLE
SCAN31
18-26,249


SCAN 31
SET UP OUTPUT SELECTORS IN SELECTOR TABLE
READ IN REPORT DESC. TABLE
COMPARE REPORT #'s AGAINST FIRST (NEXT) ENTRY
MATCH ?
YES
INSERT SAVED STATEMENT IN REPORT DESC. TABLE ENTRY
NO
END OF REPORT DESC. TABLE ?
NO
PICK - UP OEDT SDA AND SAVE IT
YES
ANY REPORT #'s MATCHED ?
WRITE REPORT DESC. TABLE BACK ON DISK
NO
ALL REPORT #'s MATCHED ?
NO
ERROR: TERMINATE
YES
EXIT TO IDLOOP
TYPE MESSAGE "STATEMENT NNN HAS BEEN SAVED"
WRITE OUTPUT SELECTORS IN OEDT
ALLOCATE SPACE FOR SAVER STATEMENT
WRITE STATEMENT TO DISK AREA JUST ALLOCATED
ADD SDA TO SAVED STATEMENT DIRECTORY
IB - 26,250

LOD

Purpose

LOD processes an Output Program, checking the output format statements syntactically, and condensing them into a convenient form for interpretation and execution by the output phase of REPGEN.

Calling Sequence

Enter LOD on the console typewriter. Return will be to IDLOOP.

Description

The Output Descriptor Card is read and Report Descriptor Table is called from disk. A search is instituted for the report indicated on the Output Descriptor Card.

When the report is found, the SOT Table for this report is called in from disk and updated with the current information from the Output Descriptor Card. An SDA is now allocated for the condensed output format statements.

The SOT Table is now put back on disk with the present run recorded as in error status. The condensed output statement SDA is put into the appropriate DSKACC call.

The LOD statement cards are read in next and are condensed by LOD. Each numbered statement is allotted a complete 245 character record within the assigned SDA. When an attribute name is encountered during processing of the statement cards, an Output Entry Descriptor Table (OEDT -- an expandable 2800 character record) for this report is brought into core from disk. The attribute name is replaced by a six digit value from the OEDT. This value specifies the attribute value length and its relative position in the intermediate tape created by the first phase of REPGEN.

The OEDT usually will remain in core throughout the processing of the present statement cards except when an additional segment of the OEDT has to be called in from disk. On the next reference to the OEDT the first segment of the table will be brought back into core.

When the final statement has been processed, the report descriptor table will be written back onto disk, after being submitted to a minor update. This update involves putting an indicator for output media into the table.

A second wind-up operation puts the updated SOT Table onto disk. A successful set of statements get an "A" or "C" in position Ø of the table. Otherwise an "E" is left in the table.

Phrase Types and Condensed Form

Type 1

DO XXX	1DXXX
REPEAT XXX	1RXXX
GO TO XXX	1GXXX
CONTINUE XXX	1CXXX

Type 2

SET OVERFLOW AFTER NN	2SONN
SET SWITCH N	2SSN
SET COUNTER NN LENGTH=PP	2SCNNPP
SET REMAINDER LENGTH=NN	2SCØØNN
CLEAR SWITCH N	2CSN
CLEAR SWITCHES	2CS
CLEAR COUNTER NN	2CCNN
CLEAR COUNTERS	2CC
CLEAR	2

Type 3

SKIP TO A NEW PAGE	3E1
SKIP NN	3CNN
STACKER N	3SN
WRITE TAPE MARK	3T

Type 4

END LINE	4Ø
END CARD	4Ø
END HEADER XXX	4HXXX
END TRAILER XXX	4TXXX
END ENTRY	4Ø
NEXT ENTRY	4
END	1

Type 5

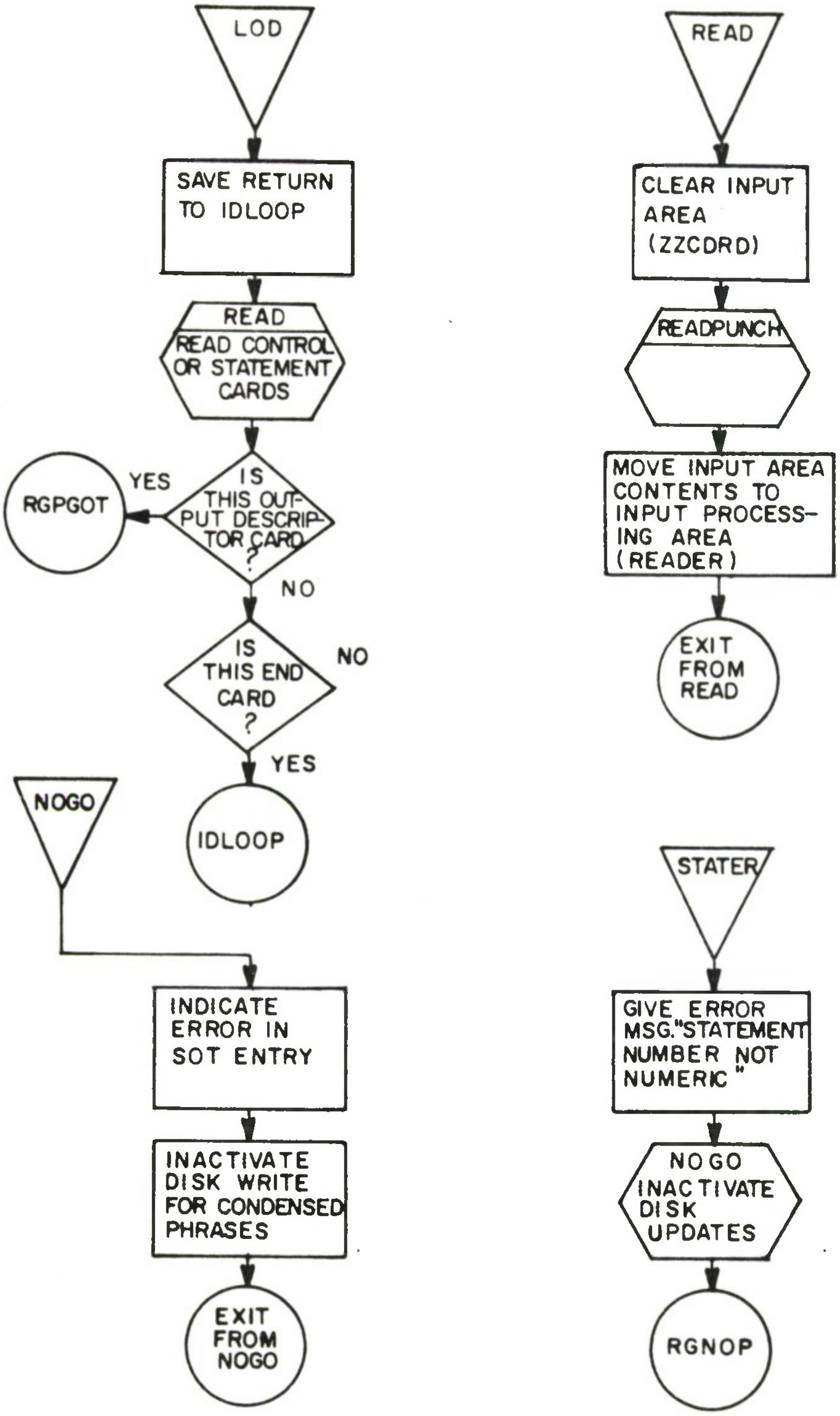
IF OVERFLOW	5Ø
IF FINISHED	5F

Type 5 (continued)

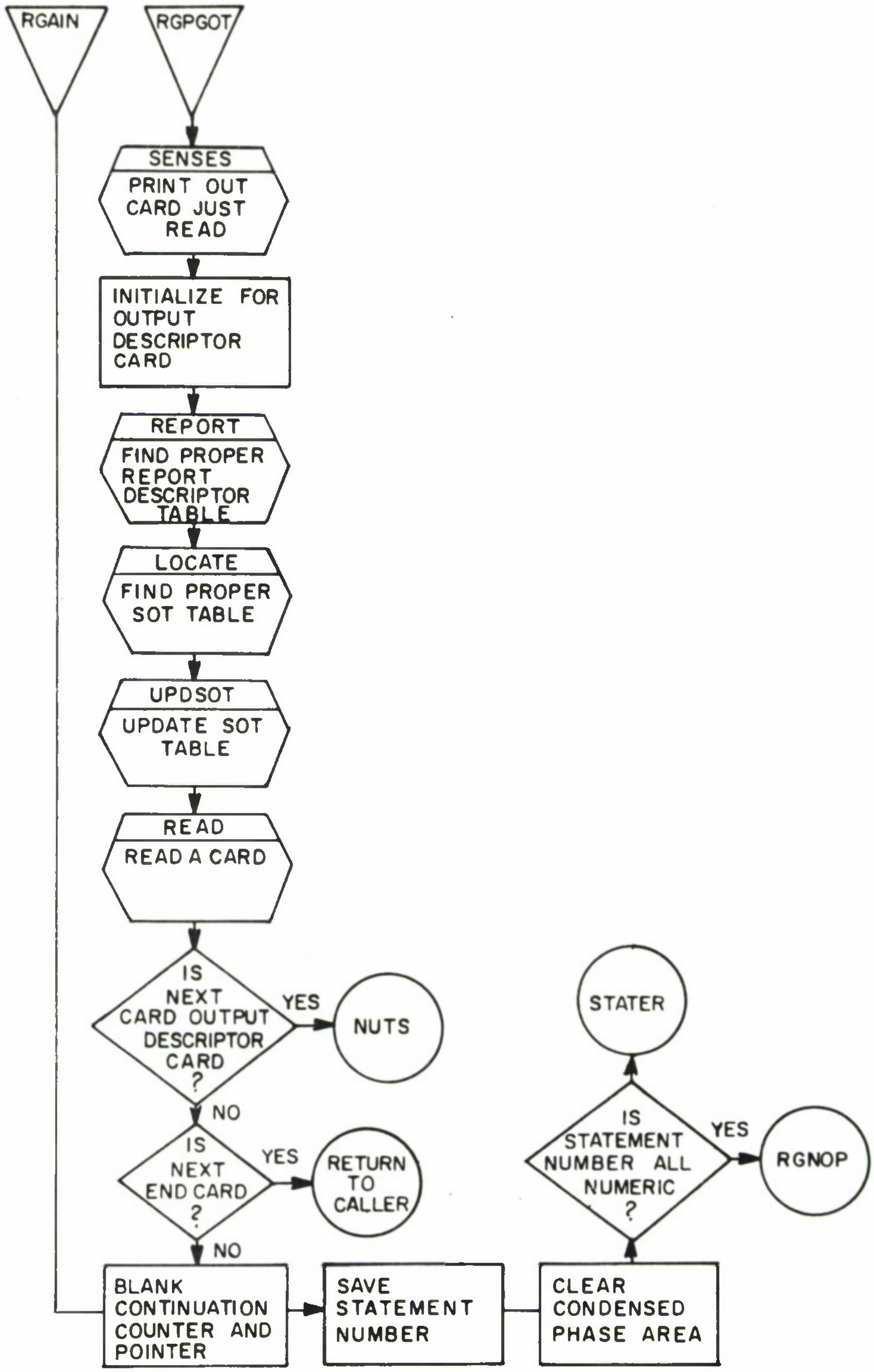
```
IF SWITCH X                                  5SX

               <                                     U
     N                                         N     S N
IF   b (....)  = N (....)/                     5 b ....  / b ....
     D         U D                             D     T D
               >
```

Type 6

```
              N                                   N
        L     S                               L   S
COL XXX     = D (....)                   6XXX     D ....
        R     C                               R   C
              b                                   b

          N                                   N
COL XXX =   (....)                      6XXXR   ....
          S                                   S

          D                                   D
COL XXX = C (....)                      6XXXL C ....
          b                                   b

                          +                       N          A
        L     N  N        - N                 L   S N ....   S N ....
COL XXX     =   (  (....)       (....))  6XXX                (
        R     S  b        ÷ b                 R   b          ,   b
                          x

                      +                                    A
          N  N        - N                     N N          S N
COL XXX =   (  (....)       (....))     6XXXR     ....     (
          S  b        ÷ b                     S b          ,   b
                      x

             N                                N
COUNTER XX = D (....)                   6CXX  D ....
             b                                b

                         +                             A
             N  N        - N                  N N      S N
COUNTER XX =   (  (....)       (....))  6CXX      .... (
             S  b        ÷ b                  S b      ,   b
                         x
```

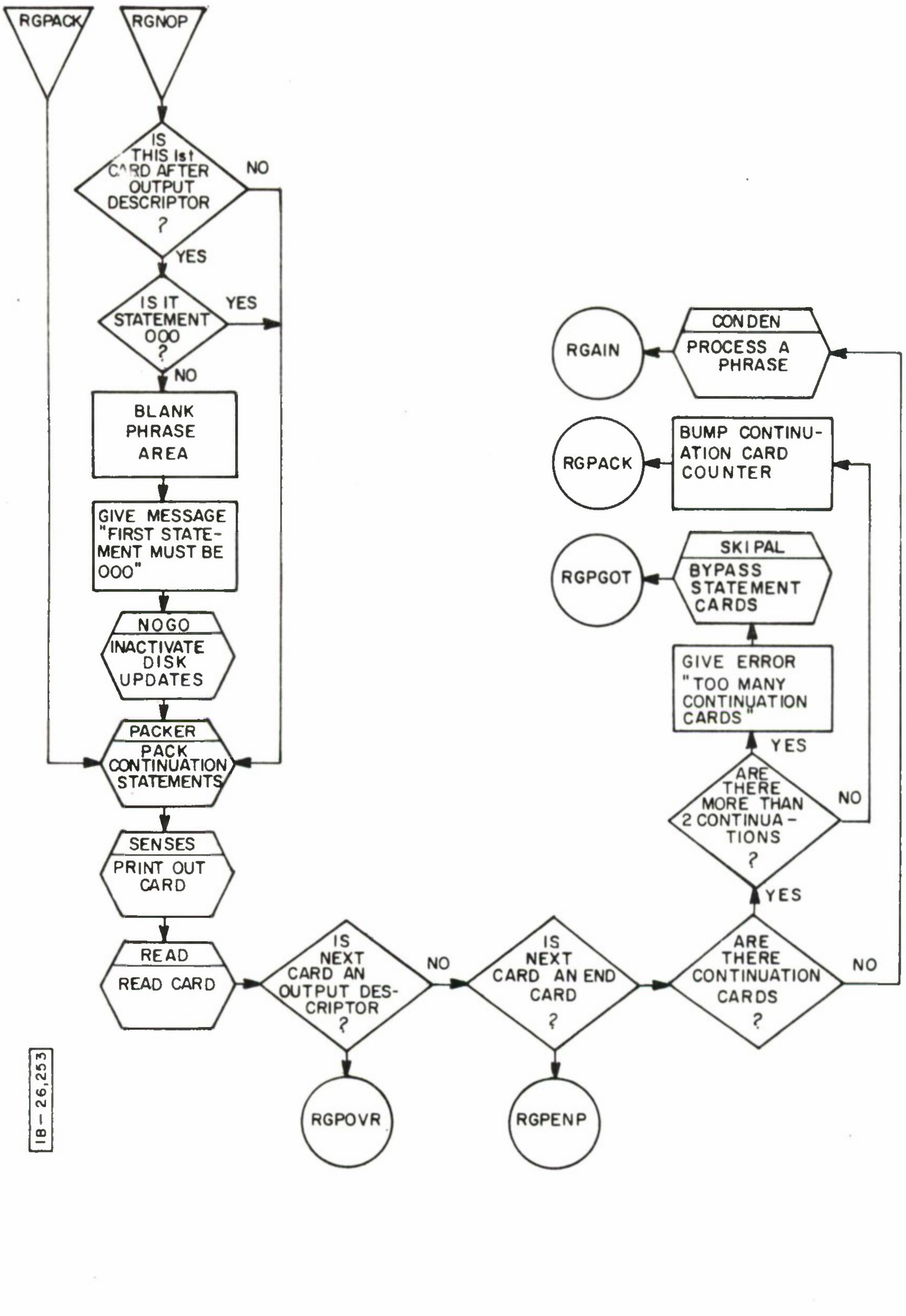

LOD
SAVE RETURN TO IDLOOP
READ
READ CONTROL OR STATEMENT CARDS
IS THIS OUTPUT DESCRIPTOR CARD ?
YES
RGPGOT
NO
IS THIS END CARD ?
NO
YES
IDLOOP
READ
CLEAR INPUT AREA (ZZCDRD)
READPUNCH
MOVE INPUT AREA CONTENTS TO INPUT PROCESSING AREA (READER)
EXIT FROM READ
NOGO
INDICATE ERROR IN SOT ENTRY
INACTIVATE DISK WRITE FOR CONDENSED PHRASES
EXIT FROM NOGO
STATER
GIVE ERROR MSG."STATEMENT NUMBER NOT NUMERIC "
NOGO INACTIVATE DISK UPDATES
RGNOP

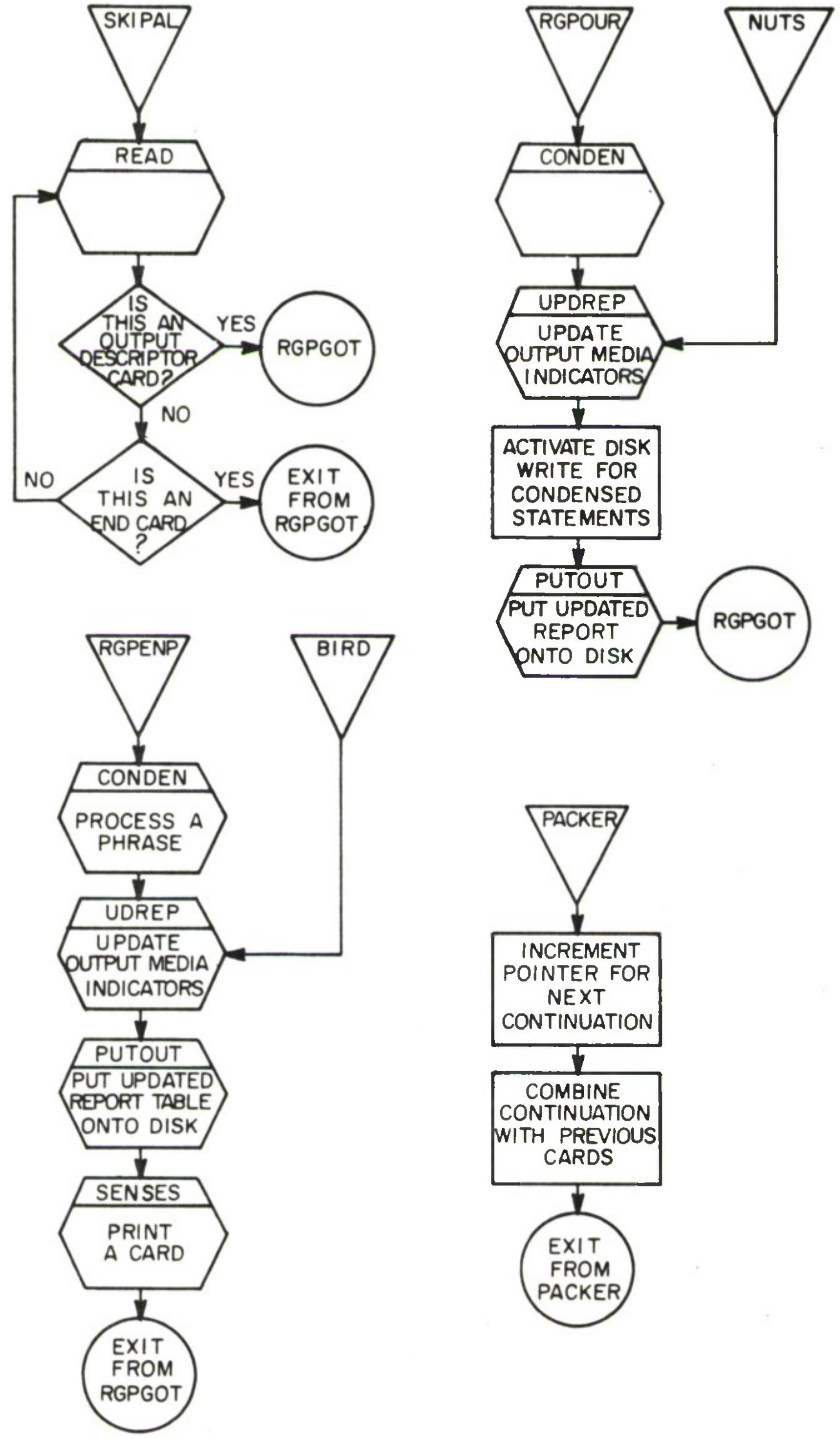
IB-26,251



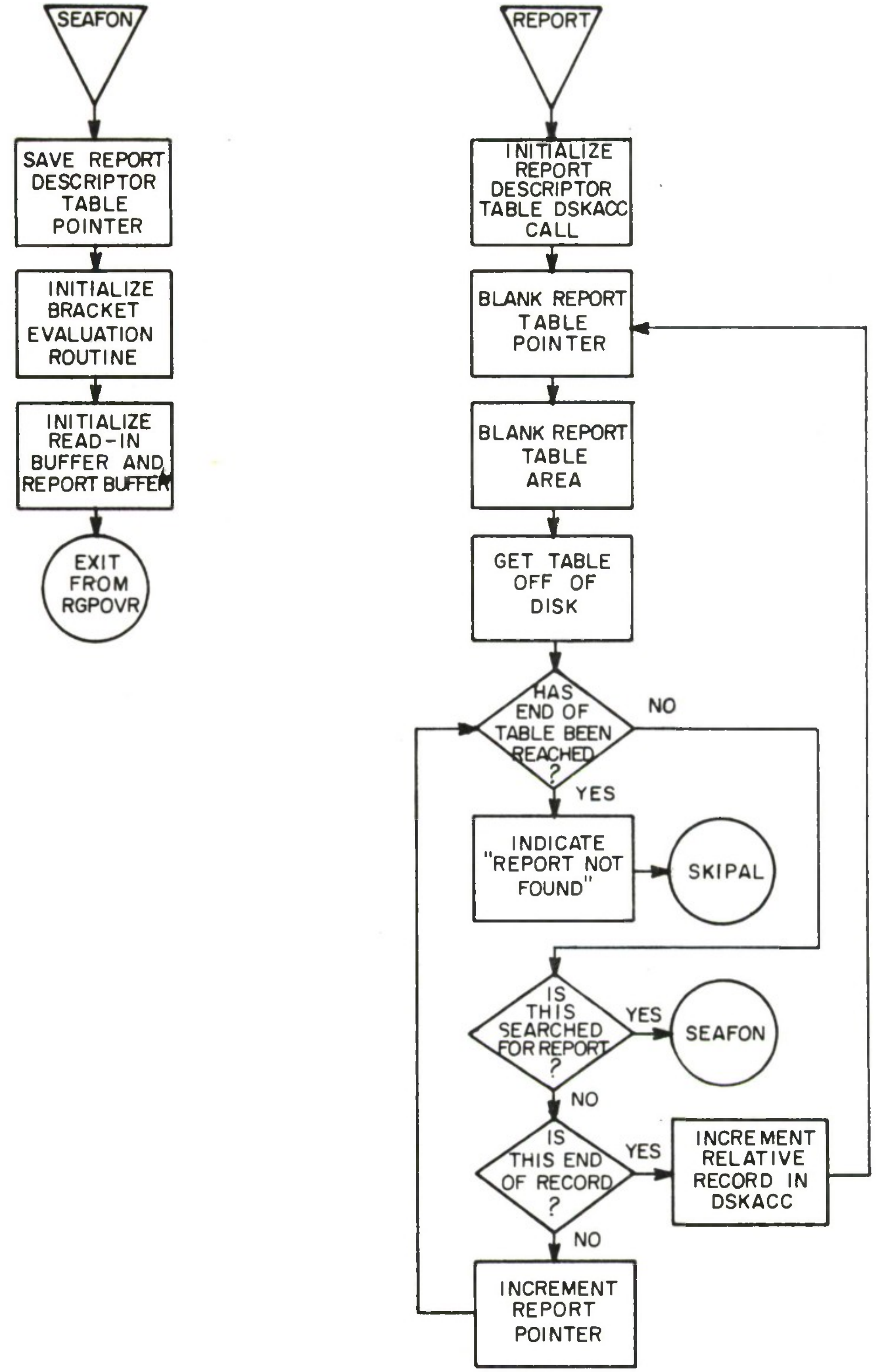
RGAIN
RGPGOT
SENSES
PRINT OUT CARD JUST READ
INITIALIZE FOR OUTPUT DESCRIPTOR CARD
REPORT
FIND PROPER REPORT DESCRIPTOR TABLE
LOCATE
FIND PROPER SOT TABLE
UPDSOT
UPDATE SOT TABLE
READ
READ A CARD
IS NEXT CARD OUTPUT DESCRIPTOR CARD ?
YES
NUTS
NO
IS NEXT END CARD ?
YES
RETURN TO CALLER
NO
BLANK CONTINUATION COUNTER AND POINTER
SAVE STATEMENT NUMBER
CLEAR CONDENSED PHASE AREA
IS STATEMENT NUMBER ALL NUMERIC ?
YES
RGNOP
STATER

1B – 26,252

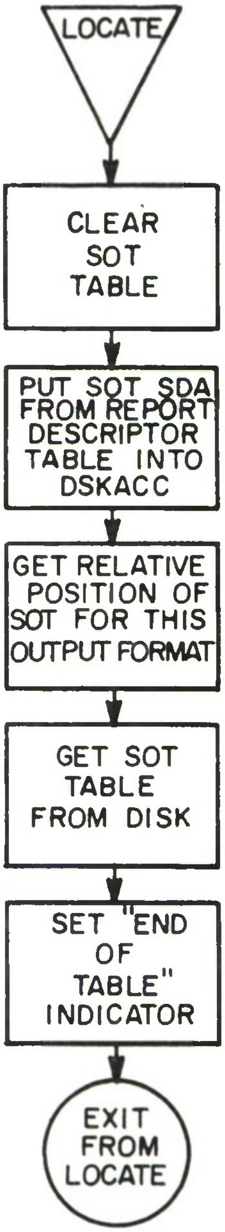

RGPACK
RGNOP
IS THIS Ist CARD AFTER OUTPUT DESCRIPTOR ?
NO
YES
IS IT STATEMENT 000 ?
YES
NO
BLANK PHRASE AREA
GIVE MESSAGE "FIRST STATEMENT MUST BE 000"
NOGO
INACTIVATE DISK UPDATES
PACKER
PACK CONTINUATION STATEMENTS
SENSES
PRINT OUT CARD
READ
READ CARD
IS NEXT CARD AN OUTPUT DESCRIPTOR ?
NO
RGPOVR
IS NEXT CARD AN END CARD ?
RGPENP
ARE THERE CONTINUATION CARDS ?
NO
YES
ARE THERE MORE THAN 2 CONTINUATIONS ?
NO
YES
GIVE ERROR "TOO MANY CONTINUATION CARDS"
SKIPAL
BYPASS STATEMENT CARDS
RGPGOT
BUMP CONTINUATION CARD COUNTER
RGPACK
CONDEN
PROCESS A PHRASE
RGAIN
IB-26,253


SKIPAL
READ
IS THIS AN OUTPUT DESCRIPTOR CARD?
YES
RGPGOT
NO
NO
IS THIS AN END CARD ?
YES
EXIT FROM RGPGOT
RGPOUR
NUTS
CONDEN
UPDREP
UPDATE OUTPUT MEDIA INDICATORS
ACTIVATE DISK WRITE FOR CONDENSED STATEMENTS
PUTOUT
PUT UPDATED REPORT ONTO DISK
RGPGOT
RGPENP
BIRD
CONDEN
PROCESS A PHRASE
UDREP
UPDATE OUTPUT MEDIA INDICATORS
PUTOUT
PUT UPDATED REPORT TABLE ONTO DISK
SENSES
PRINT A CARD
EXIT FROM RGPGOT
PACKER
INCREMENT POINTER FOR NEXT CONTINUATION
COMBINE CONTINUATION WITH PREVIOUS CARDS
EXIT FROM PACKER

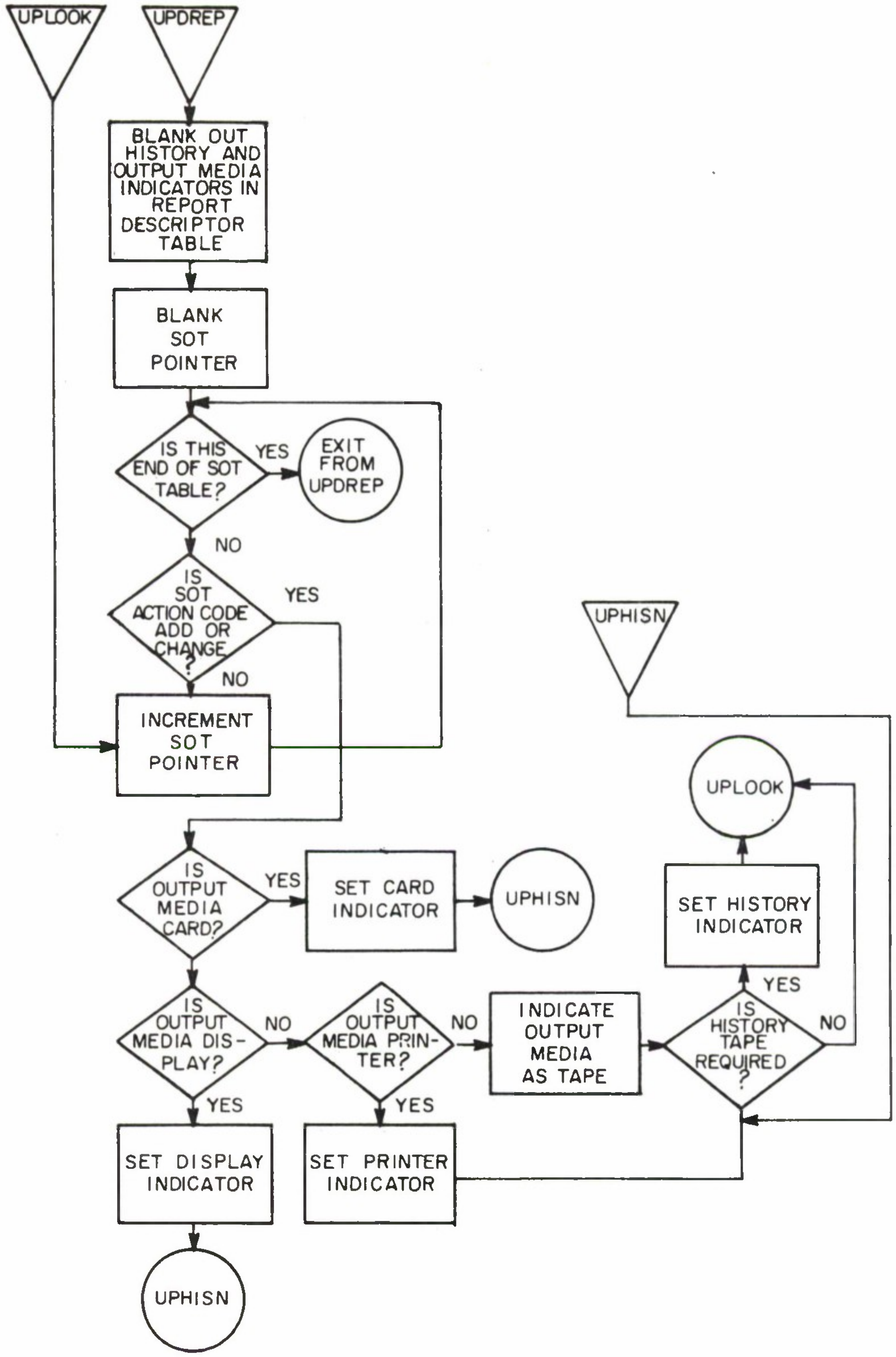
IB-26,198


SEAFON
SAVE REPORT DESCRIPTOR TABLE POINTER
INITIALIZE BRACKET EVALUATION ROUTINE
INITIALIZE READ-IN BUFFER AND REPORT BUFFER
EXIT FROM RGPOVR
REPORT
INITIALIZE REPORT DESCRIPTOR TABLE DSKACC CALL
BLANK REPORT TABLE POINTER
BLANK REPORT TABLE AREA
GET TABLE OFF OF DISK
HAS END OF TABLE BEEN REACHED ?
NO
YES
INDICATE "REPORT NOT FOUND"
SKIPAL
IS THIS SEARCHED FOR REPORT ?
YES
SEAFON
NO
IS THIS END OF RECORD ?
YES
INCREMENT RELATIVE RECORD IN DSKACC
NO
INCREMENT REPORT POINTER

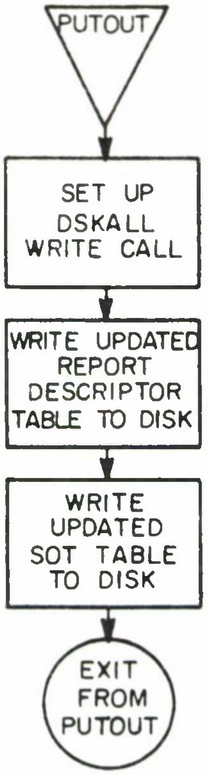
1B-26,197

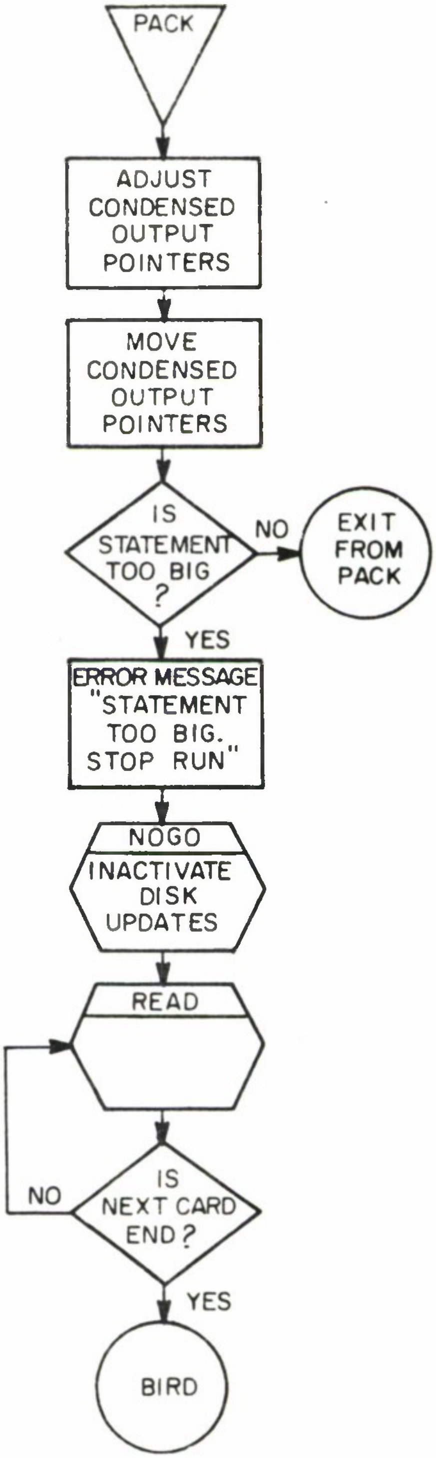

LOCATE
CLEAR SOT TABLE
PUT SOT SDA FROM REPORT DESCRIPTOR TABLE INTO DSKACC
GET RELATIVE POSITION OF SOT FOR THIS OUTPUT FORMAT
GET SOT TABLE FROM DISK
SET "END OF TABLE" INDICATOR
EXIT FROM LOCATE

IB – 26,193

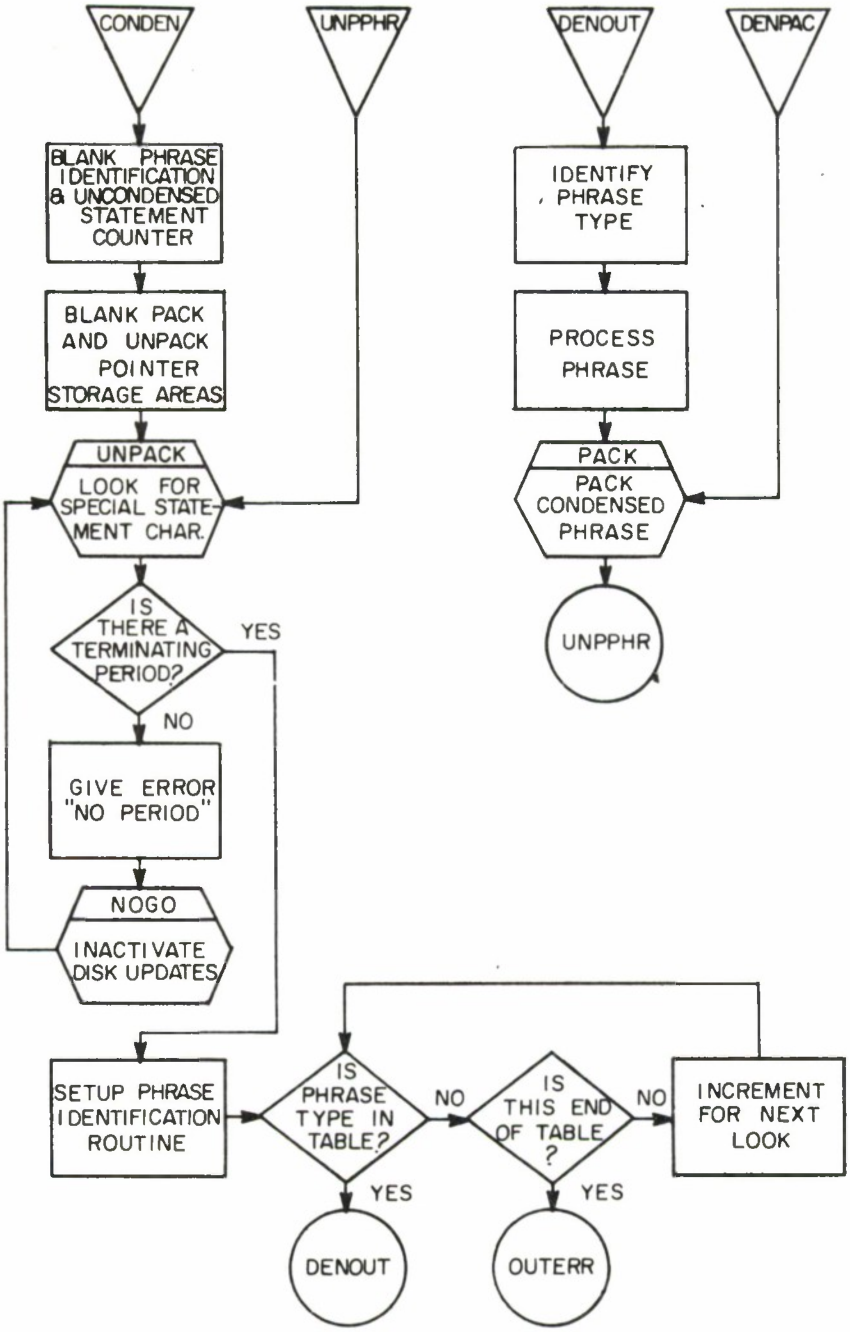

UPLOOK
UPDREP
BLANK OUT HISTORY AND OUTPUT MEDIA INDICATORS IN REPORT DESCRIPTOR TABLE
BLANK SOT POINTER
IS THIS END OF SOT TABLE?
YES
EXIT FROM UPDREP
NO
IS SOT ACTION CODE ADD OR CHANGE ?
YES
NO
INCREMENT SOT POINTER
UPHISN
UPLOOK
IS OUTPUT MEDIA CARD?
YES
SET CARD INDICATOR
UPHISN
SET HISTORY INDICATOR
YES
IS OUTPUT MEDIA DIS-PLAY?
NO
IS OUTPUT MEDIA PRIN-TER?
NO
INDICATE OUTPUT MEDIA AS TAPE
IS HISTORY TAPE REQUIRED ?
NO
YES
YES
SET DISPLAY INDICATOR
SET PRINTER INDICATOR
UPHISN

IB – 26,194

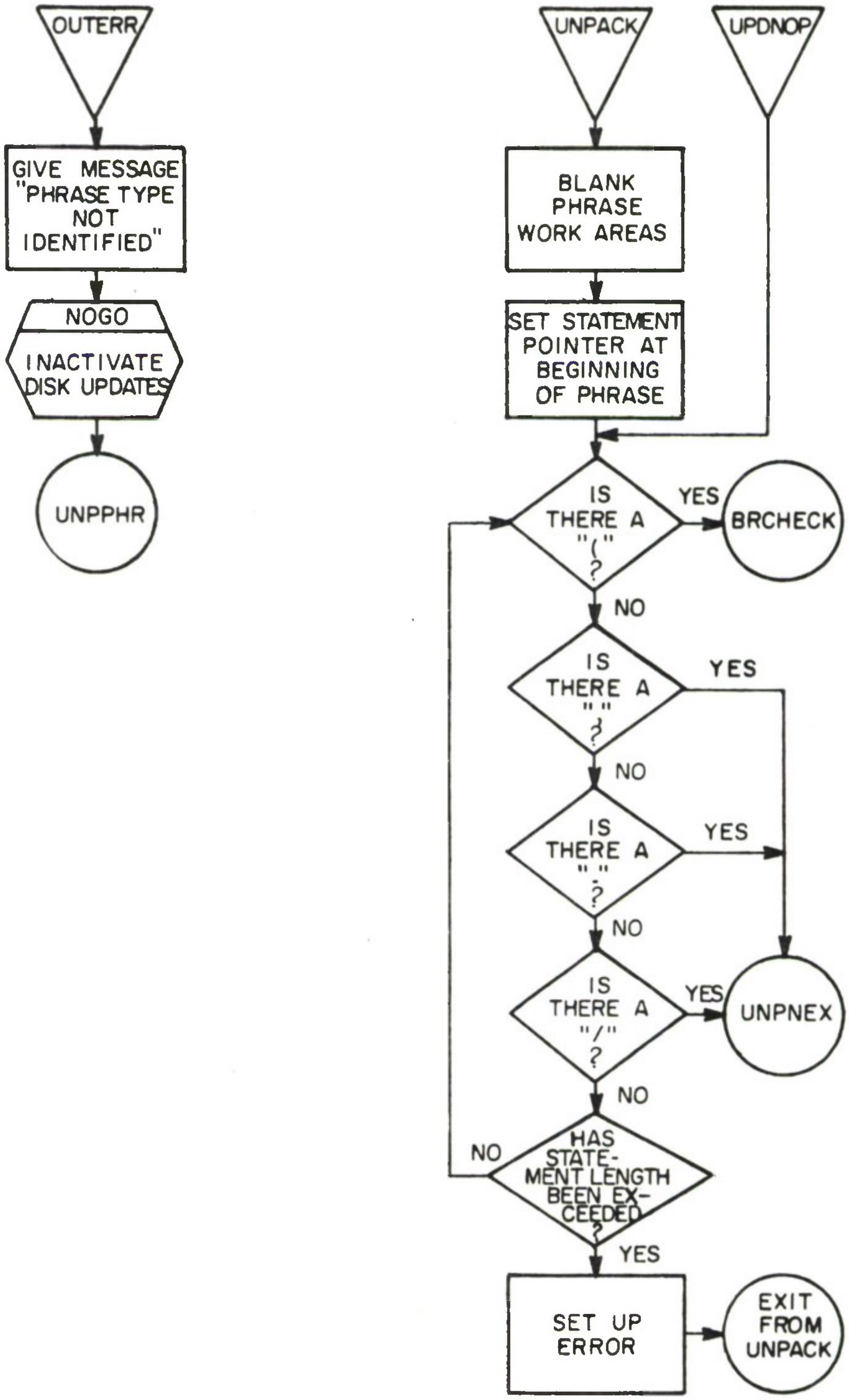

PUTOUT
SET UP DSKALL WRITE CALL
WRITE UPDATED REPORT DESCRIPTOR TABLE TO DISK
WRITE UPDATED SOT TABLE TO DISK
EXIT FROM PUTOUT


PACK
ADJUST CONDENSED OUTPUT POINTERS
MOVE CONDENSED OUTPUT POINTERS
IS STATEMENT TOO BIG ?
NO
EXIT FROM PACK
YES
ERROR MESSAGE "STATEMENT TOO BIG. STOP RUN"
NOGO
INACTIVATE DISK UPDATES
READ
IS NEXT CARD END ?
NO
YES
BIRD

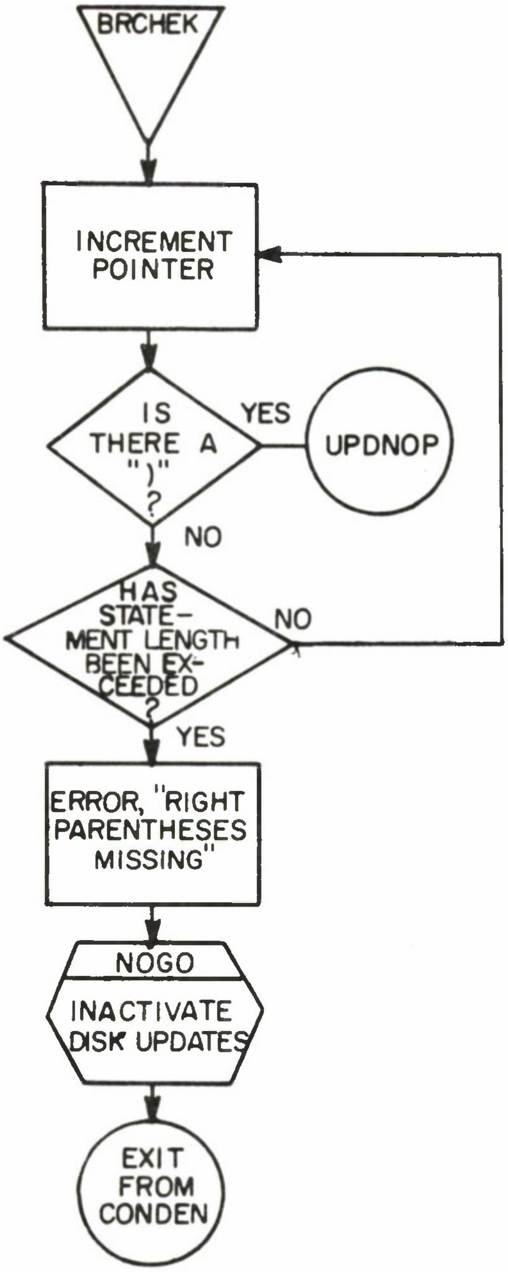
IB-26,195

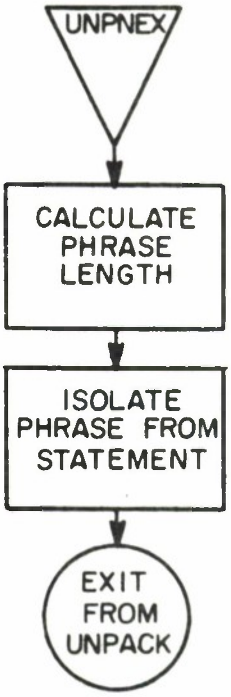

CONDEN
UNPPHR
DENOUT
DENPAC
BLANK PHRASE IDENTIFICATION & UNCONDENSED STATEMENT COUNTER
BLANK PACK AND UNPACK POINTER STORAGE AREAS
UNPACK
LOOK FOR SPECIAL STATEMENT CHAR.
IS THERE A TERMINATING PERIOD?
YES
NO
GIVE ERROR "NO PERIOD"
NOGO
INACTIVATE DISK UPDATES
SETUP PHRASE IDENTIFICATION ROUTINE
IS PHRASE TYPE IN TABLE?
NO
YES
DENOUT
IS THIS END OF TABLE ?
NO
YES
OUTERR
INCREMENT FOR NEXT LOOK
IDENTIFY PHRASE TYPE
PROCESS PHRASE
PACK
PACK CONDENSED PHRASE
UNPPHR

1B-26,196

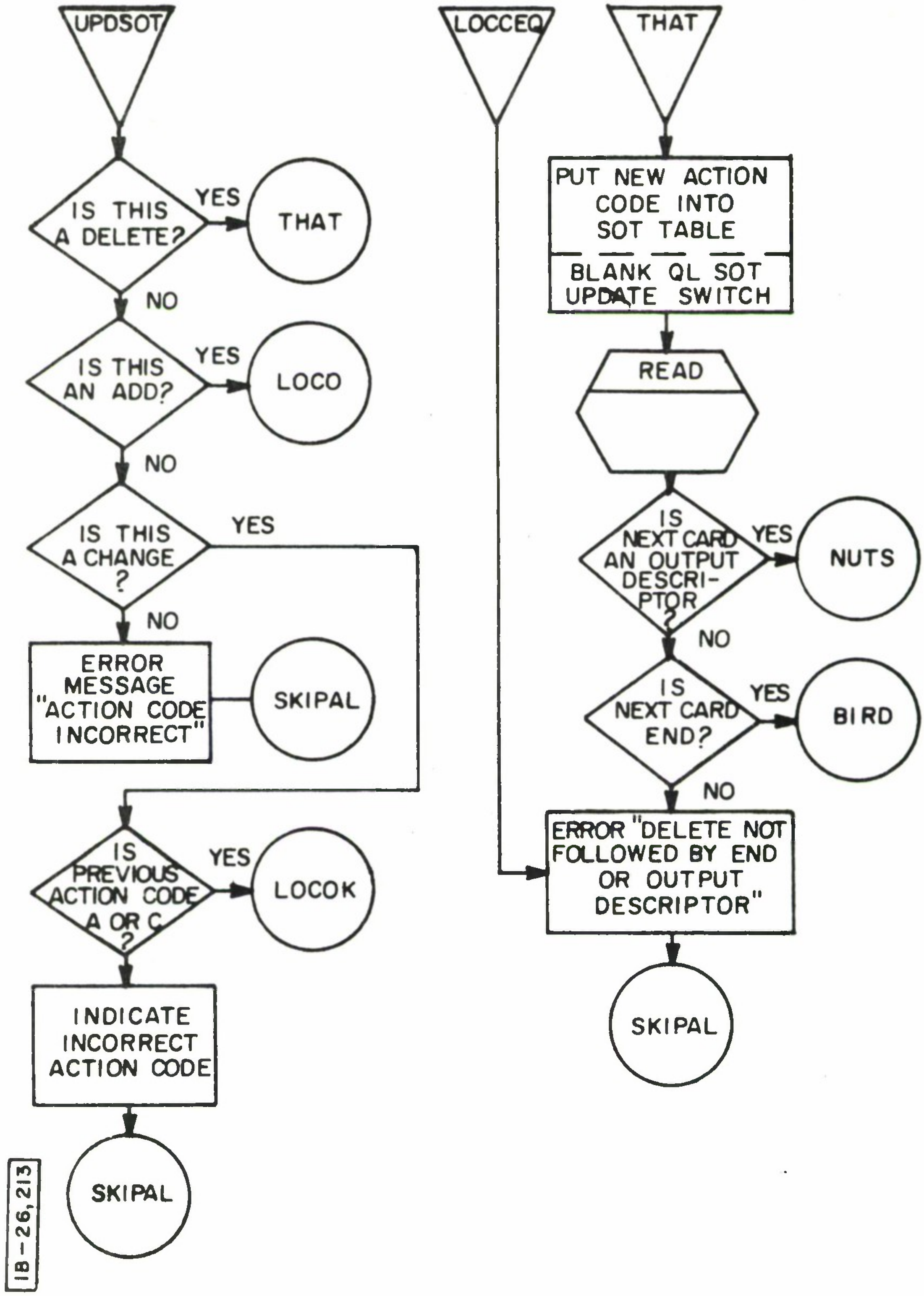

OUTERR
GIVE MESSAGE "PHRASE TYPE NOT IDENTIFIED"
NOGO
INACTIVATE DISK UPDATES
UNPPHR
UNPACK
UPDNOP
BLANK PHRASE WORK AREAS
SET STATEMENT POINTER AT BEGINNING OF PHRASE
IS THERE A "(" ?
YES
BRCHECK
NO
IS THERE A "," ?
YES
NO
IS THERE A "." ?
YES
NO
IS THERE A "/" ?
YES
UNPNEX
NO
HAS STATE-MENT LENGTH BEEN EX-CEEDED ?
NO
YES
SET UP ERROR
EXIT FROM UNPACK

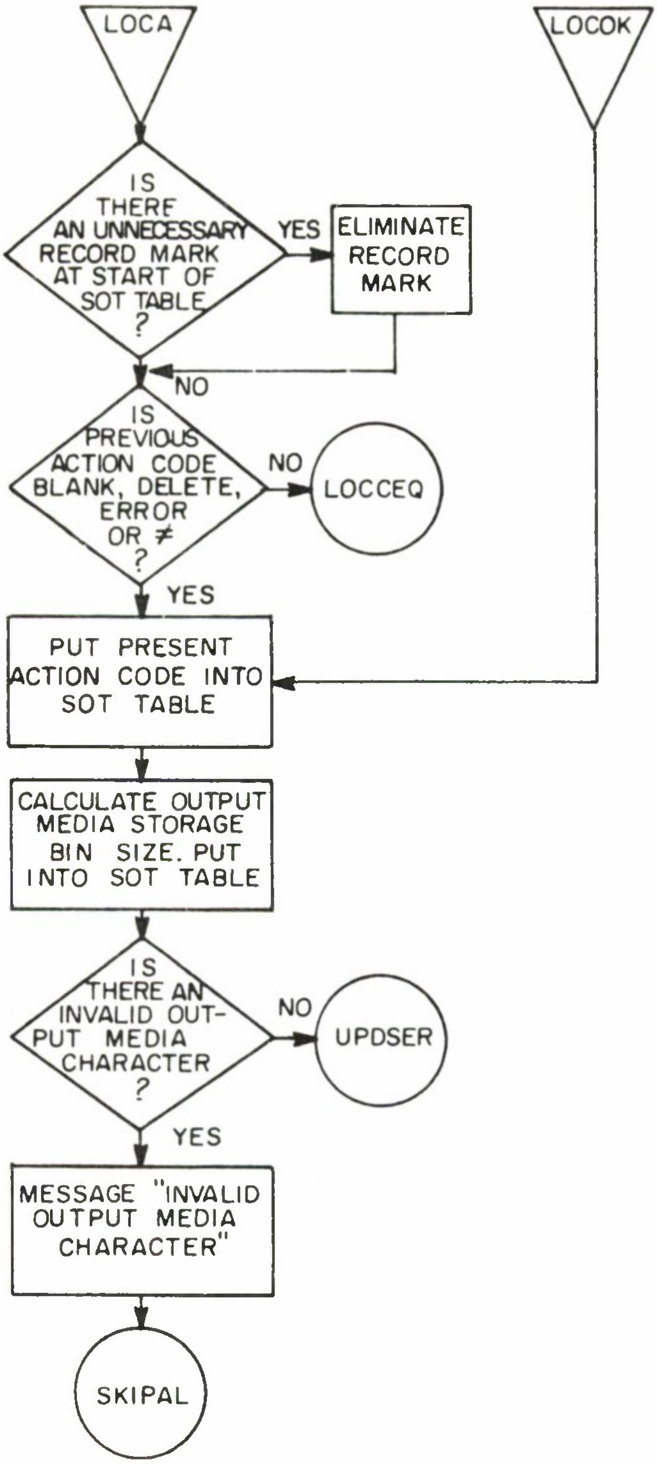
18-26,190


BRCHEK
INCREMENT POINTER
IS THERE A ")" ?
YES
UPDNOP
NO
HAS STATE-MENT LENGTH BEEN EX-CEEDED ?
NO
YES
ERROR, "RIGHT PARENTHESES MISSING"
NOGO
INACTIVATE DISK UPDATES
EXIT FROM CONDEN

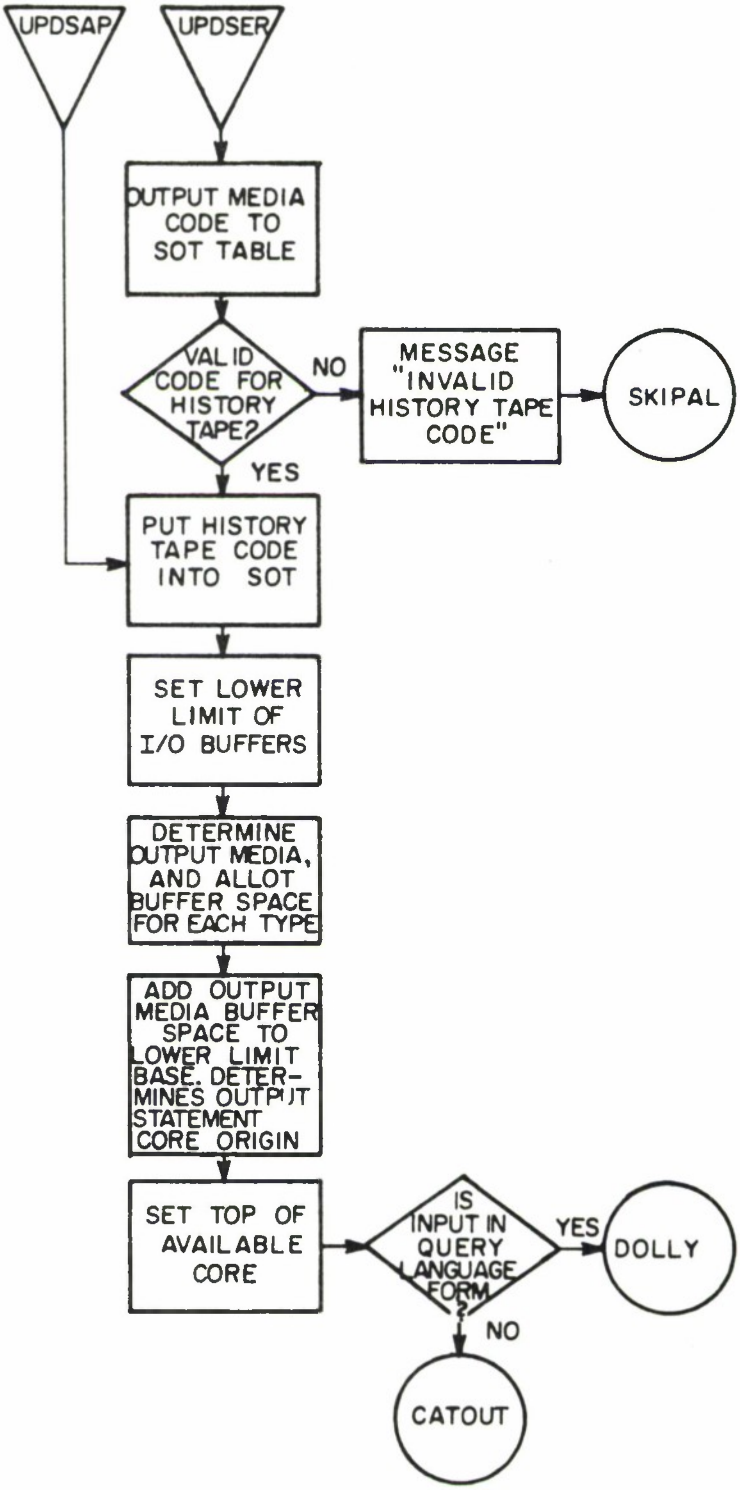

UNPNEX
CALCULATE PHRASE LENGTH
ISOLATE PHRASE FROM STATEMENT
EXIT FROM UNPACK

IB-26,191

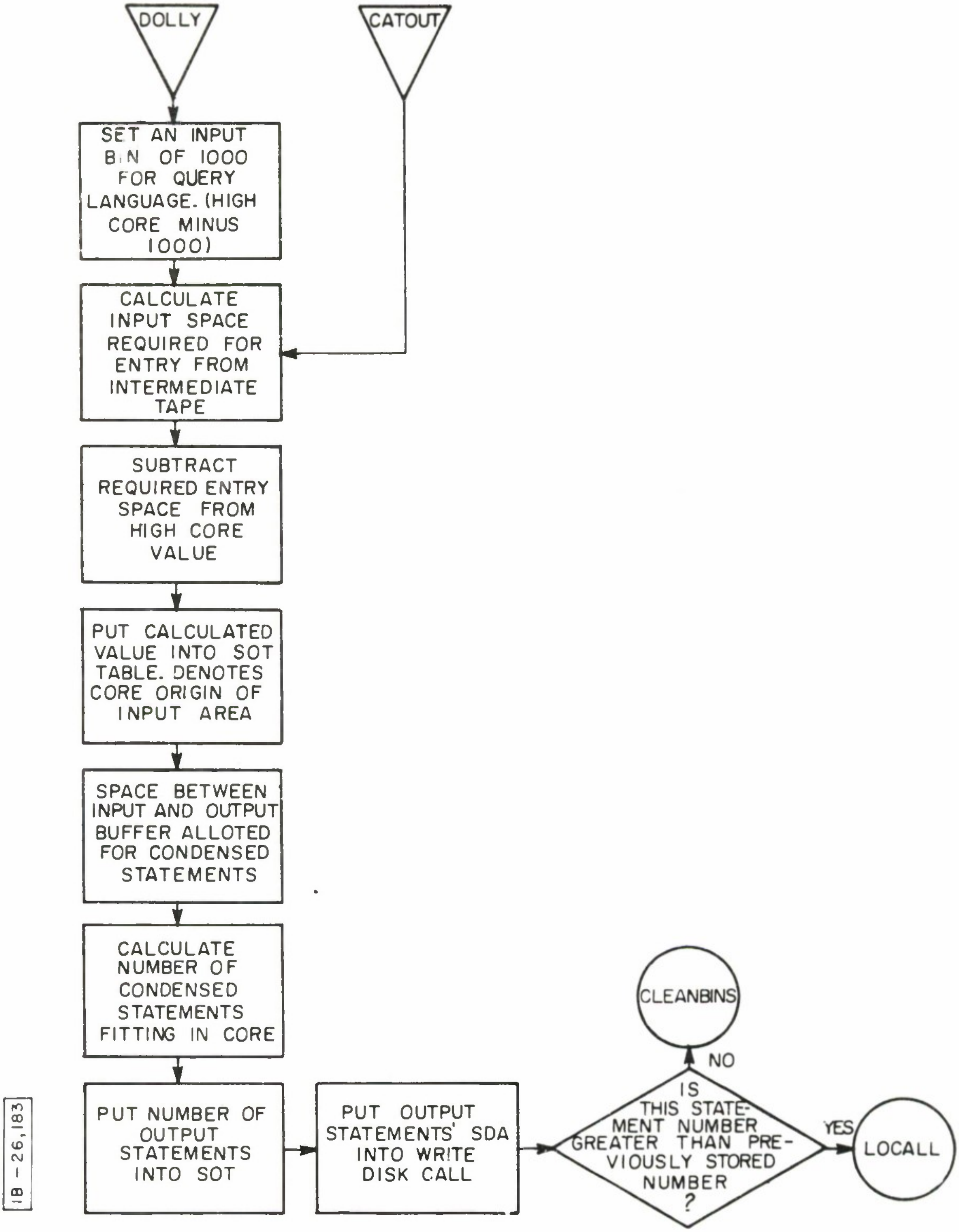

UPDSOT
IS THIS A DELETE?
YES
THAT
NO
IS THIS AN ADD?
YES
LOCO
NO
IS THIS A CHANGE ?
YES
NO
ERROR MESSAGE "ACTION CODE INCORRECT"
SKIPAL
IS PREVIOUS ACTION CODE A OR C ?
YES
LOCOK
INDICATE INCORRECT ACTION CODE
SKIPAL
LOCCEQ
THAT
PUT NEW ACTION CODE INTO SOT TABLE
BLANK QL SOT UPDATE SWITCH
READ
IS NEXT CARD AN OUTPUT DESCRIPTOR ?
YES
NUTS
NO
IS NEXT CARD END?
YES
BIRD
NO
ERROR "DELETE NOT FOLLOWED BY END OR OUTPUT DESCRIPTOR"
SKIPAL
IB-26,213

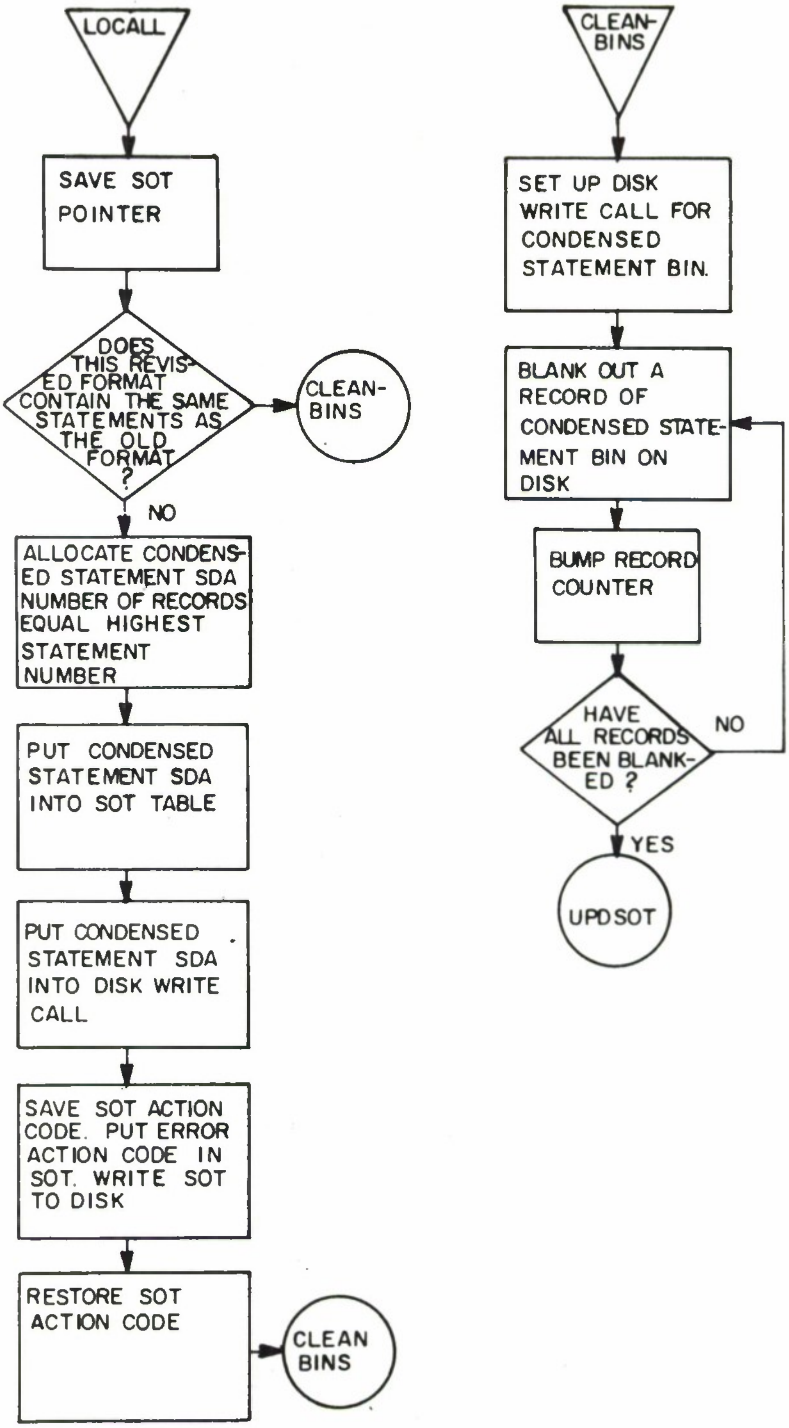

LOCA
LOCOK
IS THERE AN UNNECESSARY RECORD MARK AT START OF SOT TABLE ?
YES
ELIMINATE RECORD MARK
NO
IS PREVIOUS ACTION CODE BLANK, DELETE, ERROR OR ≠ ?
NO
LOCCEQ
YES
PUT PRESENT ACTION CODE INTO SOT TABLE
CALCULATE OUTPUT MEDIA STORAGE BIN SIZE. PUT INTO SOT TABLE
IS THERE AN INVALID OUTPUT MEDIA CHARACTER ?
NO
UPDSER
YES
MESSAGE "INVALID OUTPUT MEDIA CHARACTER"
SKIPAL

1B-26,214

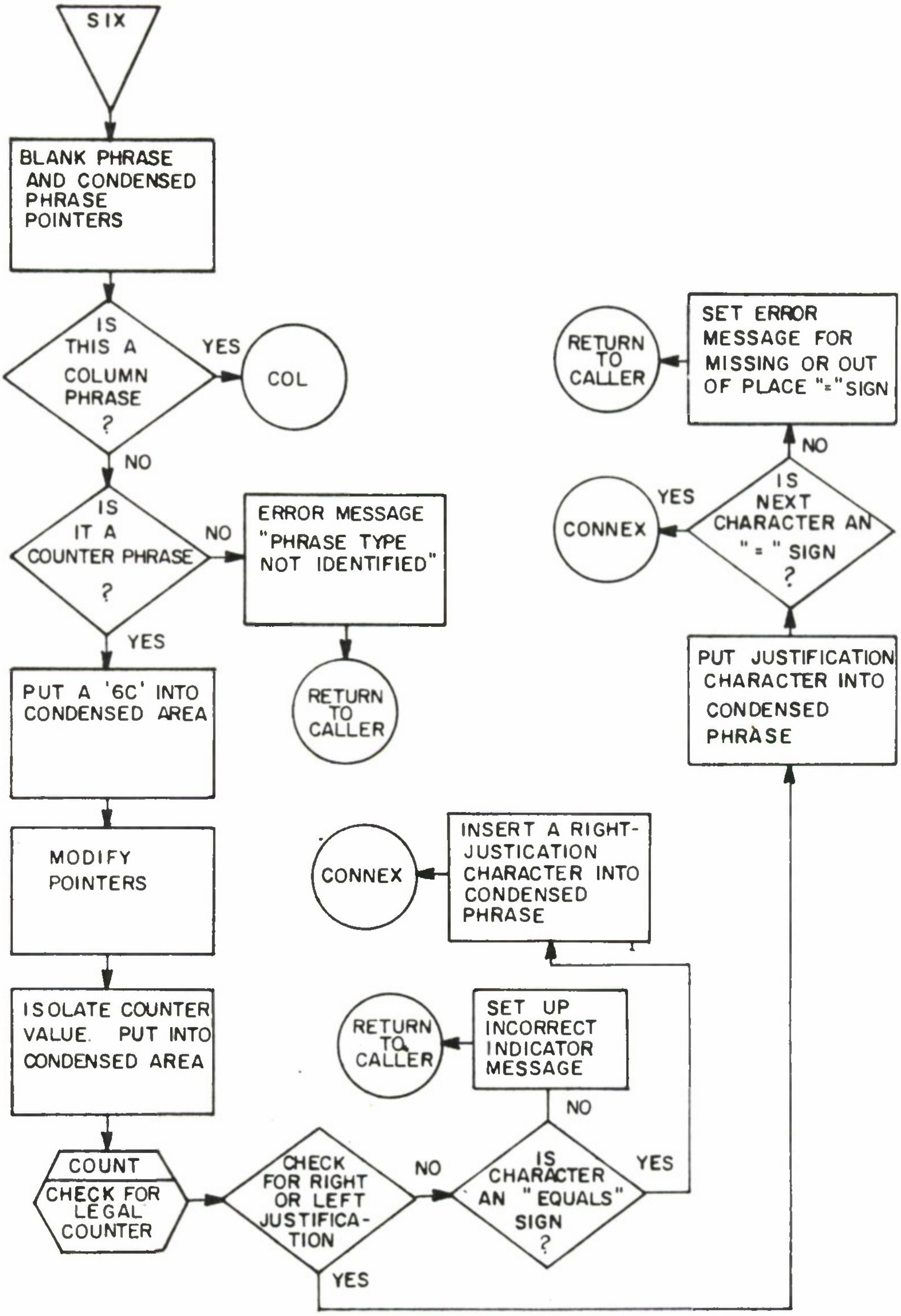

UPDSAP
UPDSER
OUTPUT MEDIA CODE TO SOT TABLE
VALID CODE FOR HISTORY TAPE?
NO
MESSAGE "INVALID HISTORY TAPE CODE"
SKIPAL
YES
PUT HISTORY TAPE CODE INTO SOT
SET LOWER LIMIT OF I/O BUFFERS
DETERMINE OUTPUT MEDIA, AND ALLOT BUFFER SPACE FOR EACH TYPE
ADD OUTPUT MEDIA BUFFER SPACE TO LOWER LIMIT BASE. DETERMINES OUTPUT STATEMENT CORE ORIGIN
SET TOP OF AVAILABLE CORE
IS INPUT IN QUERY LANGUAGE FORM ?
YES
DOLLY
NO
CATOUT

IB-26,182

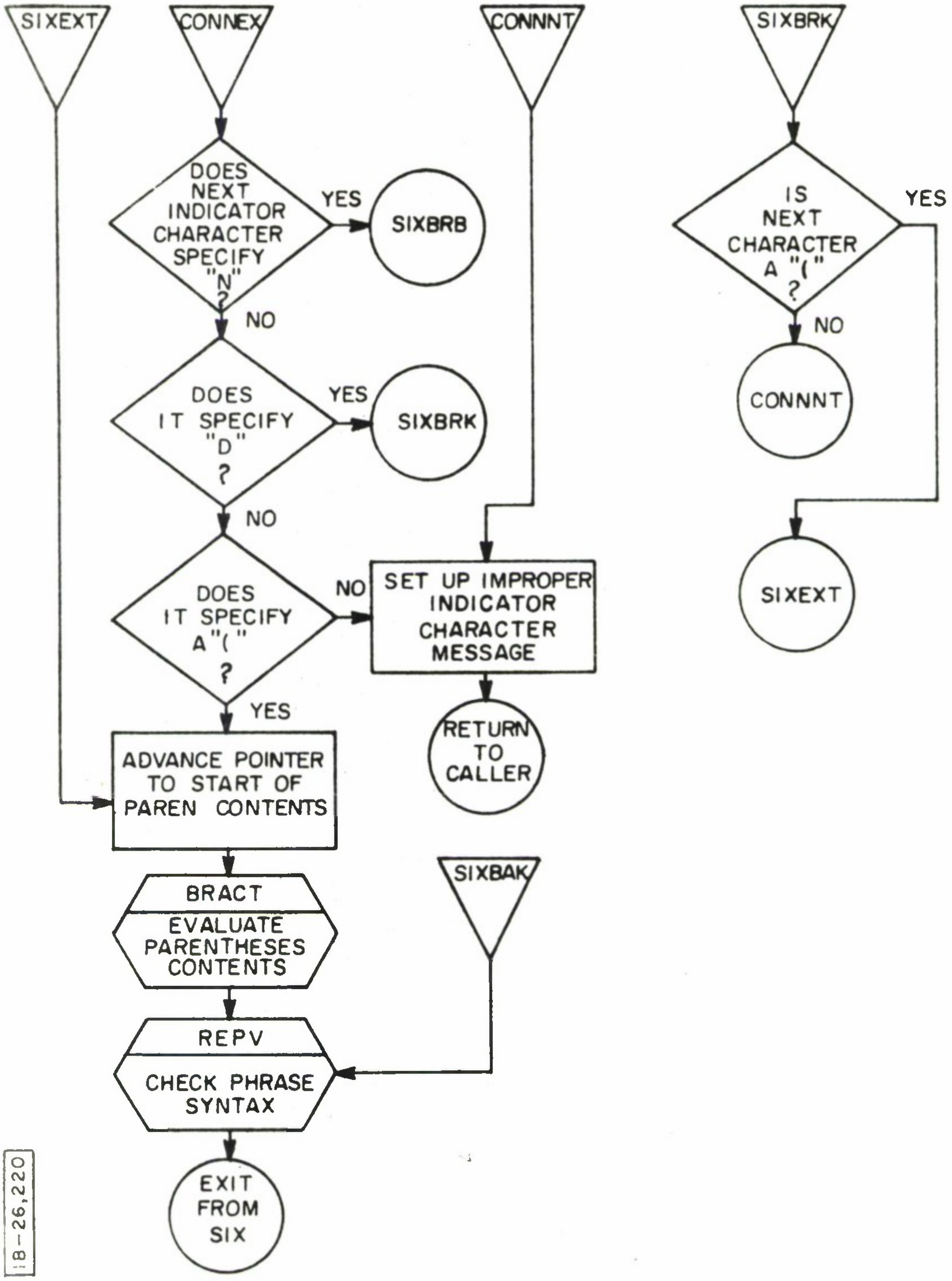

DOLLY
CATOUT
SET AN INPUT BIN OF 1000 FOR QUERY LANGUAGE. (HIGH CORE MINUS 1000)
CALCULATE INPUT SPACE REQUIRED FOR ENTRY FROM INTERMEDIATE TAPE
SUBTRACT REQUIRED ENTRY SPACE FROM HIGH CORE VALUE
PUT CALCULATED VALUE INTO SOT TABLE. DENOTES CORE ORIGIN OF INPUT AREA
SPACE BETWEEN INPUT AND OUTPUT BUFFER ALLOTED FOR CONDENSED STATEMENTS
CALCULATE NUMBER OF CONDENSED STATEMENTS FITTING IN CORE
PUT NUMBER OF OUTPUT STATEMENTS INTO SOT
PUT OUTPUT STATEMENTS' SDA INTO WRITE DISK CALL
IS THIS STATEMENT NUMBER GREATER THAN PREVIOUSLY STORED NUMBER ?
NO
CLEANBINS
YES
LOCALL
IB - 26,183

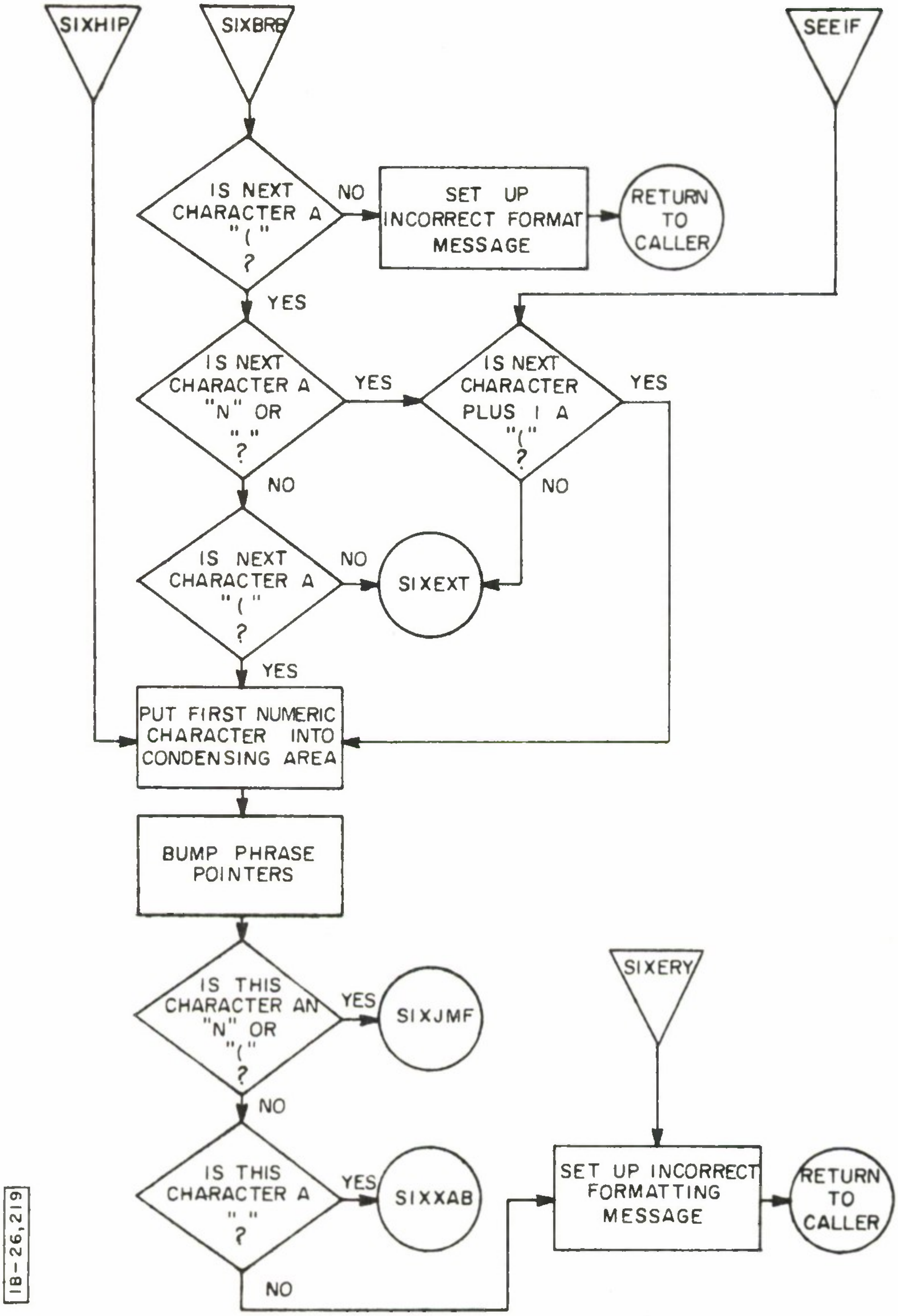

LOCALL
SAVE SOT POINTER
DOES THIS REVISED FORMAT CONTAIN THE SAME STATEMENTS AS THE OLD FORMAT ?
CLEAN-BINS
NO
ALLOCATE CONDENSED STATEMENT SDA NUMBER OF RECORDS EQUAL HIGHEST STATEMENT NUMBER
PUT CONDENSED STATEMENT SDA INTO SOT TABLE
PUT CONDENSED STATEMENT SDA INTO DISK WRITE CALL
SAVE SOT ACTION CODE. PUT ERROR ACTION CODE IN SOT. WRITE SOT TO DISK
RESTORE SOT ACTION CODE
CLEAN BINS
CLEAN-BINS
SET UP DISK WRITE CALL FOR CONDENSED STATEMENT BIN.
BLANK OUT A RECORD OF CONDENSED STATEMENT BIN ON DISK
BUMP RECORD COUNTER
HAVE ALL RECORDS BEEN BLANKED ?
NO
YES
UPDSOT

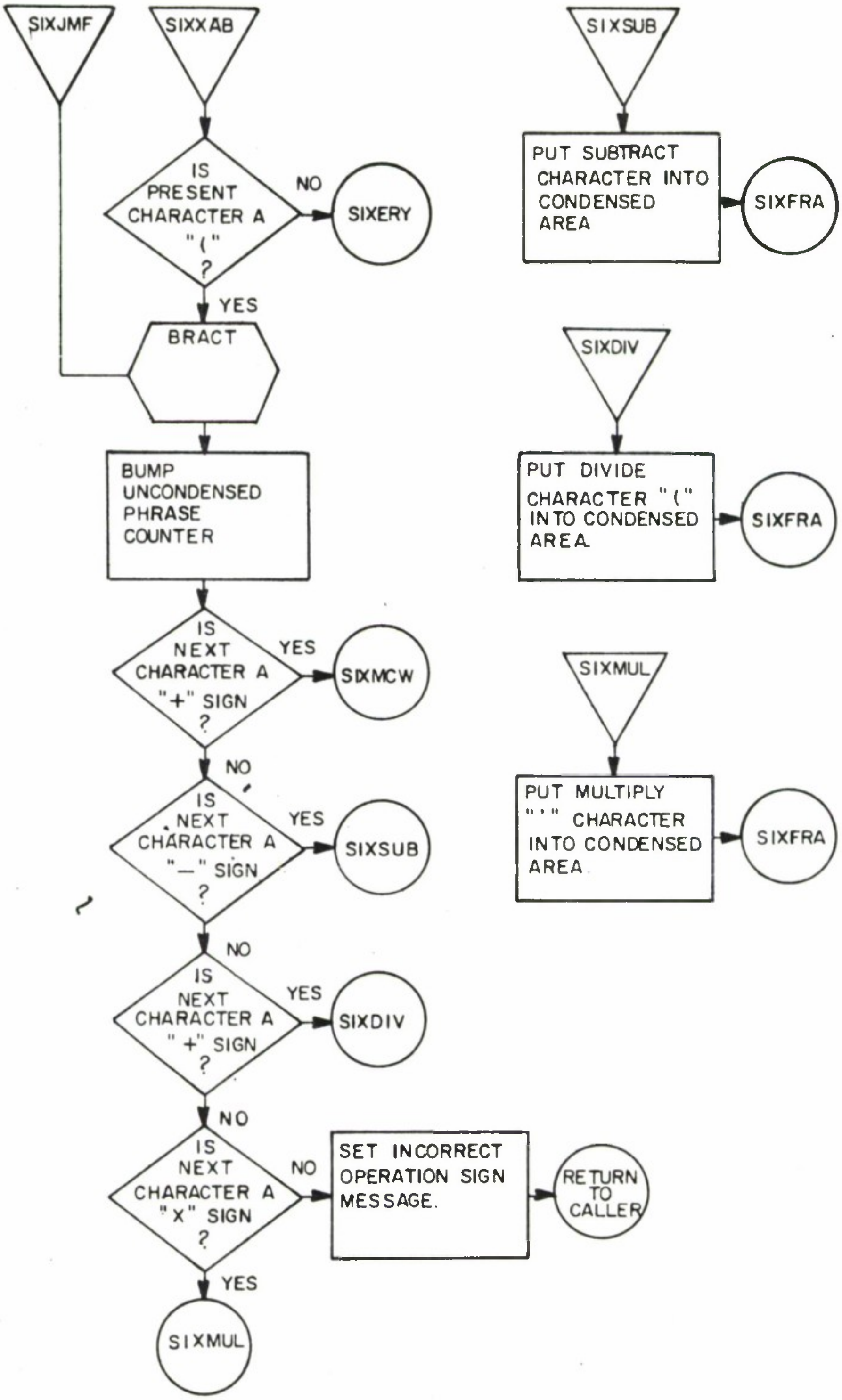
IB – 26,254


SIX
BLANK PHRASE AND CONDENSED PHRASE POINTERS
IS THIS A COLUMN PHRASE ?
YES
COL
NO
IS IT A COUNTER PHRASE ?
NO
ERROR MESSAGE "PHRASE TYPE NOT IDENTIFIED"
RETURN TO CALLER
YES
PUT A '6C' INTO CONDENSED AREA
MODIFY POINTERS
ISOLATE COUNTER VALUE. PUT INTO CONDENSED AREA
COUNT
CHECK FOR LEGAL COUNTER
CHECK FOR RIGHT OR LEFT JUSTIFICA-TION
NO
YES
IS CHARACTER AN "EQUALS" SIGN ?
NO
SET UP INCORRECT INDICATOR MESSAGE
RETURN TO CALLER
YES
INSERT A RIGHT-JUSTICATION CHARACTER INTO CONDENSED PHRASE
CONNEX
PUT JUSTIFICATION CHARACTER INTO CONDENSED PHRASE
IS NEXT CHARACTER AN "=" SIGN ?
YES
CONNEX
NO
SET ERROR MESSAGE FOR MISSING OR OUT OF PLACE "=" SIGN
RETURN TO CALLER

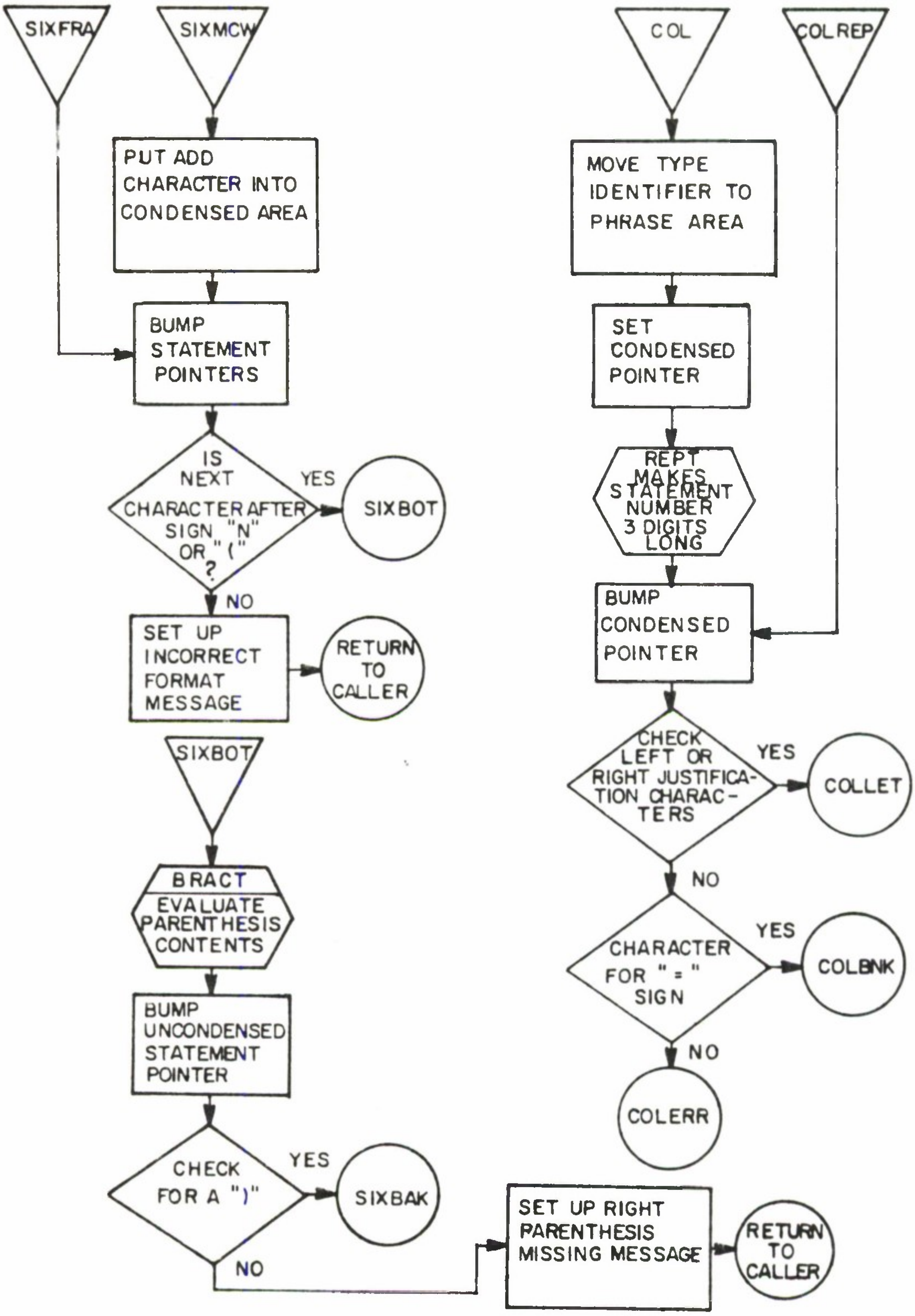
IB-26,255


SIXEXT
CONNEX
CONNNT
SIXBRK
DOES NEXT INDICATOR CHARACTER SPECIFY "N" ?
YES
SIXBRB
NO
DOES IT SPECIFY "D" ?
YES
SIXBRK
NO
DOES IT SPECIFY A "(" ?
NO
SET UP IMPROPER INDICATOR CHARACTER MESSAGE
RETURN TO CALLER
YES
ADVANCE POINTER TO START OF PAREN CONTENTS
BRACT
EVALUATE PARENTHESES CONTENTS
REPV
CHECK PHRASE SYNTAX
SIXBAK
EXIT FROM SIX
IS NEXT CHARACTER A "(" ?
YES
NO
CONNNT
SIXEXT
18-26,220

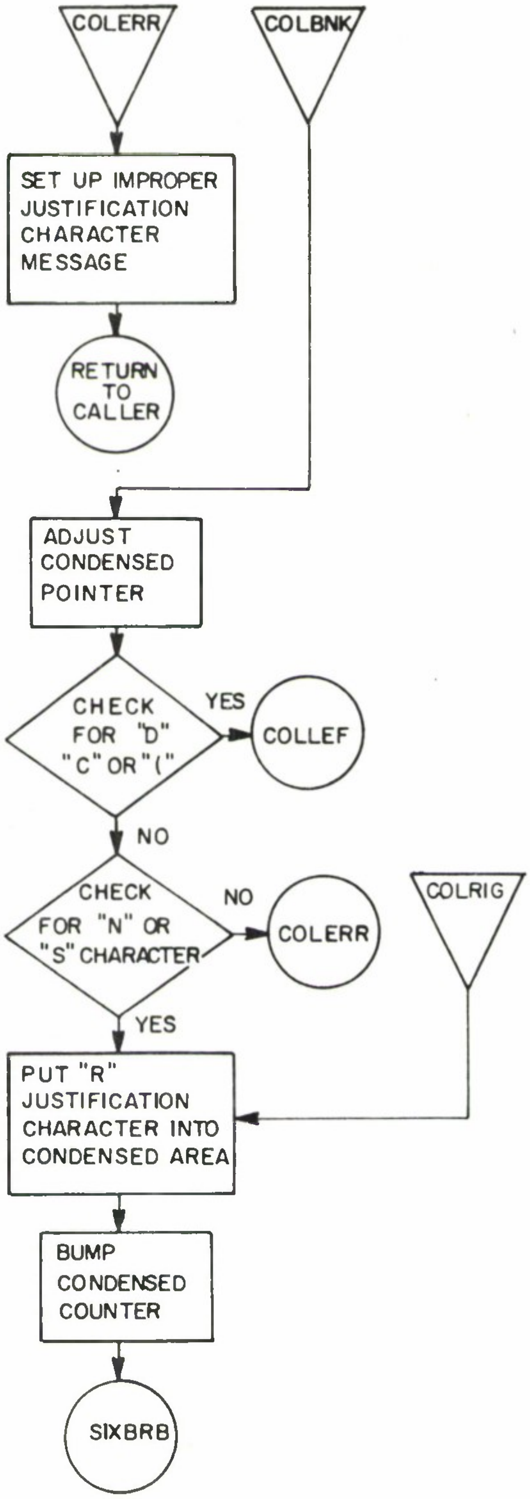

SIXHIP
SIXBRB
SEEIF
IS NEXT CHARACTER A "(" ?
NO
SET UP INCORRECT FORMAT MESSAGE
RETURN TO CALLER
YES
IS NEXT CHARACTER A "N" OR "." ?
YES
IS NEXT CHARACTER PLUS 1 A "(" ?
YES
NO
NO
IS NEXT CHARACTER A "(" ?
NO
SIXEXT
YES
PUT FIRST NUMERIC CHARACTER INTO CONDENSING AREA
BUMP PHRASE POINTERS
IS THIS CHARACTER AN "N" OR "(" ?
YES
SIXJMF
NO
SIXERY
IS THIS CHARACTER A "." ?
YES
SIXXAB
NO
SET UP INCORRECT FORMATTING MESSAGE
RETURN TO CALLER
IB-26,219


SIXJMF
SIXXAB
IS PRESENT CHARACTER A "(" ?
NO
SIXERY
YES
BRACT
BUMP UNCONDENSED PHRASE COUNTER
IS NEXT CHARACTER A "+" SIGN ?
YES
SIXMCW
NO
IS NEXT CHARACTER A "—" SIGN ?
YES
SIXSUB
NO
IS NEXT CHARACTER A "+" SIGN ?
YES
SIXDIV
NO
IS NEXT CHARACTER A "X" SIGN ?
NO
SET INCORRECT OPERATION SIGN MESSAGE.
RETURN TO CALLER
YES
SIXMUL
SIXSUB
PUT SUBTRACT CHARACTER INTO CONDENSED AREA
SIXFRA
SIXDIV
PUT DIVIDE CHARACTER "(" INTO CONDENSED AREA
SIXFRA
SIXMUL
PUT MULTIPLY "'" CHARACTER INTO CONDENSED AREA
SIXFRA

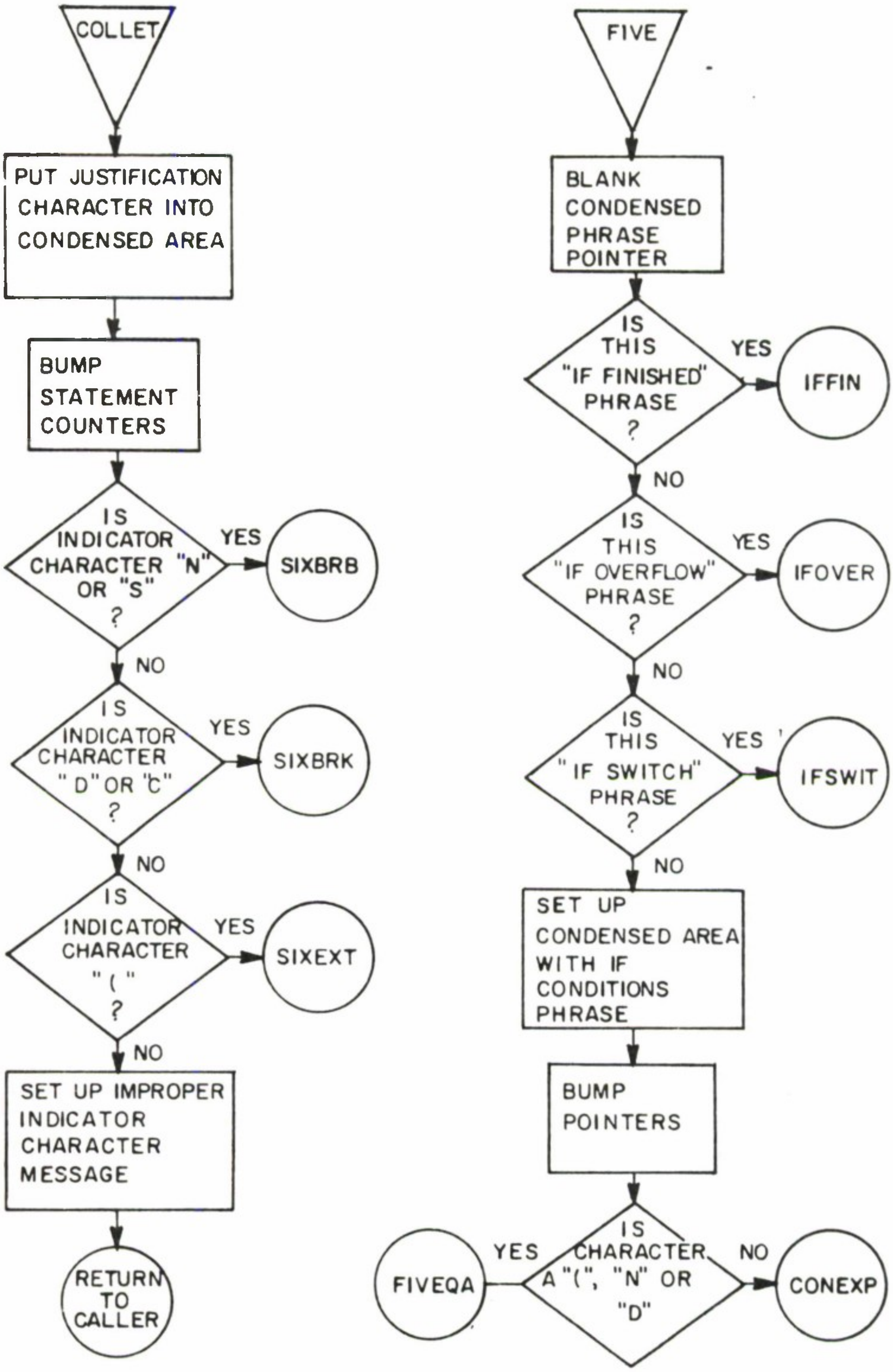
IB-26,256


SIXFRA
SIXMCW
PUT ADD CHARACTER INTO CONDENSED AREA
BUMP STATEMENT POINTERS
IS NEXT CHARACTER AFTER SIGN "N" OR "(" ?
YES
SIXBOT
NO
SET UP INCORRECT FORMAT MESSAGE
RETURN TO CALLER
SIXBOT
BRACT
EVALUATE PARENTHESIS CONTENTS
BUMP UNCONDENSED STATEMENT POINTER
CHECK FOR A ")"
YES
SIXBAK
NO
SET UP RIGHT PARENTHESIS MISSING MESSAGE
RETURN TO CALLER
COL
COLREP
MOVE TYPE IDENTIFIER TO PHRASE AREA
SET CONDENSED POINTER
REPT
MAKES STATEMENT NUMBER 3 DIGITS LONG
BUMP CONDENSED POINTER
CHECK LEFT OR RIGHT JUSTIFICATION CHARACTERS
YES
COLLET
NO
CHARACTER FOR "=" SIGN
YES
COLBNK
NO
COLERR

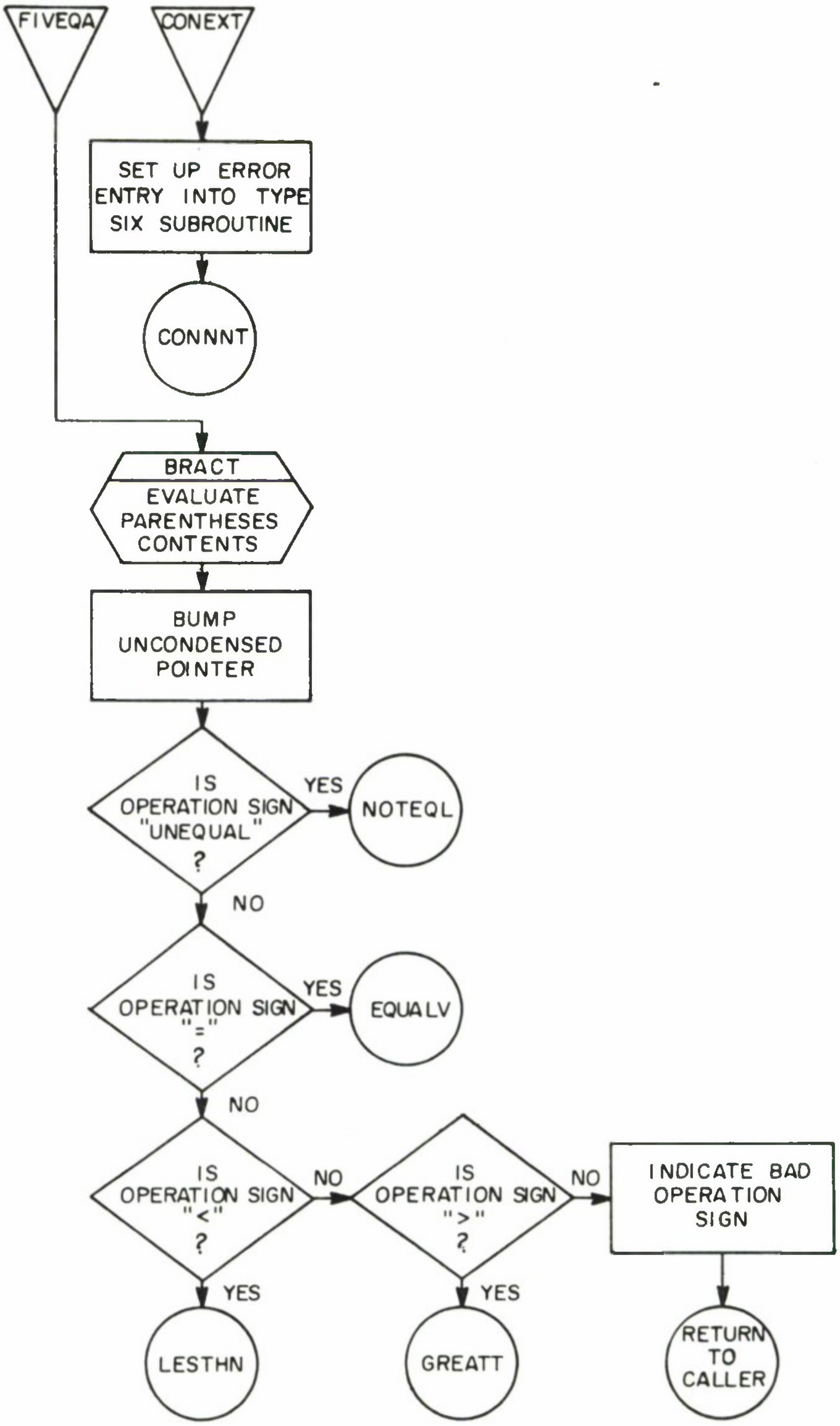
IB-26,257


COLERR
COLBNK
COLLEF
SET UP IMPROPER JUSTIFICATION CHARACTER MESSAGE
RETURN TO CALLER
PUT "L" JUSTIFICATION CHARACTER INTO CONDENSED AREA
BUMP CONDENSED COUNTER
SIXEXT
ADJUST CONDENSED POINTER
CHECK FOR "D" "C" OR "("
YES
COLLEF
NO
CHECK FOR "N" OR "S" CHARACTER
NO
COLERR
COLRIG
YES
PUT "R" JUSTIFICATION CHARACTER INTO CONDENSED AREA
BUMP CONDENSED COUNTER
SIXBRB

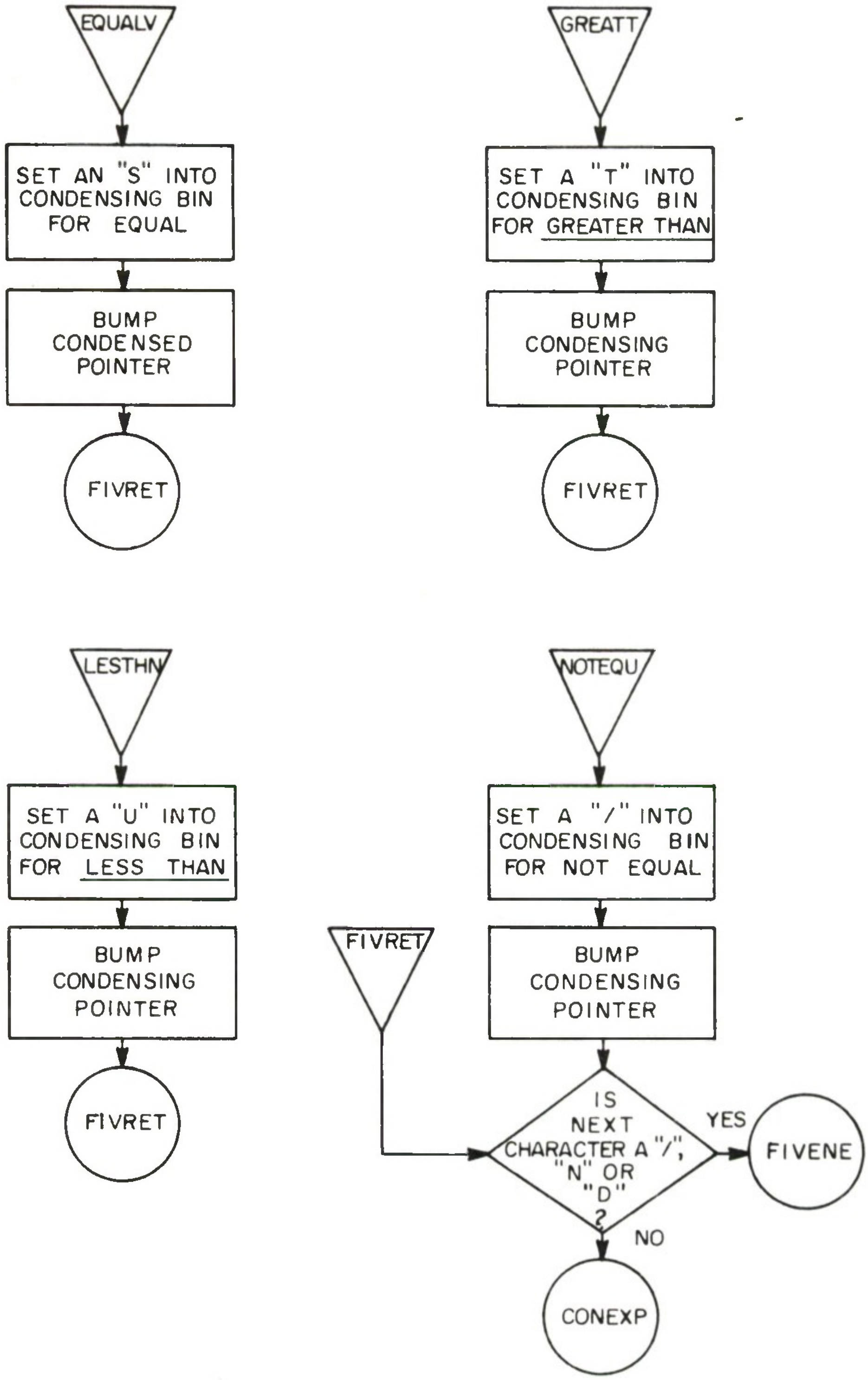
18-26,258


COLLET
PUT JUSTIFICATION CHARACTER INTO CONDENSED AREA
BUMP STATEMENT COUNTERS
IS INDICATOR CHARACTER "N" OR "S" ?
YES
SIXBRB
NO
IS INDICATOR CHARACTER "D" OR "C" ?
YES
SIXBRK
NO
IS INDICATOR CHARACTER "(" ?
YES
SIXEXT
NO
SET UP IMPROPER INDICATOR CHARACTER MESSAGE
RETURN TO CALLER
FIVE
BLANK CONDENSED PHRASE POINTER
IS THIS "IF FINISHED" PHRASE ?
YES
IFFIN
NO
IS THIS "IF OVERFLOW" PHRASE ?
YES
IFOVER
NO
IS THIS "IF SWITCH" PHRASE ?
YES
IFSWIT
NO
SET UP CONDENSED AREA WITH IF CONDITIONS PHRASE
BUMP POINTERS
IS CHARACTER A "(", "N" OR "D"
YES
FIVEQA
NO
CONEXP

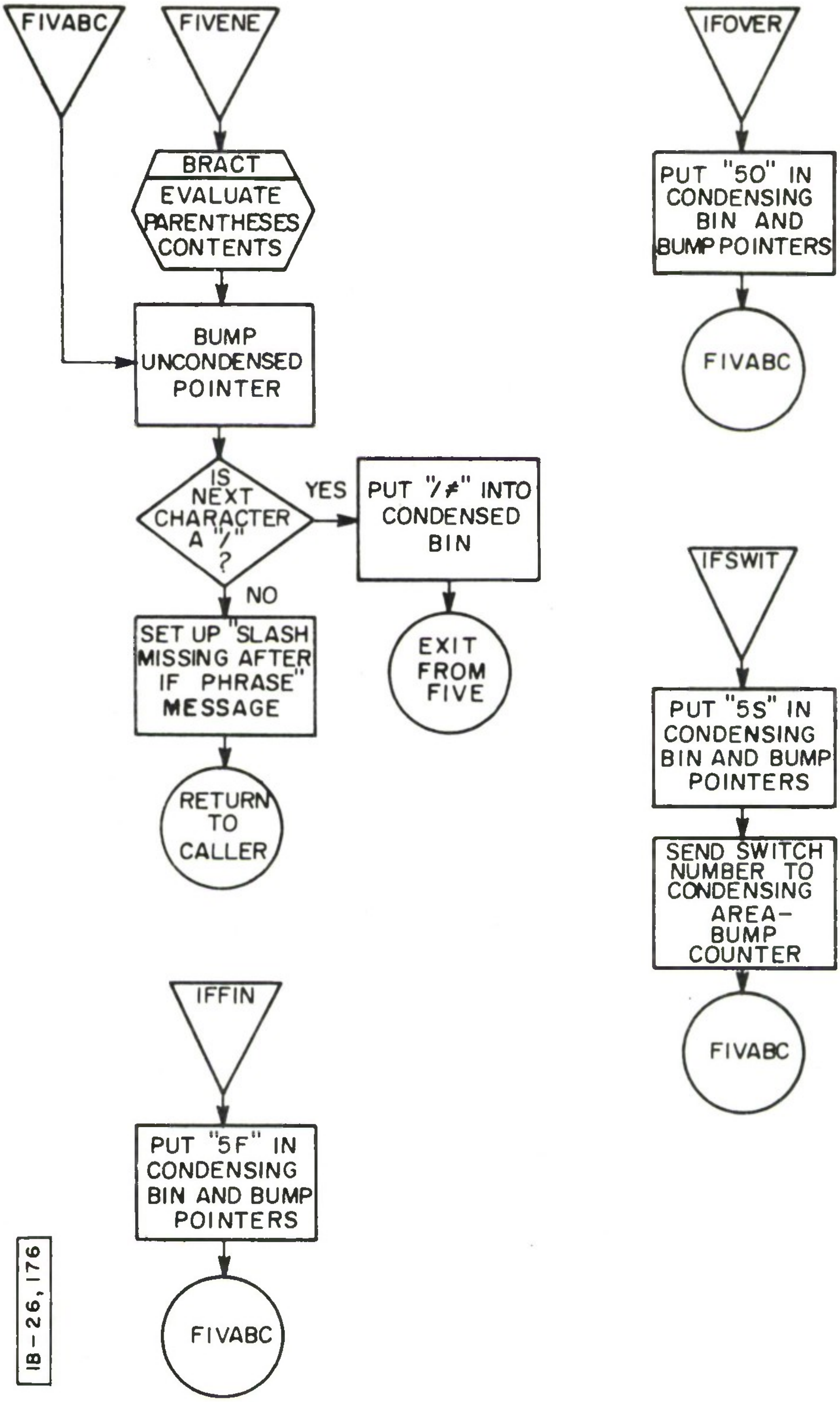
IB-26,259


FIVEQA
CONEXT
SET UP ERROR ENTRY INTO TYPE SIX SUBROUTINE
CONNNT
BRACT
EVALUATE PARENTHESES CONTENTS
BUMP UNCONDENSED POINTER
IS OPERATION SIGN "UNEQUAL" ?
YES
NOTEQL
NO
IS OPERATION SIGN "=" ?
YES
EQUALV
NO
IS OPERATION SIGN "<" ?
NO
IS OPERATION SIGN ">" ?
NO
INDICATE BAD OPERATION SIGN
YES
LESTHN
YES
GREATT
RETURN TO CALLER

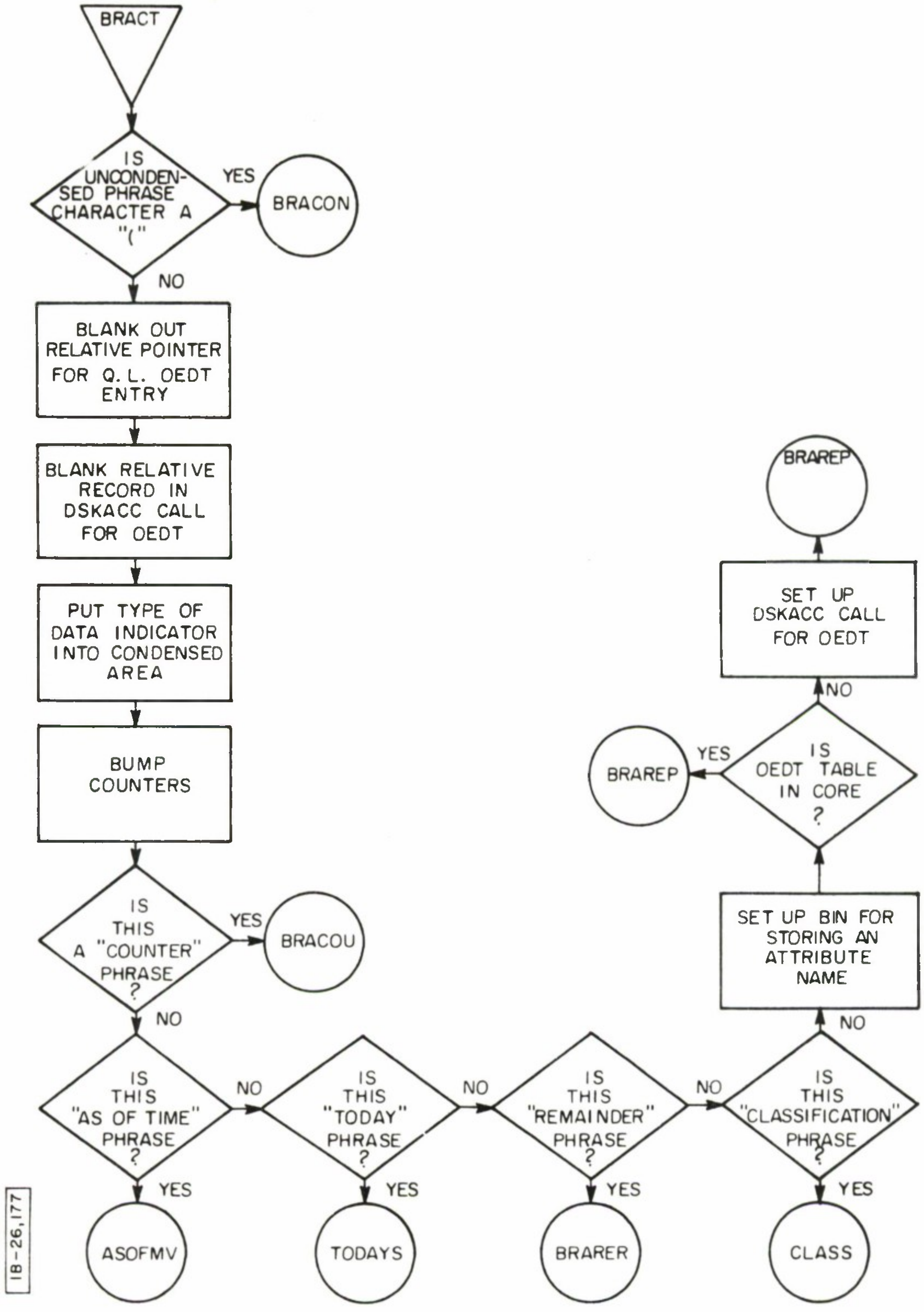
18-26,227

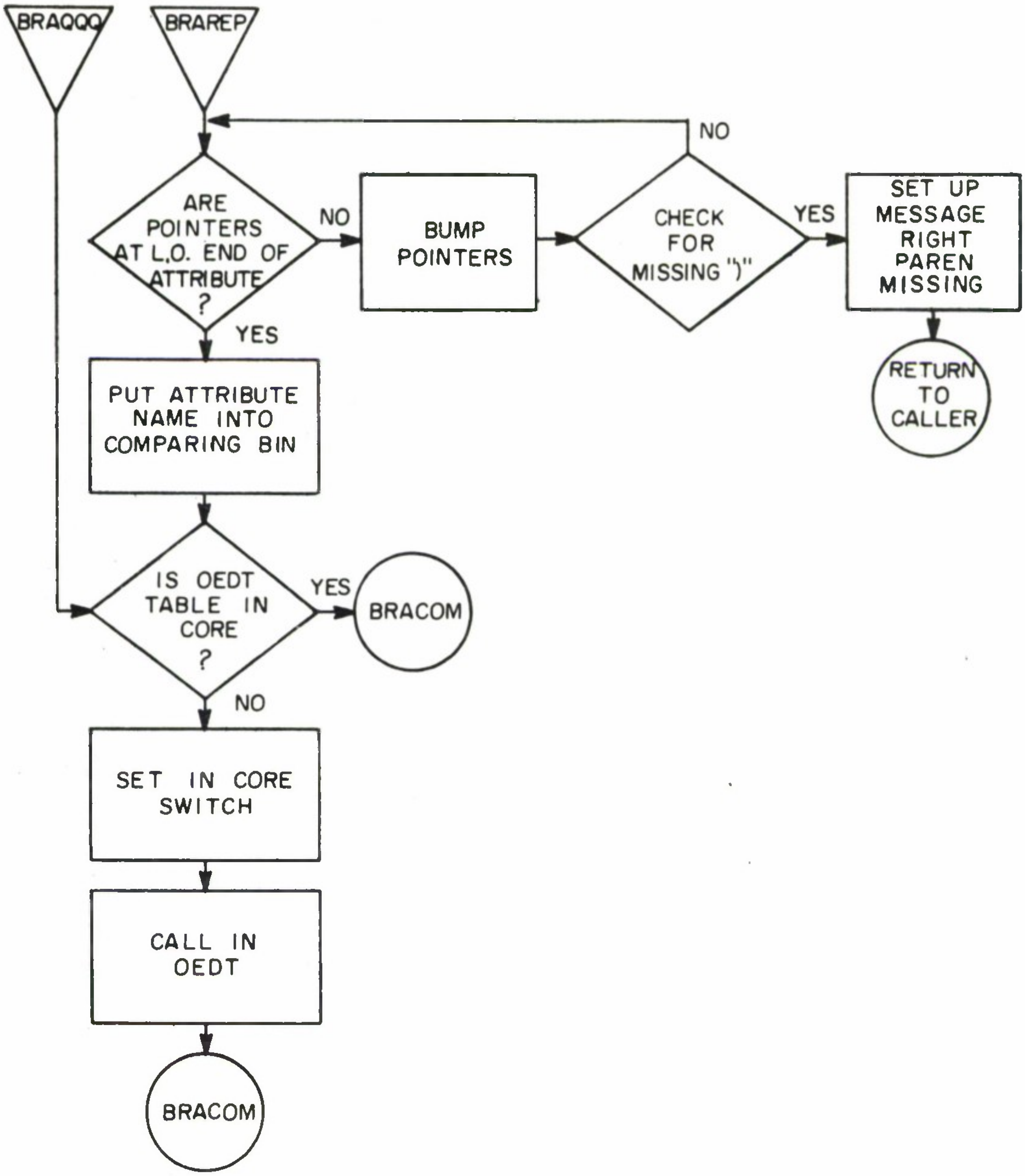

EQUALV
SET AN "S" INTO CONDENSING BIN FOR EQUAL
BUMP CONDENSED POINTER
FIVRET
GREATT
SET A "T" INTO CONDENSING BIN FOR GREATER THAN
BUMP CONDENSING POINTER
FIVRET
LESTHN
SET A "U" INTO CONDENSING BIN FOR LESS THAN
BUMP CONDENSING POINTER
FIVRET
NOTEQU
SET A "/" INTO CONDENSING BIN FOR NOT EQUAL
BUMP CONDENSING POINTER
FIVRET
IS NEXT CHARACTER A "/", "N" OR "D" ?
YES
FIVENE
NO
CONEXP

IB-26,228

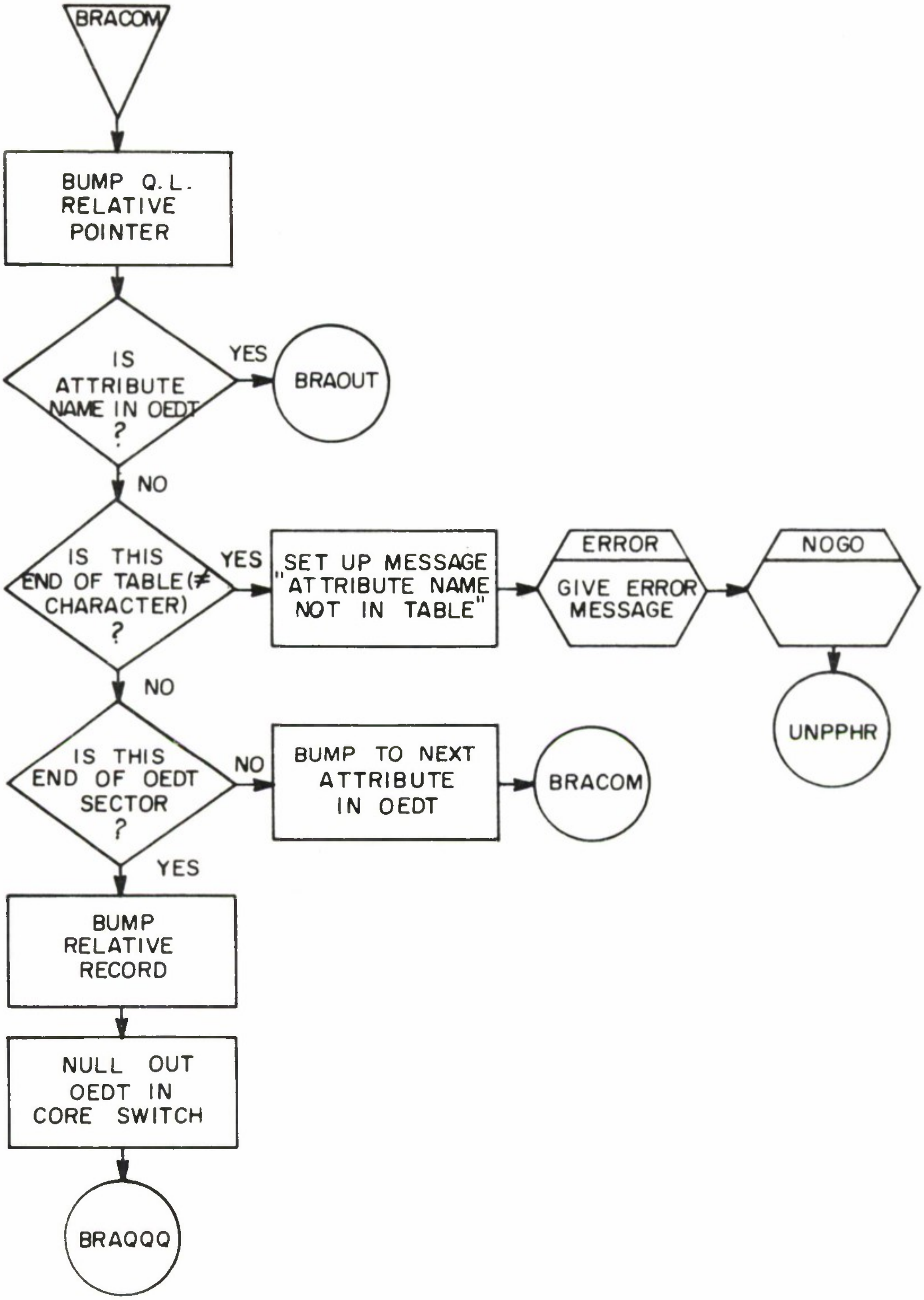

FIVABC
FIVENE
BRACT
EVALUATE PARENTHESES CONTENTS
BUMP UNCONDENSED POINTER
IS NEXT CHARACTER A "/" ?
YES
PUT "/≠" INTO CONDENSED BIN
EXIT FROM FIVE
NO
SET UP "SLASH MISSING AFTER IF PHRASE" MESSAGE
RETURN TO CALLER
IFFIN
PUT "5F" IN CONDENSING BIN AND BUMP POINTERS
FIVABC
IFOVER
PUT "5O" IN CONDENSING BIN AND BUMP POINTERS
FIVABC
IFSWIT
PUT "5S" IN CONDENSING BIN AND BUMP POINTERS
SEND SWITCH NUMBER TO CONDENSING AREA-BUMP COUNTER
FIVABC

IB-26,176

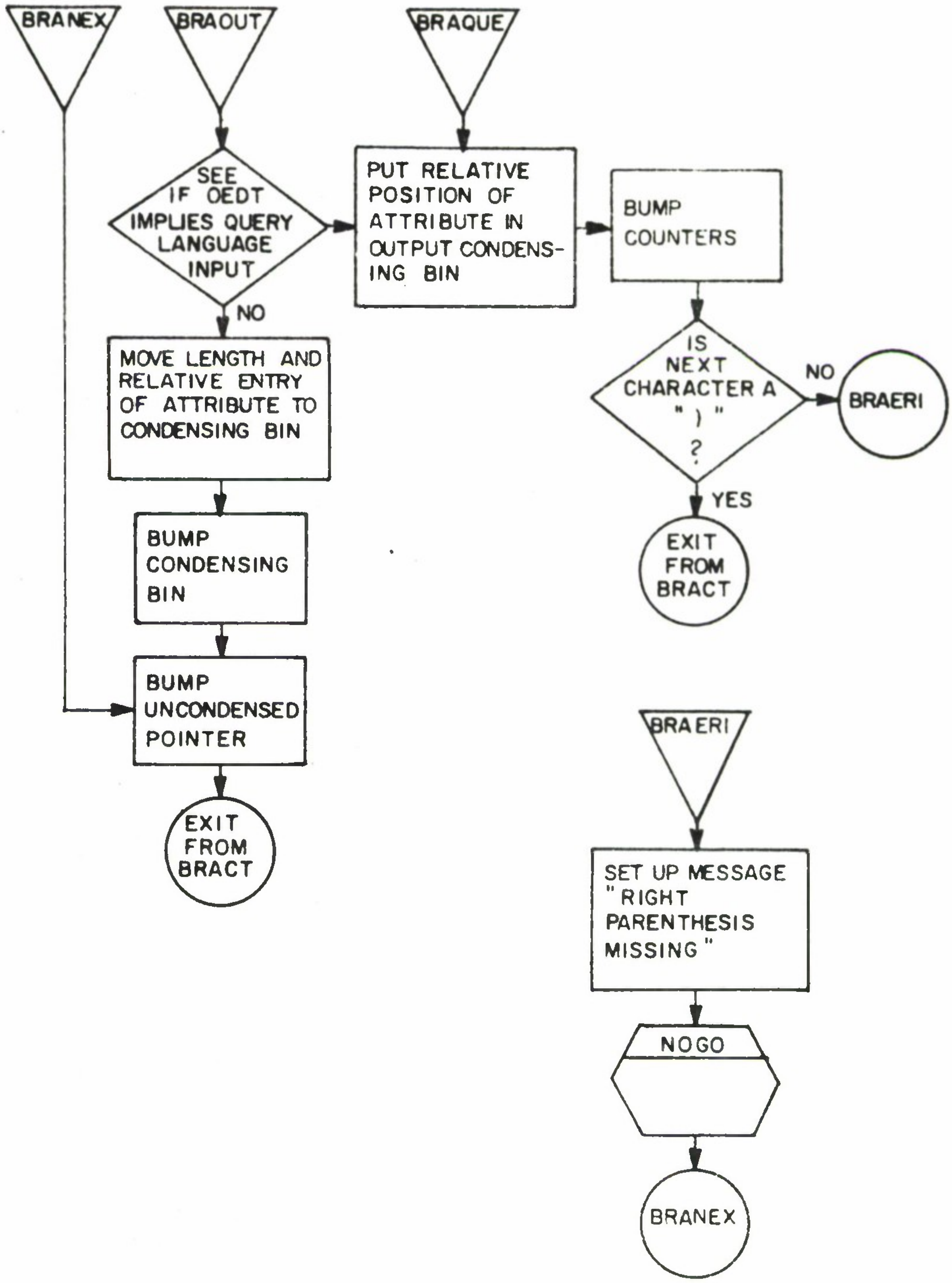

BRACT
IS UNCONDENSED PHRASE CHARACTER A "("
YES
BRACON
NO
BLANK OUT RELATIVE POINTER FOR Q.L. OEDT ENTRY
BLANK RELATIVE RECORD IN DSKACC CALL FOR OEDT
PUT TYPE OF DATA INDICATOR INTO CONDENSED AREA
BUMP COUNTERS
IS THIS A "COUNTER" PHRASE ?
YES
BRACOU
NO
IS THIS "AS OF TIME" PHRASE ?
YES
ASOFMV
NO
IS THIS "TODAY" PHRASE ?
YES
TODAYS
NO
IS THIS "REMAINDER" PHRASE ?
YES
BRARER
NO
IS THIS "CLASSIFICATION" PHRASE ?
YES
CLASS
NO
SET UP BIN FOR STORING AN ATTRIBUTE NAME
IS OEDT TABLE IN CORE ?
YES
BRAREP
NO
SET UP DSKACC CALL FOR OEDT
BRAREP
IB-26,177


BRAQQQ
BRAREP
ARE POINTERS AT L.O. END OF ATTRIBUTE ?
NO
BUMP POINTERS
CHECK FOR MISSING ")"
NO
YES
SET UP MESSAGE RIGHT PAREN MISSING
RETURN TO CALLER
YES
PUT ATTRIBUTE NAME INTO COMPARING BIN
IS OEDT TABLE IN CORE ?
YES
BRACOM
NO
SET IN CORE SWITCH
CALL IN OEDT
BRACOM

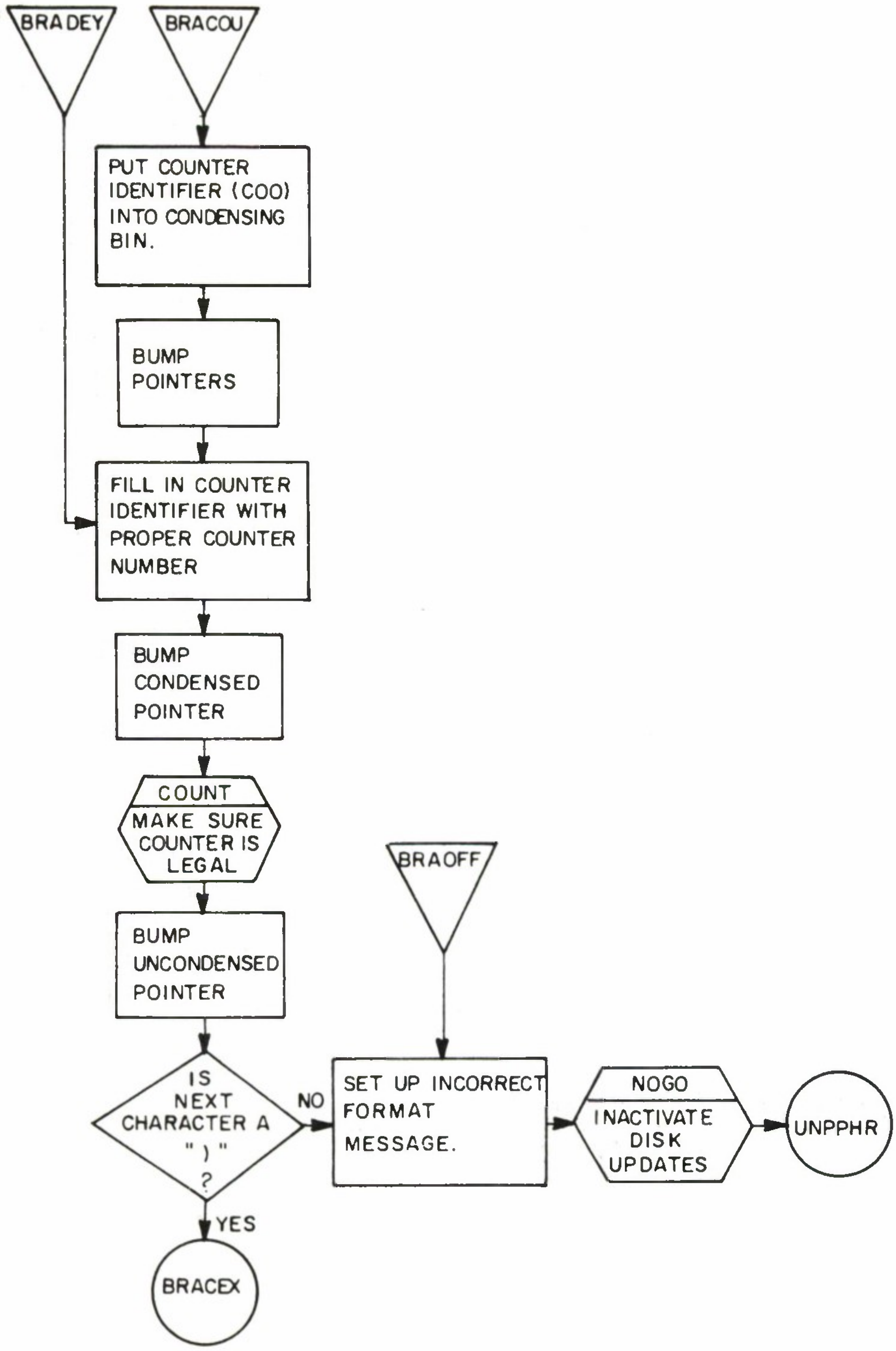
IB-26,215


BRACOM
BUMP Q.L. RELATIVE POINTER
IS ATTRIBUTE NAME IN OEDT ?
YES
BRAOUT
NO
IS THIS END OF TABLE(≠ CHARACTER) ?
YES
SET UP MESSAGE "ATTRIBUTE NAME NOT IN TABLE"
ERROR
GIVE ERROR MESSAGE
NOGO
UNPPHR
NO
IS THIS END OF OEDT SECTOR ?
NO
BUMP TO NEXT ATTRIBUTE IN OEDT
BRACOM
YES
BUMP RELATIVE RECORD
NULL OUT OEDT IN CORE SWITCH
BRAQQQ

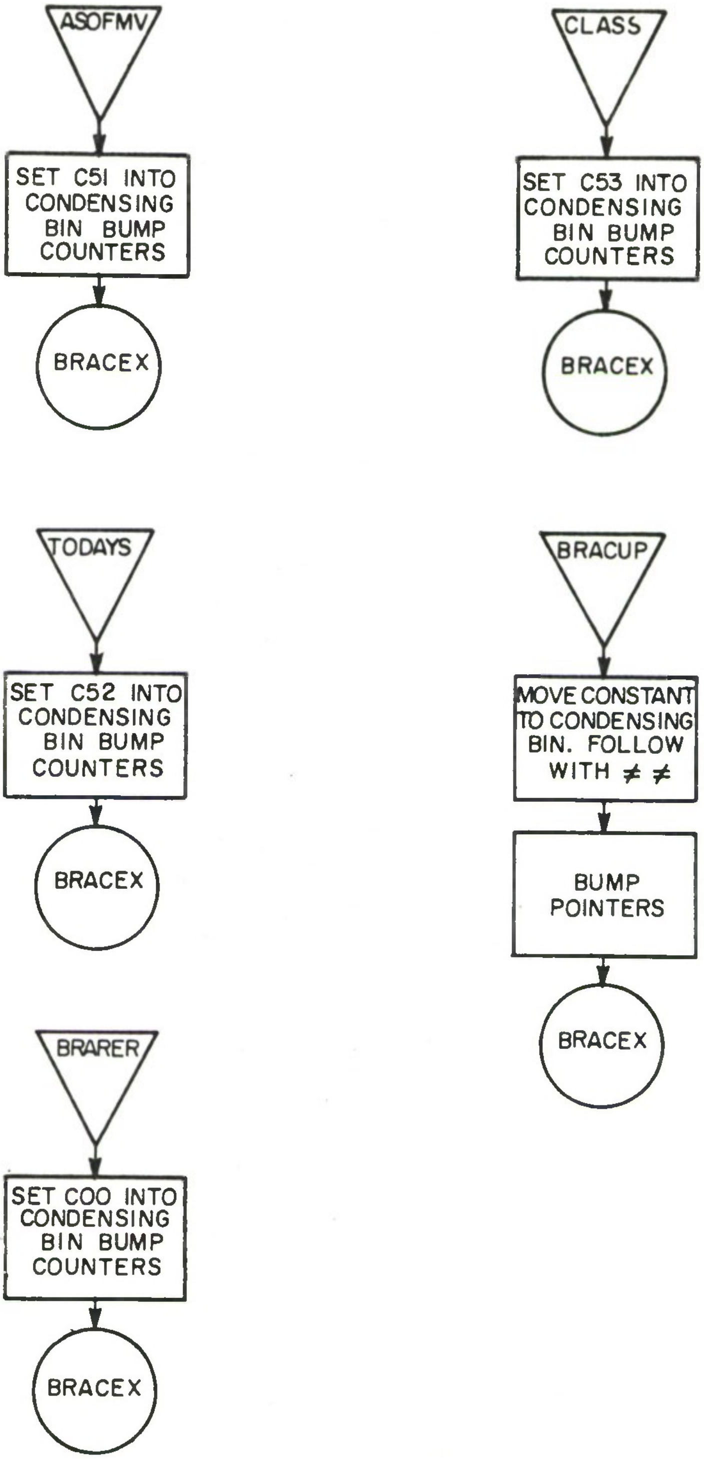
18-26,216


BRANEX
BRAOUT
BRAQUE
SEE IF OEDT IMPLIES QUERY LANGUAGE INPUT
NO
PUT RELATIVE POSITION OF ATTRIBUTE IN OUTPUT CONDENSING BIN
BUMP COUNTERS
MOVE LENGTH AND RELATIVE ENTRY OF ATTRIBUTE TO CONDENSING BIN
IS NEXT CHARACTER A ")" ?
NO
BRAERI
YES
EXIT FROM BRACT
BUMP CONDENSING BIN
BUMP UNCONDENSED POINTER
EXIT FROM BRACT
BRAERI
SET UP MESSAGE "RIGHT PARENTHESIS MISSING"
NOGO
BRANEX

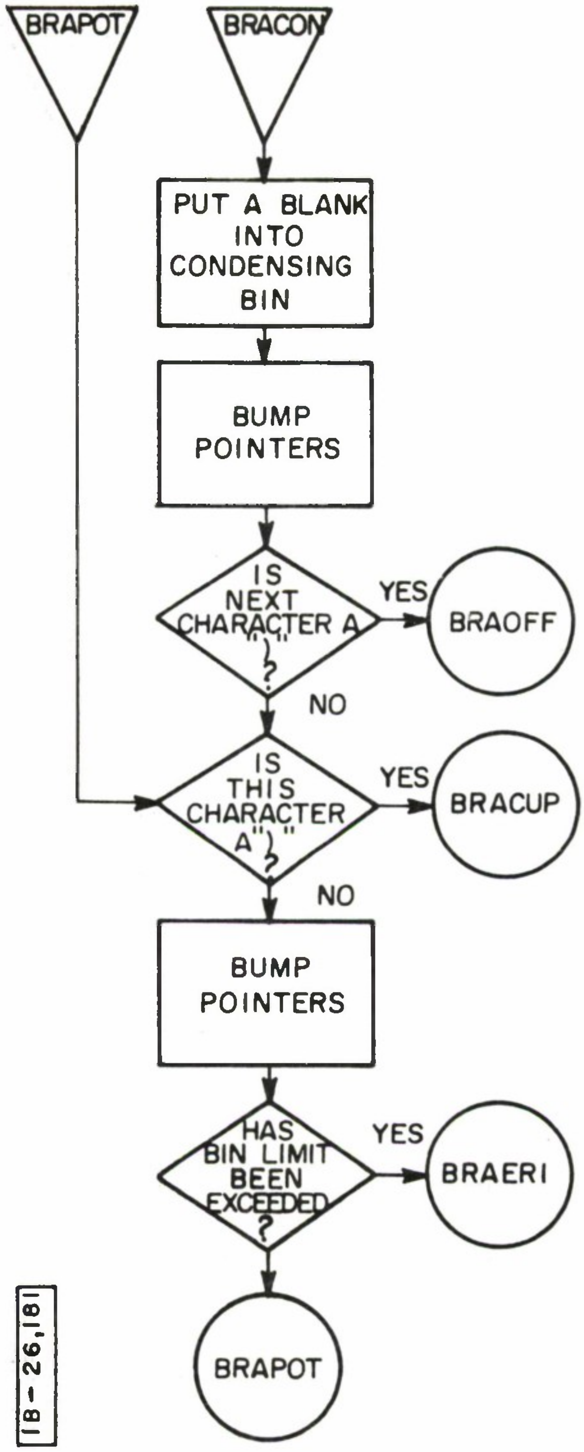
18-26,260


BRADEY
BRACOU
PUT COUNTER IDENTIFIER (COO) INTO CONDENSING BIN.
BUMP POINTERS
FILL IN COUNTER IDENTIFIER WITH PROPER COUNTER NUMBER
BUMP CONDENSED POINTER
COUNT
MAKE SURE COUNTER IS LEGAL
BRAOFF
BUMP UNCONDENSED POINTER
IS NEXT CHARACTER A ")" ?
NO
YES
SET UP INCORRECT FORMAT MESSAGE.
NOGO
INACTIVATE DISK UPDATES
UNPPHR
BRACEX

1B-26,261


ASOFMV
SET C51 INTO CONDENSING BIN BUMP COUNTERS
BRACEX
CLASS
SET C53 INTO CONDENSING BIN BUMP COUNTERS
BRACEX
TODAYS
SET C52 INTO CONDENSING BIN BUMP COUNTERS
BRACEX
BRACUP
MOVE CONSTANT TO CONDENSING BIN. FOLLOW WITH ≠ ≠
BUMP POINTERS
BRACEX
BRARER
SET COO INTO CONDENSING BIN BUMP COUNTERS
BRACEX

18-26,180


BRAPOT
BRACON
PUT A BLANK INTO CONDENSING BIN
BUMP POINTERS
IS NEXT CHARACTER A ")"?
YES
BRAOFF
NO
IS THIS CHARACTER A")"?
YES
BRACUP
NO
BUMP POINTERS
HAS BIN LIMIT BEEN EXCEEDED?
YES
BRAERI
BRAPOT

IB-26,181

REPGEN

Purpose

REPGEN is operated to produce a Serial File Tape from a source data file contained on cards, magnetic tape or on the system disk. The Serial File Tape is then used as input to the output phase of the Report Generation capability.

Calling Sequence

Enter on the console typewriter

REPGENrrrfcn

where rrr = 3-digit report number (no blanks)

f = Format number of the Output Program to be executed. Must be 1-7 or blank.

c = Classification code: U, C, S or T for Unclassified, Confidential, Secret or Top Secret.

n = Number of copies of printed output. Must be 1-9 or blank.

Final return will be made to IDLOOP.

Description

The Report Descriptor Table is read from disk and the data from the typewriter call is stored away. If no report number was given with the call, the PRTIND routine is operated, printing an index of REPGEN reports followed by a return to IDLOOP.

If the report number was not blank, the requested report is located in the Report Descriptor Table, mount messages are issued and the input medium is checked.

If the input medium is disk the saved Query Language statement number is taken from the Report Descriptor Table entry and used to construct a dummy call to REPEET. REPEET is called via COPS; it fetches the Query Language statement, types it on the console and passes control to GETTAC. GETTAC executes the Query Language statement, placing the retrieved data on a Serial File Tape. The option is now given to the operator to enter the output phase or to terminate.

If the input medium is not disk, the Record Descriptor Table is brought in from disk and a source data record (either card or tape) is read. Entries in the Record Descriptor Table are examined against the input record until the record type is identified, at which time the Attribute Descriptor Table for this record type is read into core.

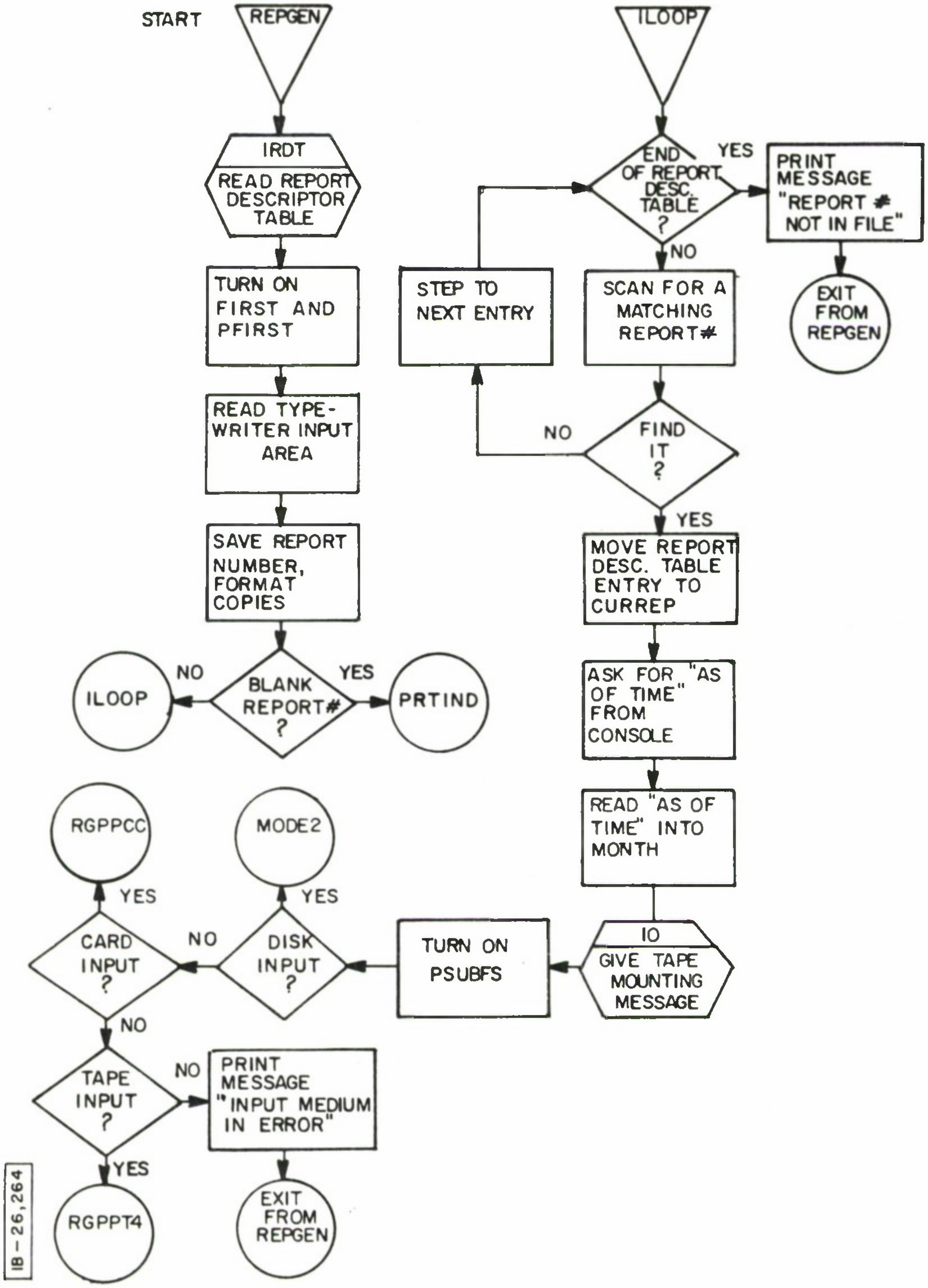
All control attributes are collected into a control field for this input record. Sub-control attributes are collected into a sub-control field.

The control field is compared with the previous control field. If they differ, a control break is noted. If the sub-control field differs from the previous sub-control field, a sub-control break is noted. If the current record caused no control or sub-control break, the record is given at once to the audit routine.

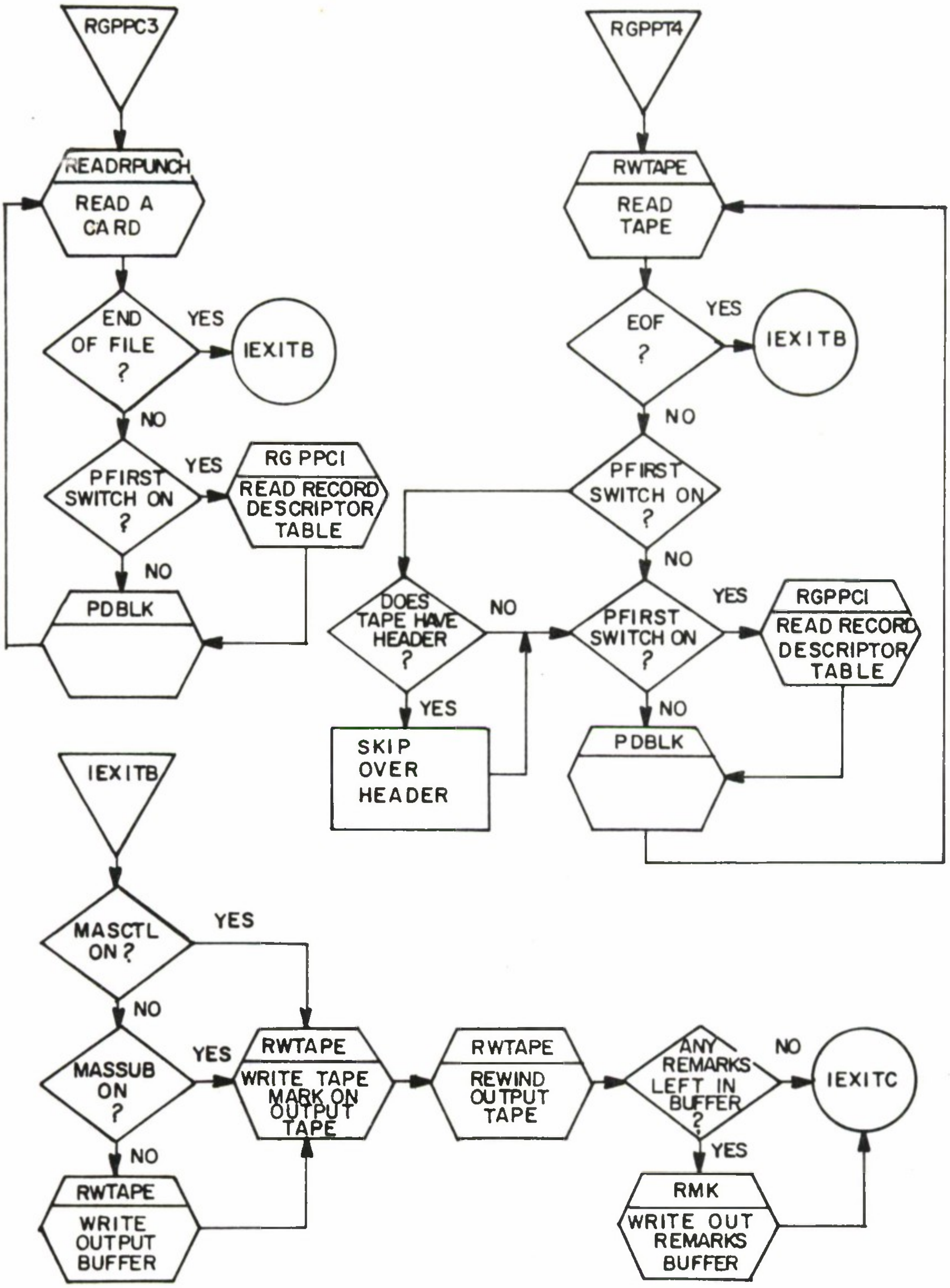
The audit routine processes each attribute of the input record separately, performing left or right justification and legality checks as required or specified. Each attribute, after it is audited, results in an entry in the DAWORK Table. The entry in DAWORK specifies the location of the attribute in the input area, its destination in the output area, and its length. If the input record contained sub-control attributes, a similar entry is made in the DAWOR2 table. After all input attributes for the current record have been audited, the DAWORK table is used to move the audited fields from the input area to the output buffer. The next input record is then read.

At the point where control and sub-control fields were checked with previous fields, if a control break or sub-control break was noted the current contents of the output buffer will be written on the Serial File Tape. If a control break occurred, the entire output image is then cleared. If a sub-control break occurred, the DAWOR2 table is used to determine those fields to selectively clear in the output image. After the output image is cleared, the current input record is given to the audit routine, and processing continues as described before.

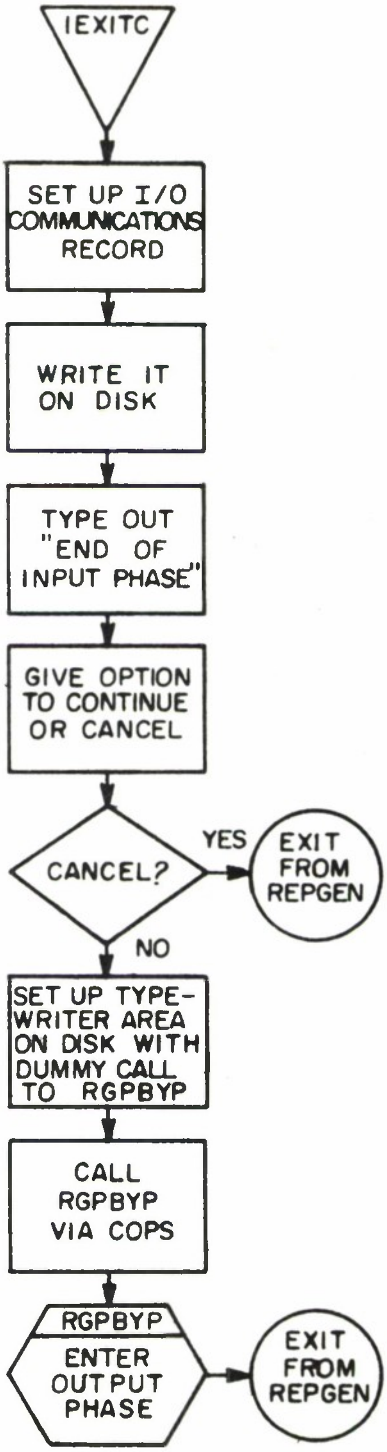
When the input data file has been fully processed and the Serial File Tape is complete, a disk record called the I/O Communication Record is written, containing the current Report Descriptor Table entry and various typewriter inputs (e.g., Report Classification Code, Format Number, AS OF TIME) to be passed along to the output phase. The operator is given a typewriter option to cancel or to continue into the output phase. If the output phase is requested, a dummy call is set up to RGPBYP. Control will eventually be returned from the output phase to REPGEN which will exit to IDLOOP.


START
REPGEN
IRDT
READ REPORT DESCRIPTOR TABLE
TURN ON FIRST AND PFIRST
READ TYPE-WRITER INPUT AREA
SAVE REPORT NUMBER, FORMAT COPIES
BLANK REPORT# ?
NO
YES
ILOOP
PRTIND
ILOOP
END OF REPORT DESC. TABLE ?
YES
PRINT MESSAGE "REPORT # NOT IN FILE"
EXIT FROM REPGEN
NO
STEP TO NEXT ENTRY
SCAN FOR A MATCHING REPORT#
FIND IT ?
NO
YES
MOVE REPORT DESC. TABLE ENTRY TO CURREP
ASK FOR "AS OF TIME" FROM CONSOLE
READ "AS OF TIME" INTO MONTH
IO
GIVE TAPE MOUNTING MESSAGE
TURN ON PSUBFS
DISK INPUT ?
YES
MODE2
NO
CARD INPUT ?
YES
RGPPCC
NO
TAPE INPUT ?
YES
RGPPT4
NO
PRINT MESSAGE "INPUT MEDIUM IN ERROR"
EXIT FROM REPGEN

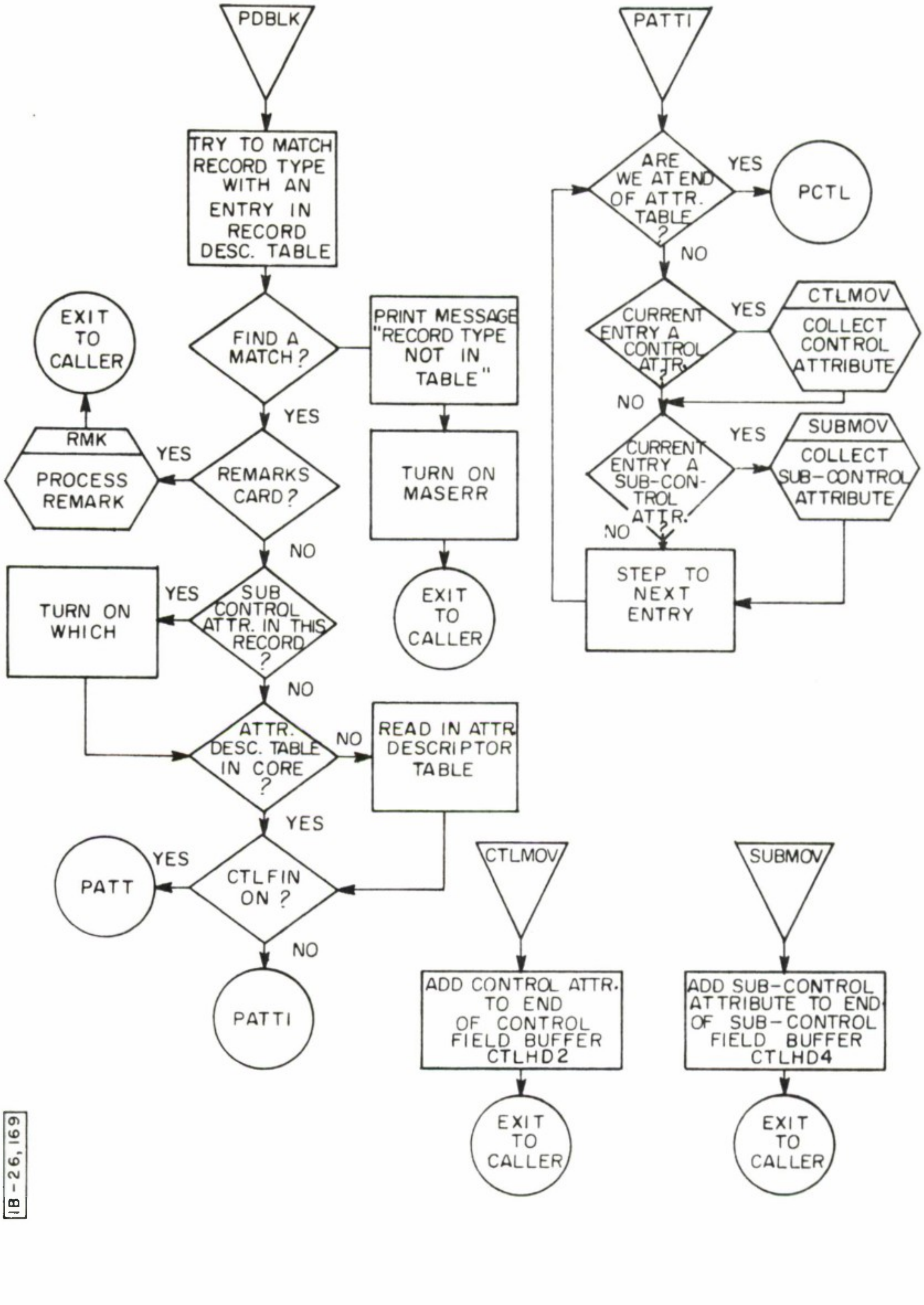
IB-26,264

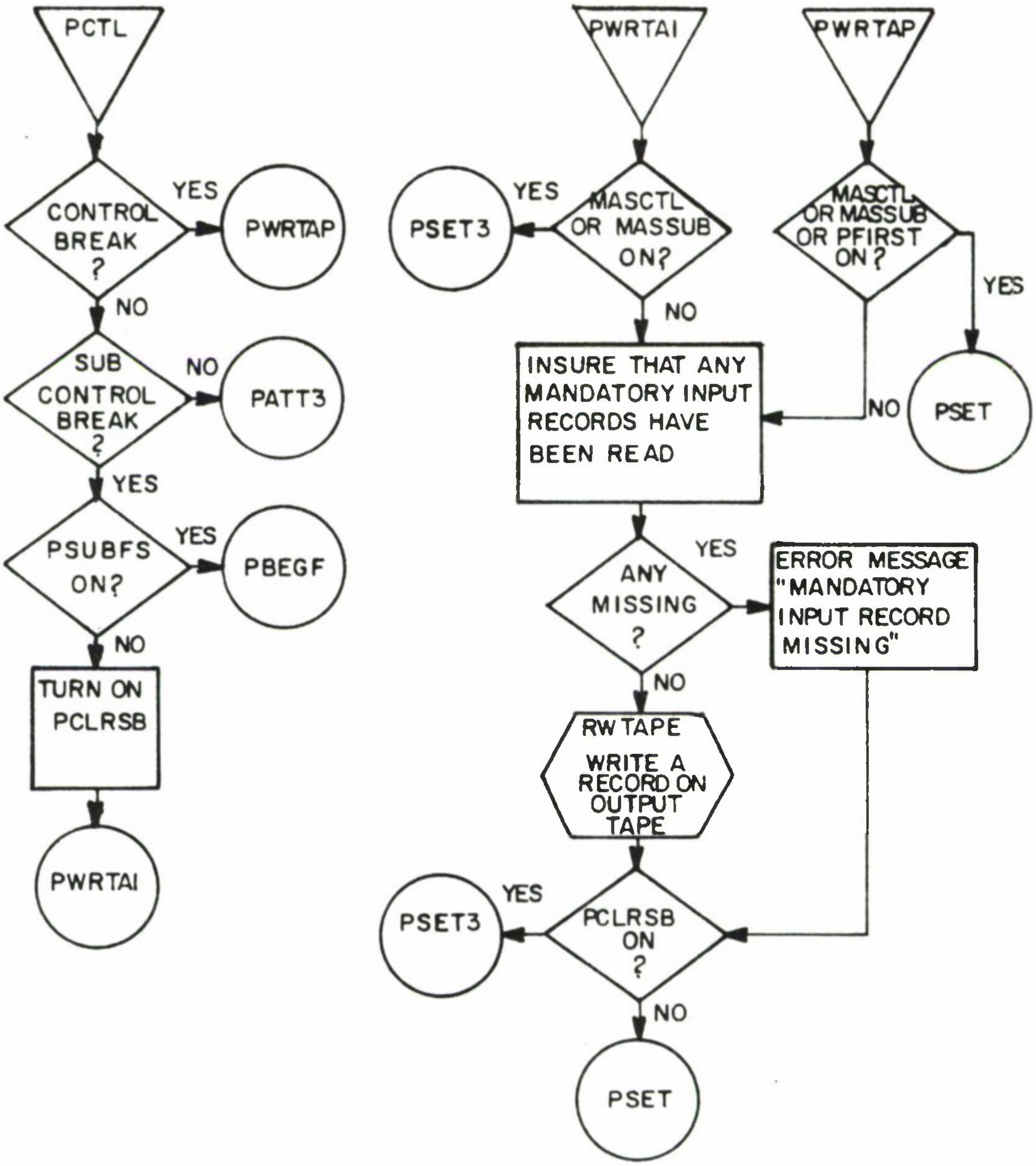

RGPPC3
READRPUNCH
READ A CARD
END OF FILE ?
YES
IEXITB
NO
PFIRST SWITCH ON ?
YES
RG PPCI
READ RECORD DESCRIPTOR TABLE
NO
PDBLK
RGPPT4
RWTAPE
READ TAPE
EOF ?
YES
IEXITB
NO
PFIRST SWITCH ON ?
NO
DOES TAPE HAVE HEADER ?
NO
YES
SKIP OVER HEADER
PFIRST SWITCH ON ?
YES
RGPPCI
READ RECORD DESCRIPTOR TABLE
NO
PDBLK
IEXITB
MASCTL ON ?
YES
NO
MASSUB ON ?
YES
NO
RWTAPE
WRITE OUTPUT BUFFER
RWTAPE
WRITE TAPE MARK ON OUTPUT TAPE
RWTAPE
REWIND OUTPUT TAPE
ANY REMARKS LEFT IN BUFFER ?
NO
IEXITC
YES
RMK
WRITE OUT REMARKS BUFFER

IB-26,265

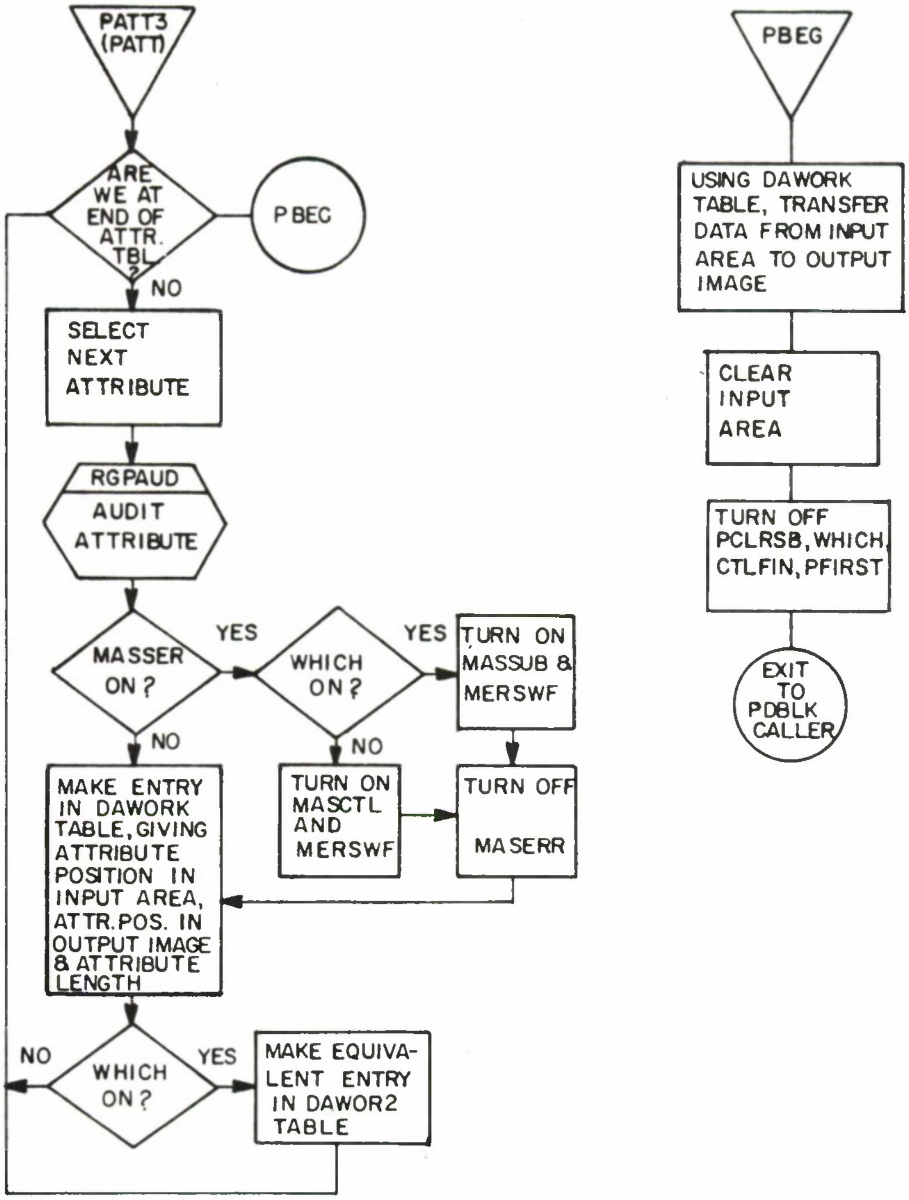

IEXITC
SET UP I/O COMMUNICATIONS RECORD
WRITE IT ON DISK
TYPE OUT "END OF INPUT PHASE"
GIVE OPTION TO CONTINUE OR CANCEL
CANCEL?
YES
EXIT FROM REPGEN
NO
SET UP TYPE-WRITER AREA ON DISK WITH DUMMY CALL TO RGPBYP
CALL RGPBYP VIA COPS
RGPBYP
ENTER OUTPUT PHASE
EXIT FROM REPGEN

18-26,168

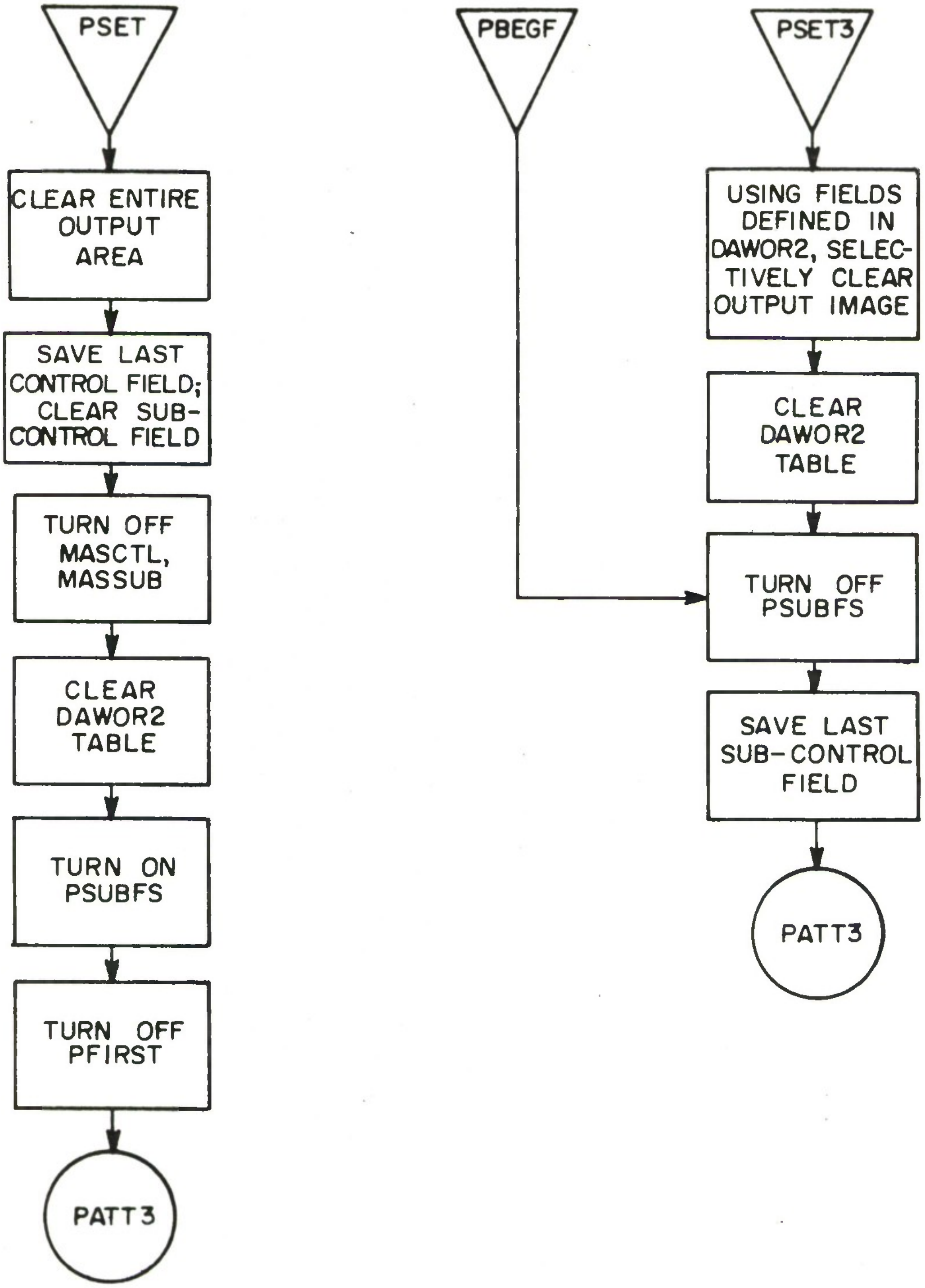

PDBLK
TRY TO MATCH RECORD TYPE WITH AN ENTRY IN RECORD DESC. TABLE
FIND A MATCH?
PRINT MESSAGE "RECORD TYPE NOT IN TABLE"
TURN ON MASERR
EXIT TO CALLER
YES
REMARKS CARD?
YES
RMK
PROCESS REMARK
EXIT TO CALLER
NO
SUB CONTROL ATTR. IN THIS RECORD?
YES
TURN ON WHICH
NO
ATTR. DESC. TABLE IN CORE?
NO
READ IN ATTR DESCRIPTOR TABLE
YES
CTLFIN ON?
YES
PATT
NO
PATTI
PATTI
ARE WE AT END OF ATTR. TABLE?
YES
PCTL
NO
CURRENT ENTRY A CONTROL ATTR?
YES
CTLMOV
COLLECT CONTROL ATTRIBUTE
NO
CURRENT ENTRY A SUB-CONTROL ATTR.?
YES
SUBMOV
COLLECT SUB-CONTROL ATTRIBUTE
NO
STEP TO NEXT ENTRY
CTLMOV
ADD CONTROL ATTR. TO END OF CONTROL FIELD BUFFER CTLHD2
EXIT TO CALLER
SUBMOV
ADD SUB-CONTROL ATTRIBUTE TO END OF SUB-CONTROL FIELD BUFFER CTLHD4
EXIT TO CALLER
IB-26,169

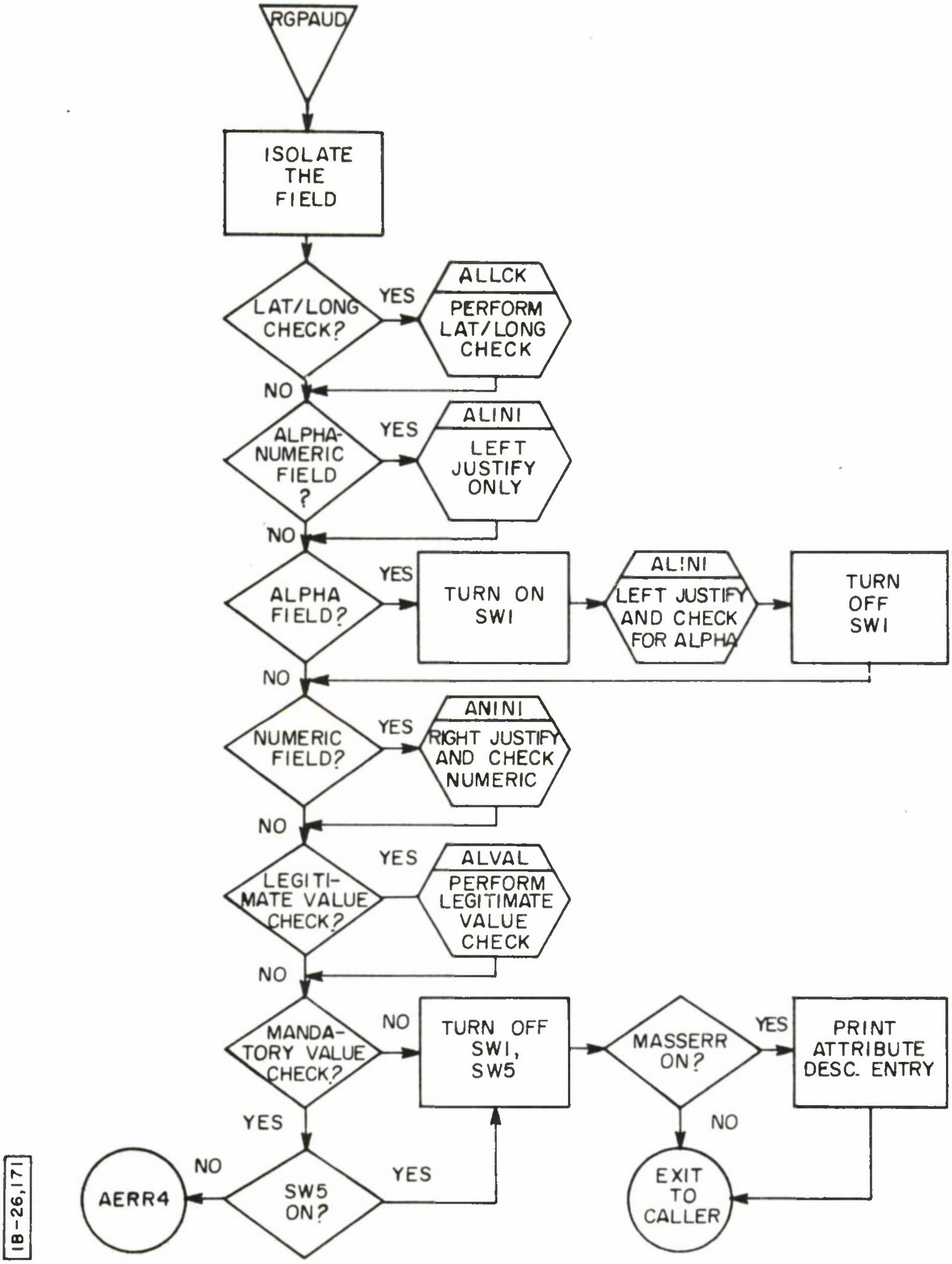

PCTL
CONTROL BREAK ?
YES
PWRTAP
NO
SUB CONTROL BREAK ?
NO
PATT3
YES
PSUBFS ON?
YES
PBEGF
NO
TURN ON PCLRSB
PWRTAI
PWRTAI
MASCTL OR MASSUB ON?
YES
PSET3
NO
INSURE THAT ANY MANDATORY INPUT RECORDS HAVE BEEN READ
PWRTAP
MASCTL OR MASSUB OR PFIRST ON ?
YES
PSET
NO
ANY MISSING ?
YES
ERROR MESSAGE "MANDATORY INPUT RECORD MISSING"
NO
RW TAPE
WRITE A RECORD ON OUTPUT TAPE
PCLRSB ON ?
YES
PSET3
NO
PSET

IB-26,262

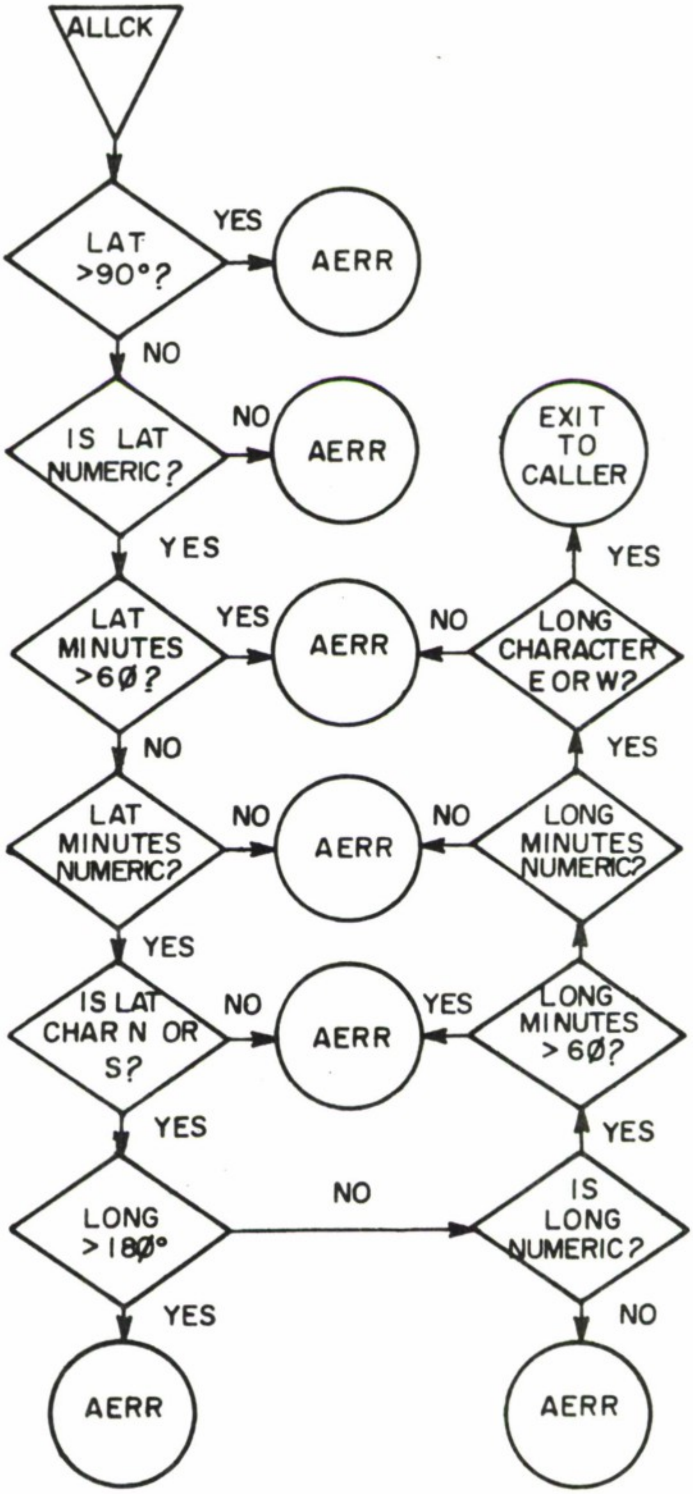

PATT3 (PATT)
ARE WE AT END OF ATTR. TBL. ?
PBEG
NO
SELECT NEXT ATTRIBUTE
RGPAUD
AUDIT ATTRIBUTE
MASSER ON ?
YES
WHICH ON ?
YES
TURN ON MASSUB & MERSWF
NO
NO
MAKE ENTRY IN DAWORK TABLE, GIVING ATTRIBUTE POSITION IN INPUT AREA, ATTR. POS. IN OUTPUT IMAGE & ATTRIBUTE LENGTH
TURN ON MASCTL AND MERSWF
TURN OFF MASERR
NO
WHICH ON ?
YES
MAKE EQUIVALENT ENTRY IN DAWOR2 TABLE
PBEG
USING DAWORK TABLE, TRANSFER DATA FROM INPUT AREA TO OUTPUT IMAGE
CLEAR INPUT AREA
TURN OFF PCLRSB, WHICH, CTLFIN, PFIRST
EXIT TO PDBLK CALLER

IB - 26,263

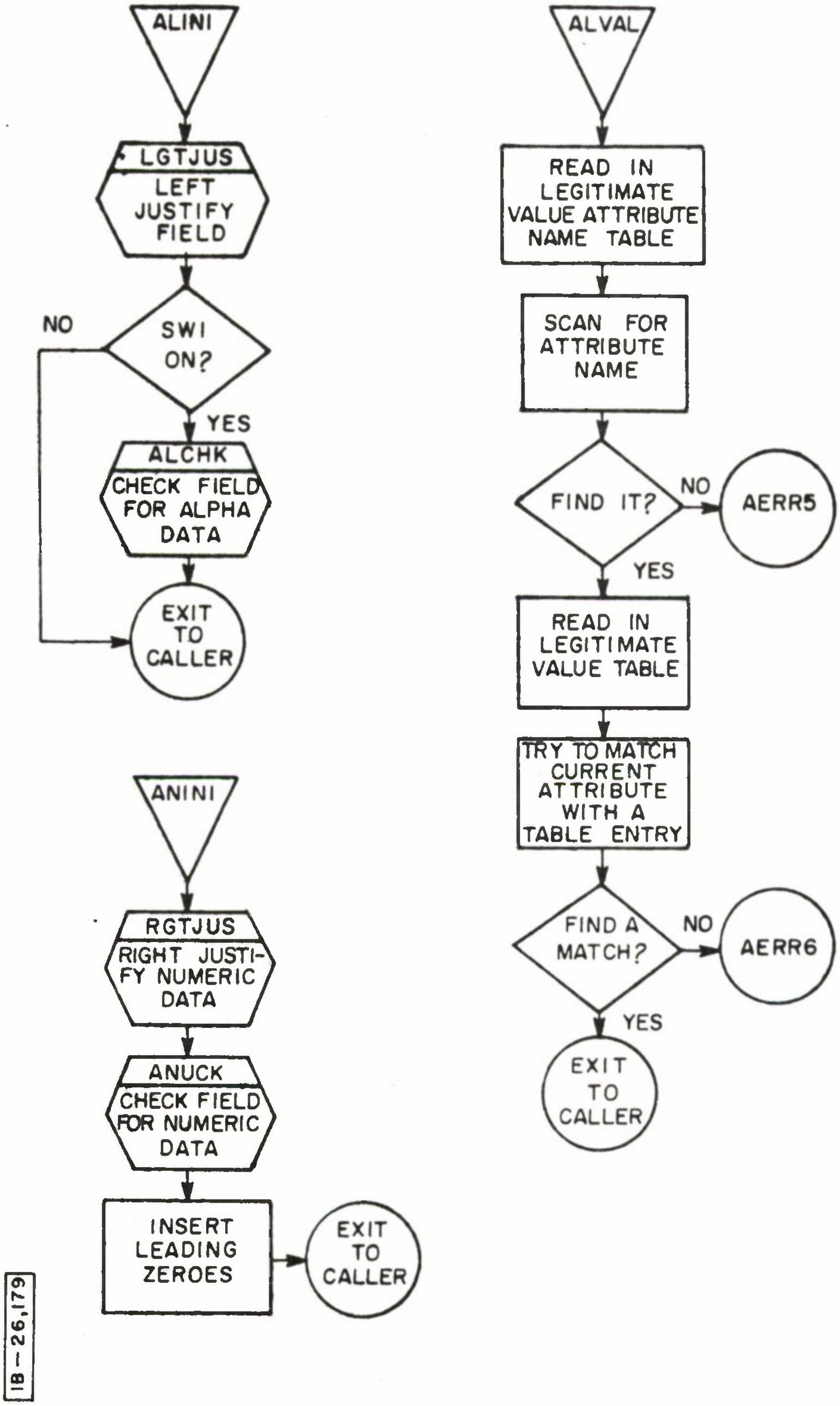

PSET
CLEAR ENTIRE OUTPUT AREA
SAVE LAST CONTROL FIELD; CLEAR SUB-CONTROL FIELD
TURN OFF MASCTL, MASSUB
CLEAR DAWOR2 TABLE
TURN ON PSUBFS
TURN OFF PFIRST
PATT3
PBEGF
PSET3
USING FIELDS DEFINED IN DAWOR2, SELECTIVELY CLEAR OUTPUT IMAGE
CLEAR DAWOR2 TABLE
TURN OFF PSUBFS
SAVE LAST SUB-CONTROL FIELD
PATT3

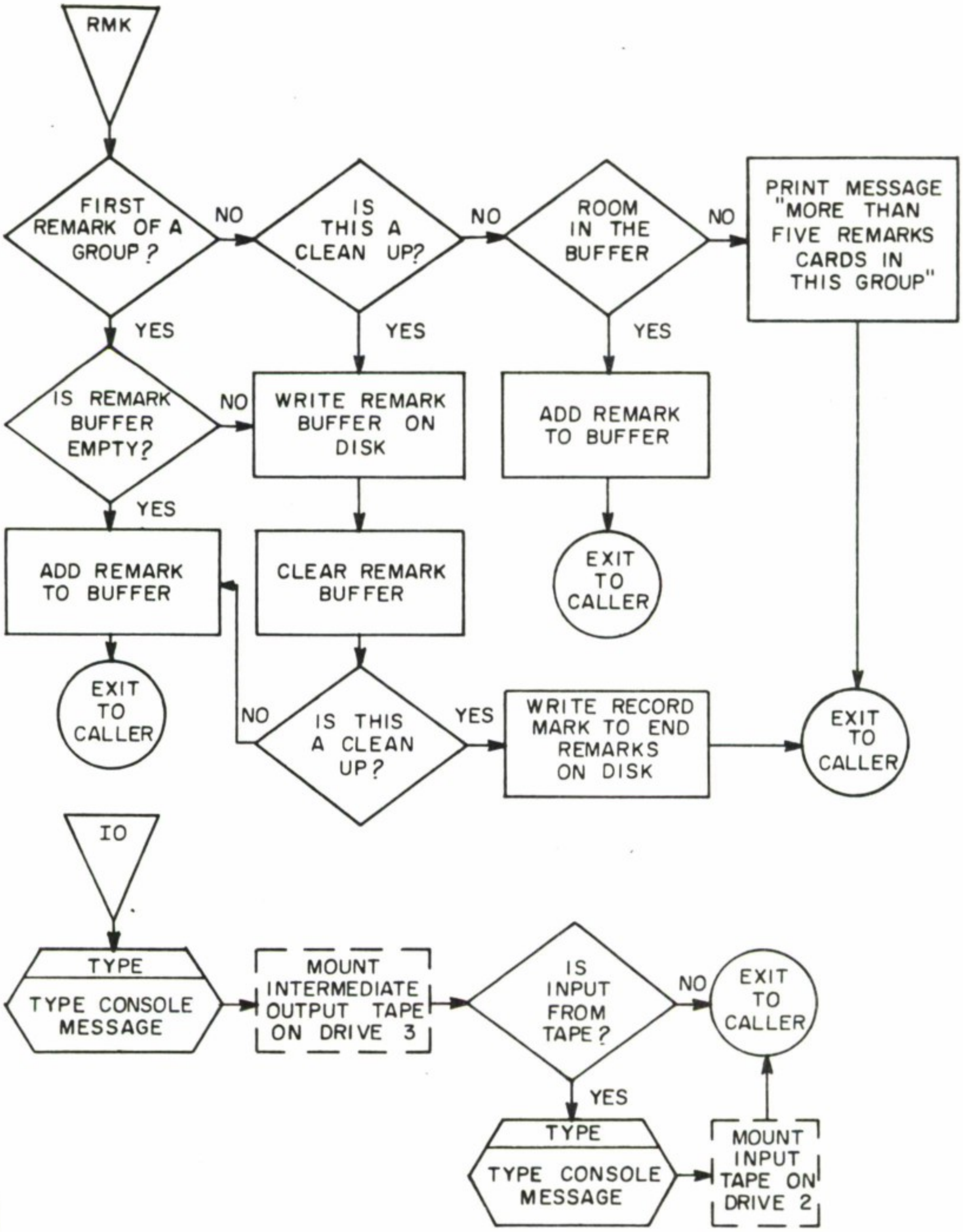


RGPAUD
ISOLATE THE FIELD
LAT/LONG CHECK?
YES
ALLCK
PERFORM LAT/LONG CHECK
NO
ALPHA-NUMERIC FIELD ?
YES
ALINI
LEFT JUSTIFY ONLY
NO
ALPHA FIELD?
YES
TURN ON SW1
AL!NI
LEFT JUSTIFY AND CHECK FOR ALPHA
TURN OFF SW1
NO
NUMERIC FIELD?
YES
ANINI
RIGHT JUSTIFY AND CHECK NUMERIC
NO
LEGITI-MATE VALUE CHECK?
YES
ALVAL
PERFORM LEGITIMATE VALUE CHECK
NO
MANDA-TORY VALUE CHECK?
NO
TURN OFF SW1, SW5
MASSERR ON?
YES
PRINT ATTRIBUTE DESC. ENTRY
YES
NO
AERR4
NO
SW5 ON?
YES
EXIT TO CALLER
IB-26,171

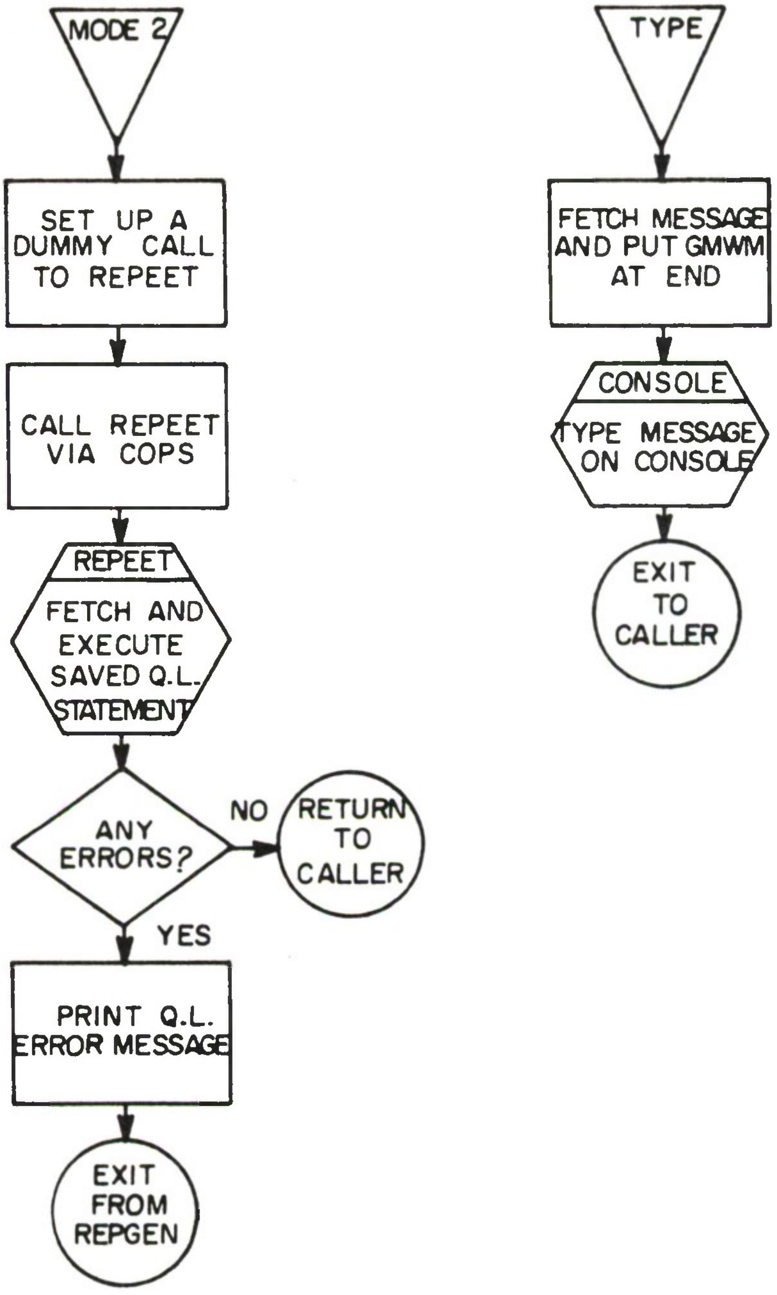

ALLCK
LAT >90°?
YES
AERR
NO
IS LAT NUMERIC?
NO
AERR
YES
LAT MINUTES >6Ø?
YES
AERR
NO
LAT MINUTES NUMERIC?
NO
AERR
YES
IS LAT CHAR N OR S?
NO
AERR
YES
LONG >18Ø°
YES
AERR
NO
IS LONG NUMERIC?
NO
AERR
YES
LONG MINUTES >6Ø?
YES
AERR
LONG MINUTES NUMERIC?
NO
AERR
YES
LONG CHARACTER E OR W?
NO
AERR
YES
EXIT TO CALLER

IB—26,178


ALINI
LGTJUS
LEFT JUSTIFY FIELD
SWI ON?
NO
YES
ALCHK
CHECK FIELD FOR ALPHA DATA
EXIT TO CALLER
ANINI
RGTJUS
RIGHT JUSTIFY NUMERIC DATA
ANUCK
CHECK FIELD FOR NUMERIC DATA
INSERT LEADING ZEROES
EXIT TO CALLER
ALVAL
READ IN LEGITIMATE VALUE ATTRIBUTE NAME TABLE
SCAN FOR ATTRIBUTE NAME
FIND IT?
NO
AERR5
YES
READ IN LEGITIMATE VALUE TABLE
TRY TO MATCH CURRENT ATTRIBUTE WITH A TABLE ENTRY
FIND A MATCH?
NO
AERR6
YES
EXIT TO CALLER
IB-26,179


RMK
FIRST REMARK OF A GROUP?
NO
YES
IS THIS A CLEAN UP?
NO
YES
ROOM IN THE BUFFER
NO
YES
PRINT MESSAGE "MORE THAN FIVE REMARKS CARDS IN THIS GROUP"
IS REMARK BUFFER EMPTY?
NO
YES
WRITE REMARK BUFFER ON DISK
ADD REMARK TO BUFFER
EXIT TO CALLER
ADD REMARK TO BUFFER
CLEAR REMARK BUFFER
EXIT TO CALLER
NO
IS THIS A CLEAN UP?
YES
WRITE RECORD MARK TO END REMARKS ON DISK
EXIT TO CALLER
IO
TYPE
TYPE CONSOLE MESSAGE
MOUNT INTERMEDIATE OUTPUT TAPE ON DRIVE 3
IS INPUT FROM TAPE?
NO
EXIT TO CALLER
YES
TYPE
TYPE CONSOLE MESSAGE
MOUNT INPUT TAPE ON DRIVE 2

IB-26,222


MODE 2
SET UP A DUMMY CALL TO REPEET
CALL REPEET VIA COPS
REPEET
FETCH AND EXECUTE SAVED Q.L. STATEMENT
ANY ERRORS?
NO
RETURN TO CALLER
YES
PRINT Q.L. ERROR MESSAGE
EXIT FROM REPGEN
TYPE
FETCH MESSAGE AND PUT GMWM AT END
CONSOLE
TYPE MESSAGE ON CONSOLE
EXIT TO CALLER

18-26,221

RGPBYP

Purpose

This routine issues mount messages for the output phase, completes the I/O Communication Record and calls in the output phase.

Calling Sequence

If called internally from REPGEN, the dummy typewriter call which is constructed is

RGPBYP***

If called externally, enter on the console typewriter

RGPBYPrrrfcn

where rrr = 3-digit report number (no blanks)

f = Format number of the Output Program to be executed. Must be 1-7 or blank.

c = Classification code: U, C, S or T for Unclassified, Confidential, Secret or Top Secret.

n = Number of copies of printed output. Must be 1-9 or blank.

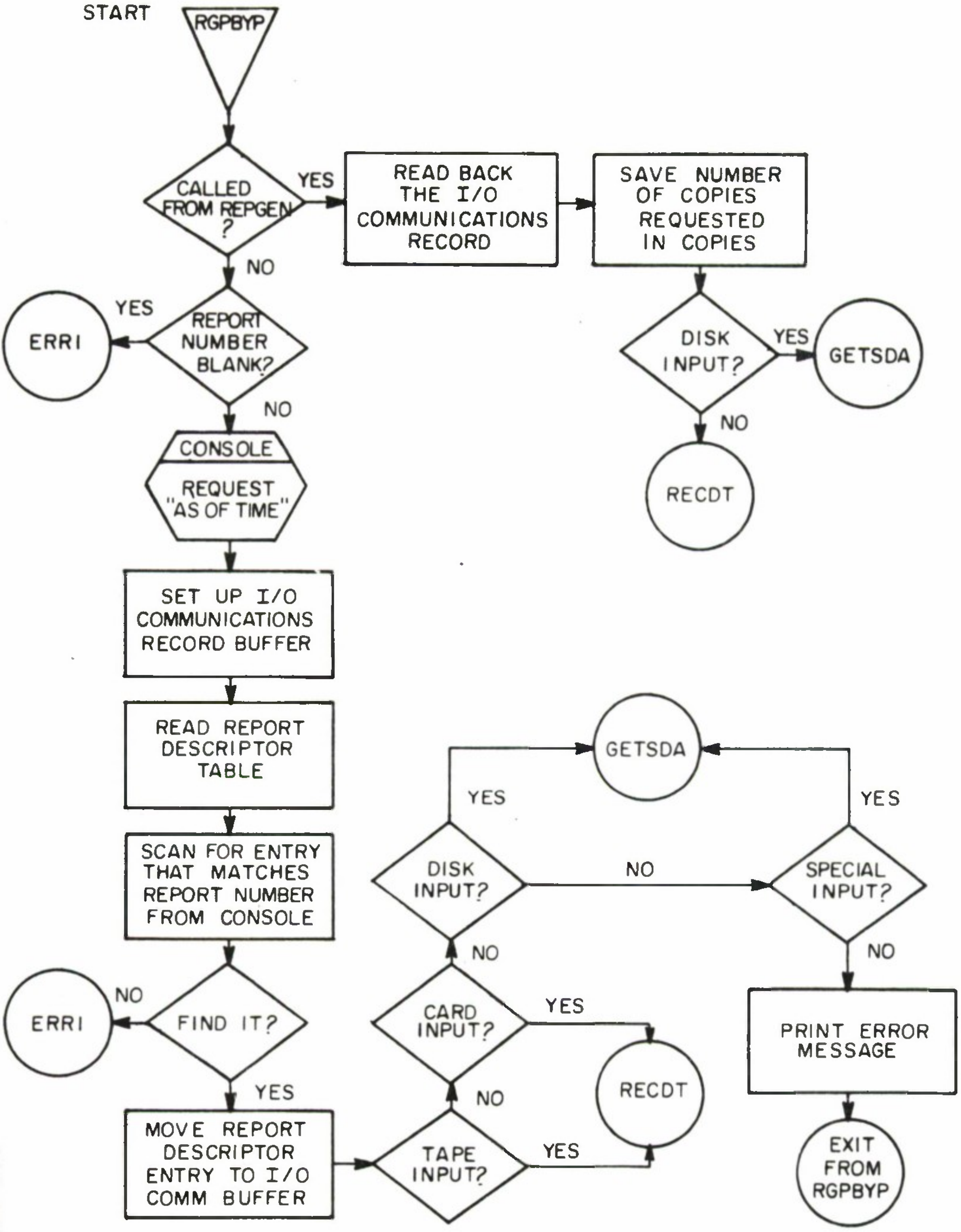
Final return is to calling program.

Description

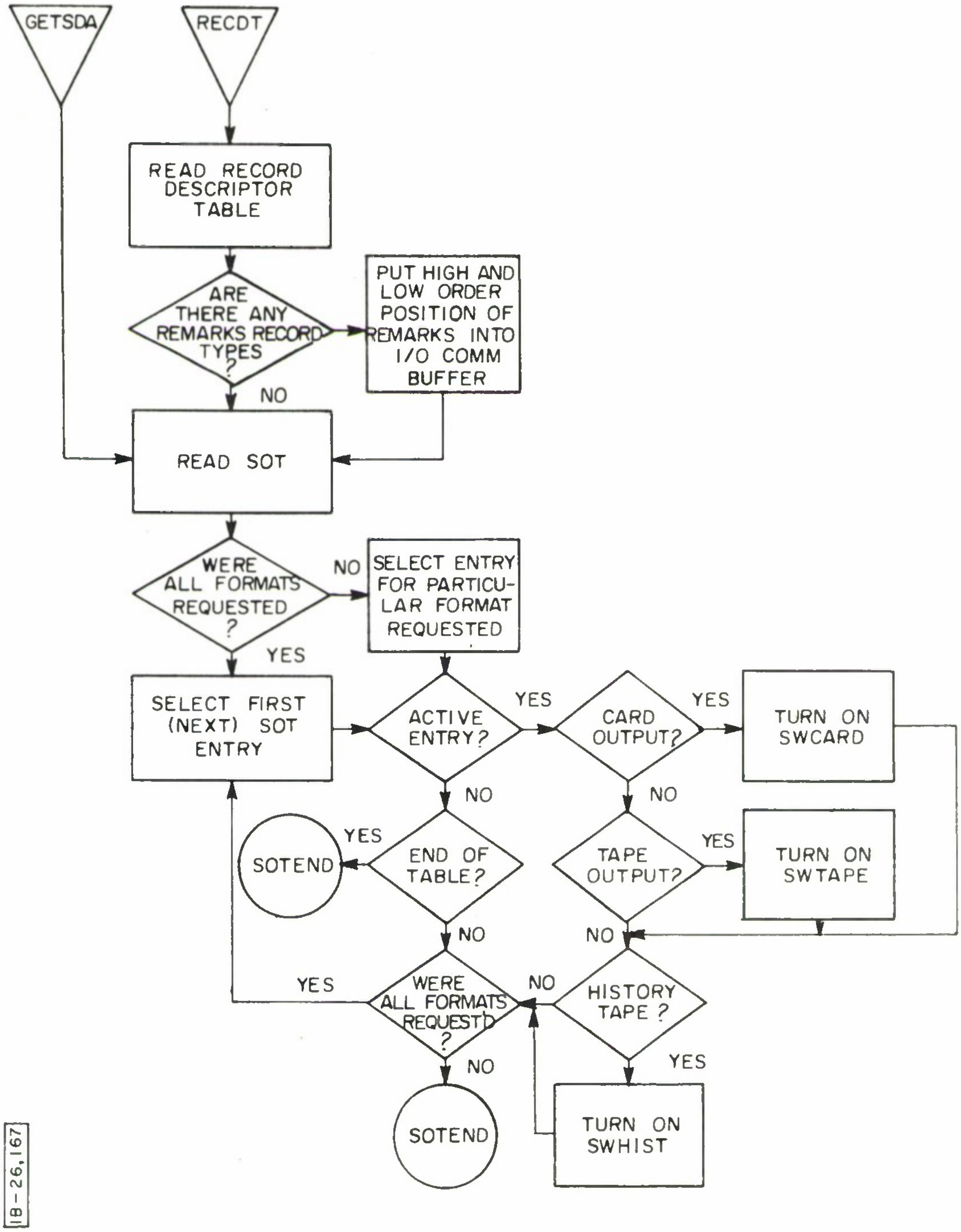
If RGPBYP was called from REPGEN, the I/O Communication Record is read back from disk. If RGPBYP was externally called, the AS OF TIME is requested, and an I/O Communication Record is constructed.

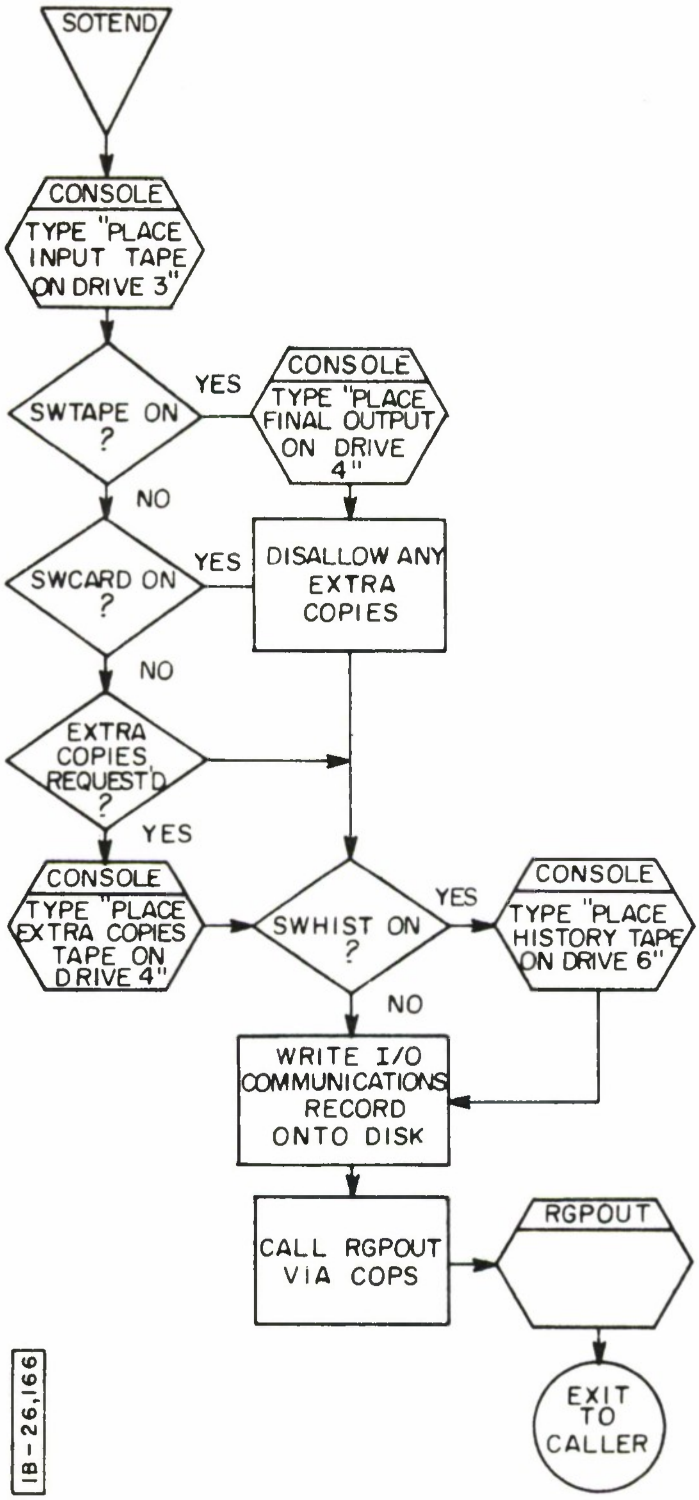
The SOT Table for the requested report is accessed and used to determine the output devices required and the mount messages to be issued. Requests for multiple copies are ignored if the output device is not the printer.

After the mount messages are issued, the I/O Communication Record is completed and written on the system disk. The output program RGPOUT is then called via COPS. After return is made from RGPOUT, RGPBYP will exit to its calling program.


START
RGPBYP
CALLED FROM REPGEN ?
YES
READ BACK THE I/O COMMUNICATIONS RECORD
SAVE NUMBER OF COPIES REQUESTED IN COPIES
NO
YES
ERRI
REPORT NUMBER BLANK?
DISK INPUT?
YES
GETSDA
NO
NO
RECDT
CONSOLE
REQUEST "AS OF TIME"
SET UP I/O COMMUNICATIONS RECORD BUFFER
READ REPORT DESCRIPTOR TABLE
GETSDA
YES
YES
SCAN FOR ENTRY THAT MATCHES REPORT NUMBER FROM CONSOLE
DISK INPUT?
NO
SPECIAL INPUT?
NO
NO
NO
ERRI
FIND IT?
CARD INPUT?
YES
PRINT ERROR MESSAGE
YES
NO
RECDT
MOVE REPORT DESCRIPTOR ENTRY TO I/O COMM BUFFER
TAPE INPUT?
YES
EXIT FROM RGPBYP

1B-26,210


GETSDA
RECDT
READ RECORD DESCRIPTOR TABLE
ARE THERE ANY REMARKS RECORD TYPES ?
PUT HIGH AND LOW ORDER POSITION OF REMARKS INTO I/O COMM BUFFER
NO
READ SOT
WERE ALL FORMATS REQUESTED ?
NO
SELECT ENTRY FOR PARTICULAR FORMAT REQUESTED
YES
SELECT FIRST (NEXT) SOT ENTRY
ACTIVE ENTRY?
YES
CARD OUTPUT?
YES
TURN ON SWCARD
NO
NO
SOTEND
YES
END OF TABLE?
TAPE OUTPUT?
YES
TURN ON SWTAPE
NO
NO
YES
WERE ALL FORMATS REQUESTD ?
NO
HISTORY TAPE ?
NO
YES
SOTEND
TURN ON SWHIST
IB-26,167


SOTEND
CONSOLE
TYPE "PLACE INPUT TAPE ON DRIVE 3"
SWTAPE ON ?
YES
NO
CONSOLE
TYPE "PLACE FINAL OUTPUT ON DRIVE 4"
SWCARD ON ?
YES
NO
DISALLOW ANY EXTRA COPIES
EXTRA COPIES REQUEST'D ?
YES
CONSOLE
TYPE "PLACE EXTRA COPIES TAPE ON DRIVE 4"
SWHIST ON ?
YES
NO
CONSOLE
TYPE "PLACE HISTORY TAPE ON DRIVE 6"
WRITE I/O COMMUNICATIONS RECORD ONTO DISK
CALL RGPOUT VIA COPS
RGPOUT
EXIT TO CALLER
IB-26,166

RGPOUT

Purpose

RGPOUT acts as the output phase supervisor, controlling the loading and execution of the various output phase programs.

Calling Sequence

RGPOUT is called internally from RGPBYP as follows:

```
B   COPS
DCW 'RGPOUT'
```

Description

The Read/Write Tape routine RWTAPE, the CONSOLE routine and the initial Communication Zone are all assembled with RGPOUT, and are loaded into core when RGPOUT is loaded. They remain in core throughout the output phase.

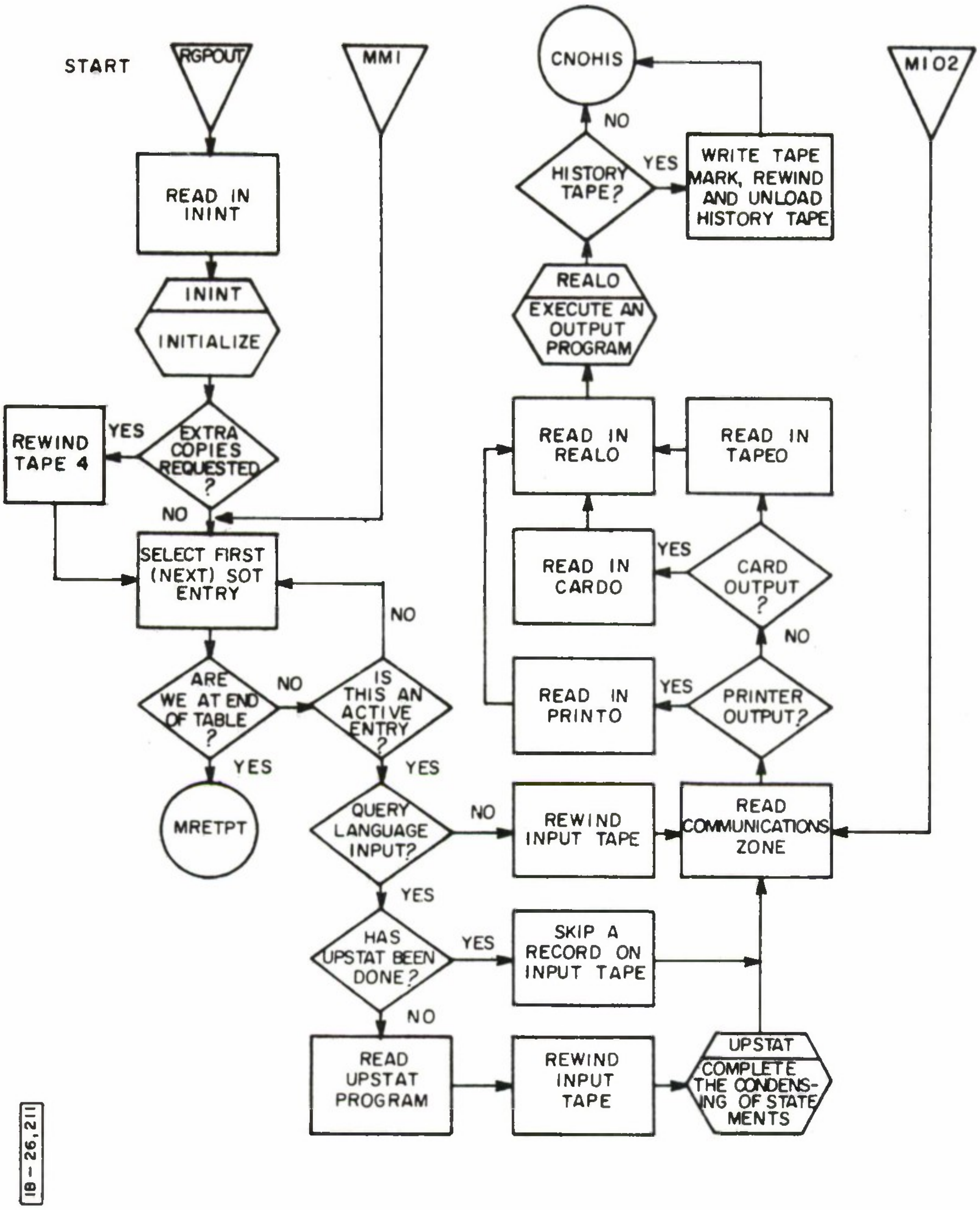
RGPOUT calls the output phase initialization routine ININT into core and executes it. Next the SOT Table entry for the output program to be executed is examined. If input was via Query Language, the SOT entry is checked to see if UPSTAT has been executed for this output program.

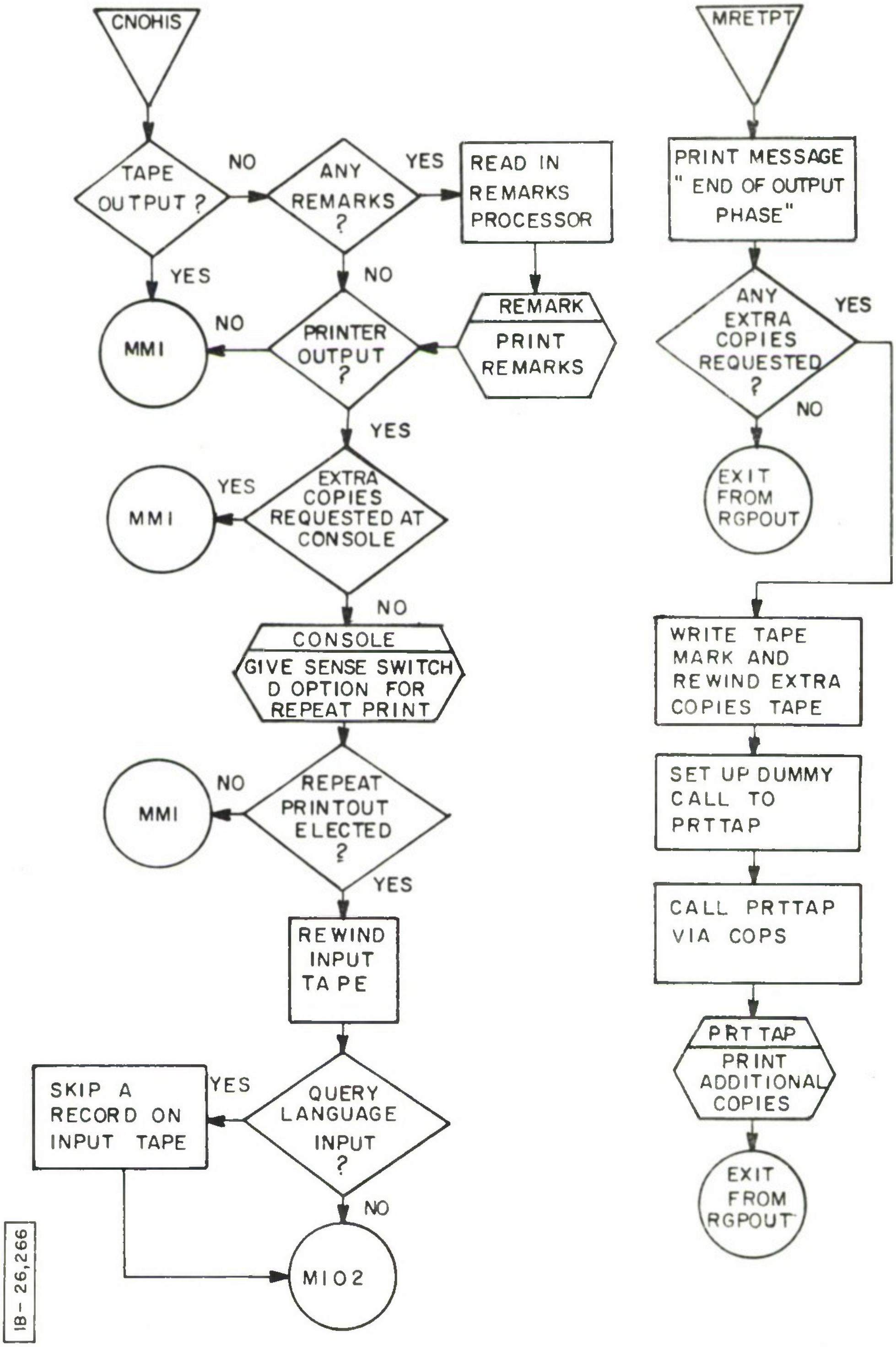
If input was via Query Language and UPSTAT has not been executed, UPSTAT is loaded and executed to complete the condensing of the output program.

The appropriate output routine, PRINTO, CARDO or TAPEO, is loaded depending upon whether the output device is the printer, the card punch or magnetic tape. The main output program REALO is loaded and executed to produce the desired output.

After REALO executes the output program, RGPOUT checks to see if any more formats were requested for this report. If so, the appropriate output program is loaded, REALO is loaded, and the next output program is executed

When all requested output programs have been executed, RGPOUT checks to see if extra copies of printed reports were requested. If so, the extra copies tape is rewound and an internal call to PRTTAP is set up and executed to yield the necessary printed copies.


START
RGPOUT
MMI
CNOHIS
MI02
READ IN ININT
ININT
INITIALIZE
EXTRA COPIES REQUESTED ?
YES
REWIND TAPE 4
NO
SELECT FIRST (NEXT) SOT ENTRY
ARE WE AT END OF TABLE ?
NO
YES
MRETPT
IS THIS AN ACTIVE ENTRY ?
NO
YES
QUERY LANGUAGE INPUT?
NO
REWIND INPUT TAPE
YES
HAS UPSTAT BEEN DONE?
YES
SKIP A RECORD ON INPUT TAPE
NO
READ UPSTAT PROGRAM
REWIND INPUT TAPE
UPSTAT
COMPLETE THE CONDENS-ING OF STATE MENTS
READ COMMUNICATIONS ZONE
PRINTER OUTPUT?
YES
READ IN PRINTO
NO
CARD OUTPUT ?
YES
READ IN CARDO
READ IN TAPEO
READ IN REALO
REALO
EXECUTE AN OUTPUT PROGRAM
HISTORY TAPE?
NO
YES
WRITE TAPE MARK, REWIND AND UNLOAD HISTORY TAPE
18 - 26,211


CNOHIS
TAPE OUTPUT ?
NO
YES
ANY REMARKS ?
YES
NO
READ IN REMARKS PROCESSOR
REMARK
PRINT REMARKS
MMI
NO
PRINTER OUTPUT ?
YES
EXTRA COPIES REQUESTED AT CONSOLE
YES
MMI
NO
CONSOLE
GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT
REPEAT PRINTOUT ELECTED ?
NO
MMI
YES
REWIND INPUT TAPE
QUERY LANGUAGE INPUT ?
YES
SKIP A RECORD ON INPUT TAPE
NO
MIO2
MRETPT
PRINT MESSAGE " END OF OUTPUT PHASE"
ANY EXTRA COPIES REQUESTED ?
YES
NO
EXIT FROM RGPOUT
WRITE TAPE MARK AND REWIND EXTRA COPIES TAPE
SET UP DUMMY CALL TO PRTTAP
CALL PRTTAP VIA COPS
PRTTAP
PRINT ADDITIONAL COPIES
EXIT FROM RGPOUT
IB- 26,266

ININT

Purpose

Performs initialization for the output phase of REPGEN.

Calling Sequence

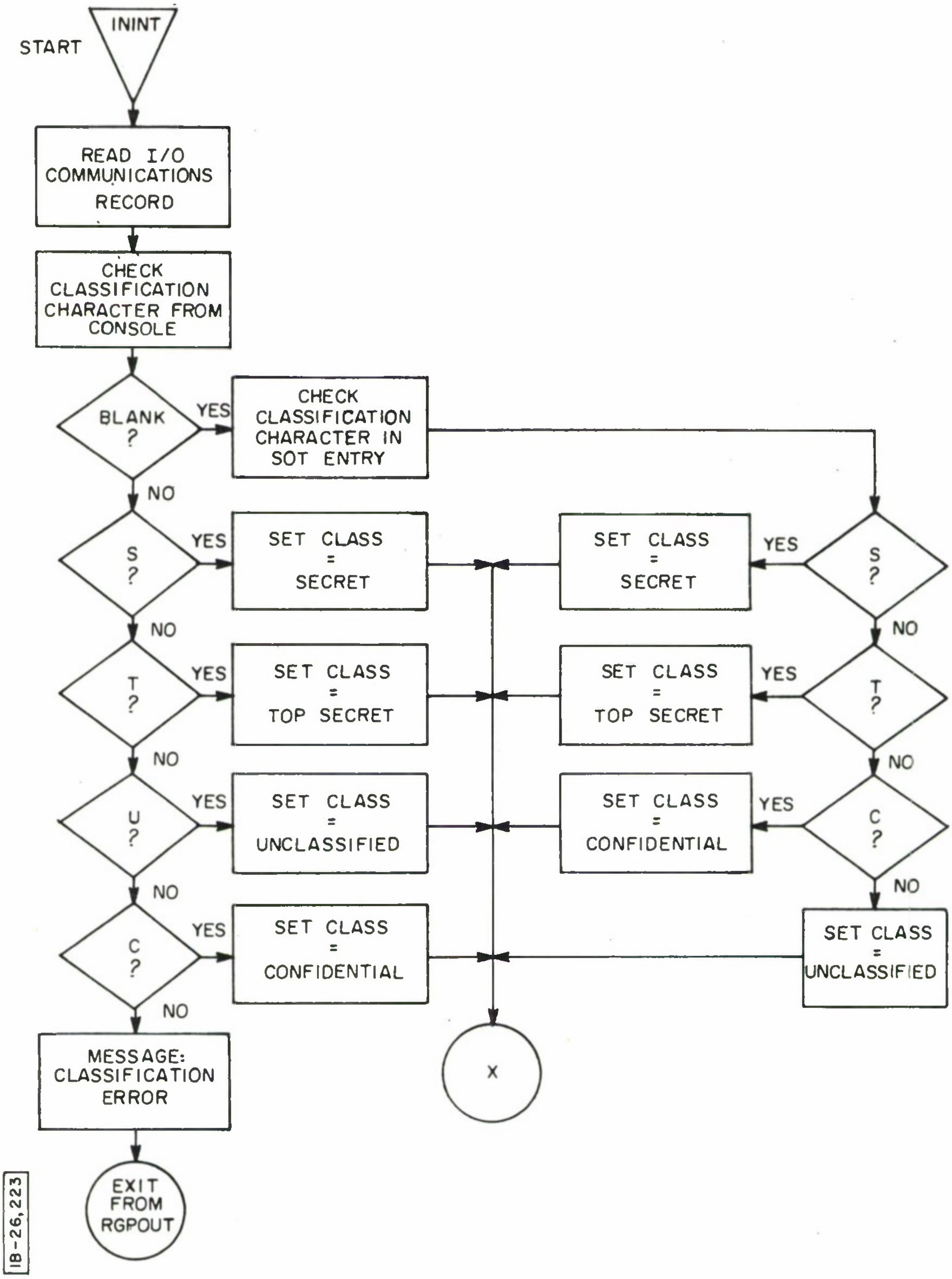
ININT is called from RGPOUT by

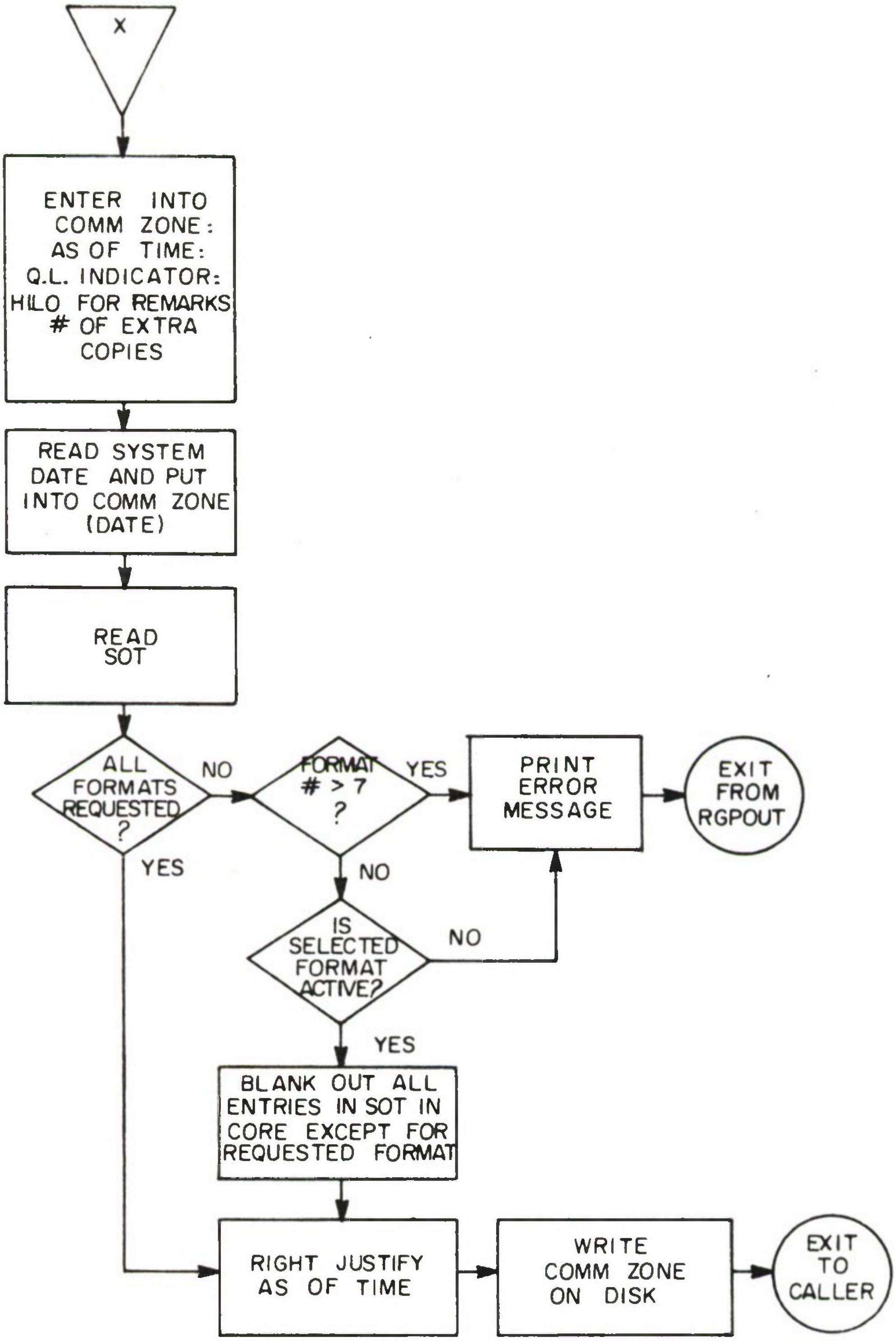
```
B    ININT

(Normal Return)
```

Description

ININT reads the I/O Communication Record from disk. The classification code is checked for U, C, S or T and causes the system attribute CLASS to be set to Unclassified, Confidential, Secret or Top Secret. Other data is taken from the I/O Communication Record and entered into the Communication Zone. The SOT Table is read into the Communication Zone. If only one specific format was requested, all other entries in the SOT Table are set to blanks. The Communication Zone is written on the disk from where it may be read back to re-initialize it between execution of output programs.

Return is made to RGPOUT.


ININT
START
READ I/O COMMUNICATIONS RECORD
CHECK CLASSIFICATION CHARACTER FROM CONSOLE
BLANK ?
YES
CHECK CLASSIFICATION CHARACTER IN SOT ENTRY
NO
S ?
YES
SET CLASS = SECRET
NO
T ?
YES
SET CLASS = TOP SECRET
NO
U ?
YES
SET CLASS = UNCLASSIFIED
NO
C ?
YES
SET CLASS = CONFIDENTIAL
NO
MESSAGE: CLASSIFICATION ERROR
EXIT FROM RGPOUT
X
S ?
YES
SET CLASS = SECRET
NO
T ?
YES
SET CLASS = TOP SECRET
NO
C ?
YES
SET CLASS = CONFIDENTIAL
NO
SET CLASS = UNCLASSIFIED
IB-26,223


X
ENTER INTO COMM ZONE: AS OF TIME: Q.L. INDICATOR: HILO FOR REMARKS # OF EXTRA COPIES
READ SYSTEM DATE AND PUT INTO COMM ZONE (DATE)
READ SOT
ALL FORMATS REQUESTED ?
NO
FORMAT # > 7 ?
YES
PRINT ERROR MESSAGE
EXIT FROM RGPOUT
YES
NO
IS SELECTED FORMAT ACTIVE?
NO
YES
BLANK OUT ALL ENTRIES IN SOT IN CORE EXCEPT FOR REQUESTED FORMAT
RIGHT JUSTIFY AS OF TIME
WRITE COMM ZONE ON DISK
EXIT TO CALLER

IB-26,224

UPSTAT

Purpose

UPSTAT is called by RGPOUT to complete the condensing of output statements which LOD could not completely condense. This only occurs in the case where the Serial File Tape is the product of a Query Language retrieval from a parallel file stored on system disk.

Calling Sequence

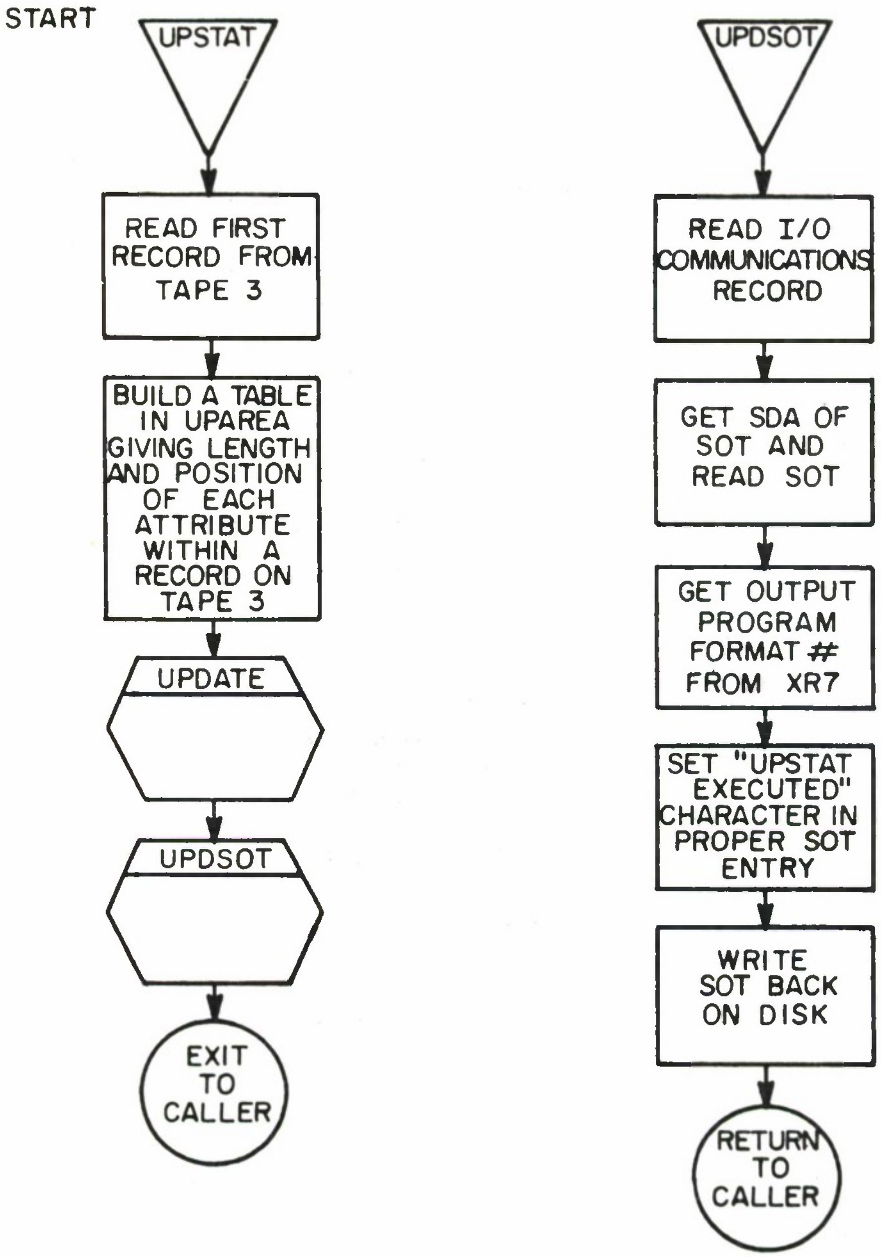
UPSTAT is called from RGPOUT by

```
B      UPSTAT
(Normal Return)
```

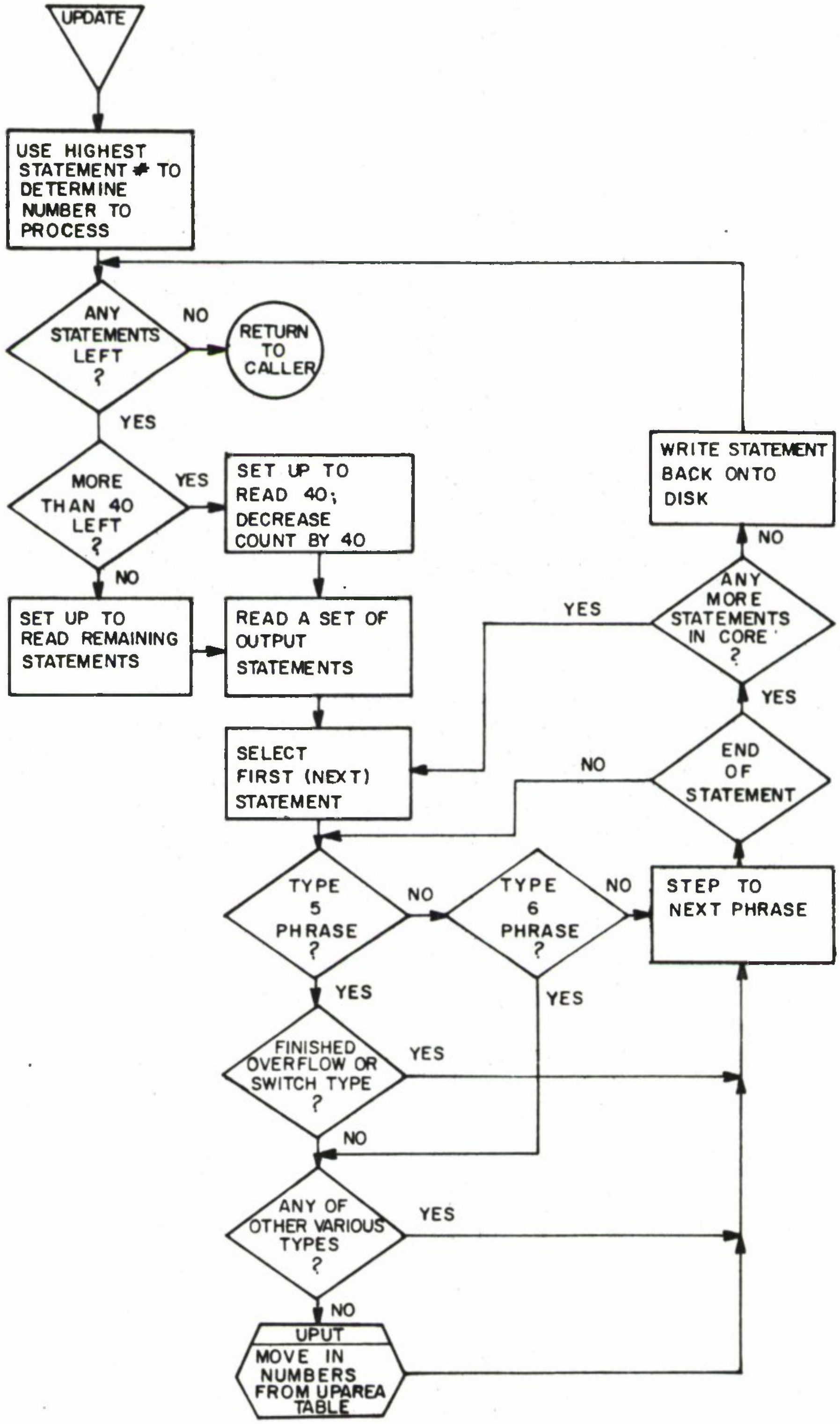
Description

When LOD condenses the statements of an output program which uses a Query Language generated Serial File Tape as input, the statements cannot be completely condensed. In particular, LOD does not know the length or position in the Serial File Tape records of each attribute. LOD does know the order in which the attributes will appear on the Serial File Tape. In the case in point, LOD assigns each attribute a number corresponding to its relative position in the Serial File Tape records, and substitutes this number for each occurrence of an attribute name in an output statement.

When UPSTAT is executed, it reads the first record of the Serial File Tape and uses it to construct a table. The table so constructed contains the length and relative position on the Serial File Tape of each attribute. The condensed output program is read from disk, and the attribute numbers are now replaced with length and position data from the table. The fully condensed statements are written back onto disk, the "UPSTAT executed" indicator is set in the appropriate SOT entry, and return is made to RGPOUT.


START
UPSTAT
READ FIRST RECORD FROM TAPE 3
BUILD A TABLE IN UPAREA GIVING LENGTH AND POSITION OF EACH ATTRIBUTE WITHIN A RECORD ON TAPE 3
UPDATE
UPDSOT
EXIT TO CALLER
UPDSOT
READ I/O COMMUNICATIONS RECORD
GET SDA OF SOT AND READ SOT
GET OUTPUT PROGRAM FORMAT # FROM XR7
SET "UPSTAT EXECUTED" CHARACTER IN PROPER SOT ENTRY
WRITE SOT BACK ON DISK
RETURN TO CALLER

IB-26,205


UPDATE
USE HIGHEST STATEMENT # TO DETERMINE NUMBER TO PROCESS
ANY STATEMENTS LEFT ?
NO
RETURN TO CALLER
YES
MORE THAN 40 LEFT ?
YES
SET UP TO READ 40; DECREASE COUNT BY 40
NO
SET UP TO READ REMAINING STATEMENTS
READ A SET OF OUTPUT STATEMENTS
SELECT FIRST (NEXT) STATEMENT
WRITE STATEMENT BACK ONTO DISK
NO
ANY MORE STATEMENTS IN CORE ?
YES
YES
END OF STATEMENT
NO
TYPE 5 PHRASE ?
NO
TYPE 6 PHRASE ?
NO
STEP TO NEXT PHRASE
YES
YES
FINISHED OVERFLOW OR SWITCH TYPE ?
YES
NO
ANY OF OTHER VARIOUS TYPES ?
YES
NO
UPUT
MOVE IN NUMBERS FROM UPAREA TABLE

IB – 26,267

PRINTO

Purpose

Print a line on the printer or read a record from the Serial File Tape.

Calling Sequence

PRINTO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

```
B       INPUT
(Normal return)
```

If the output image is to be printed, the call is

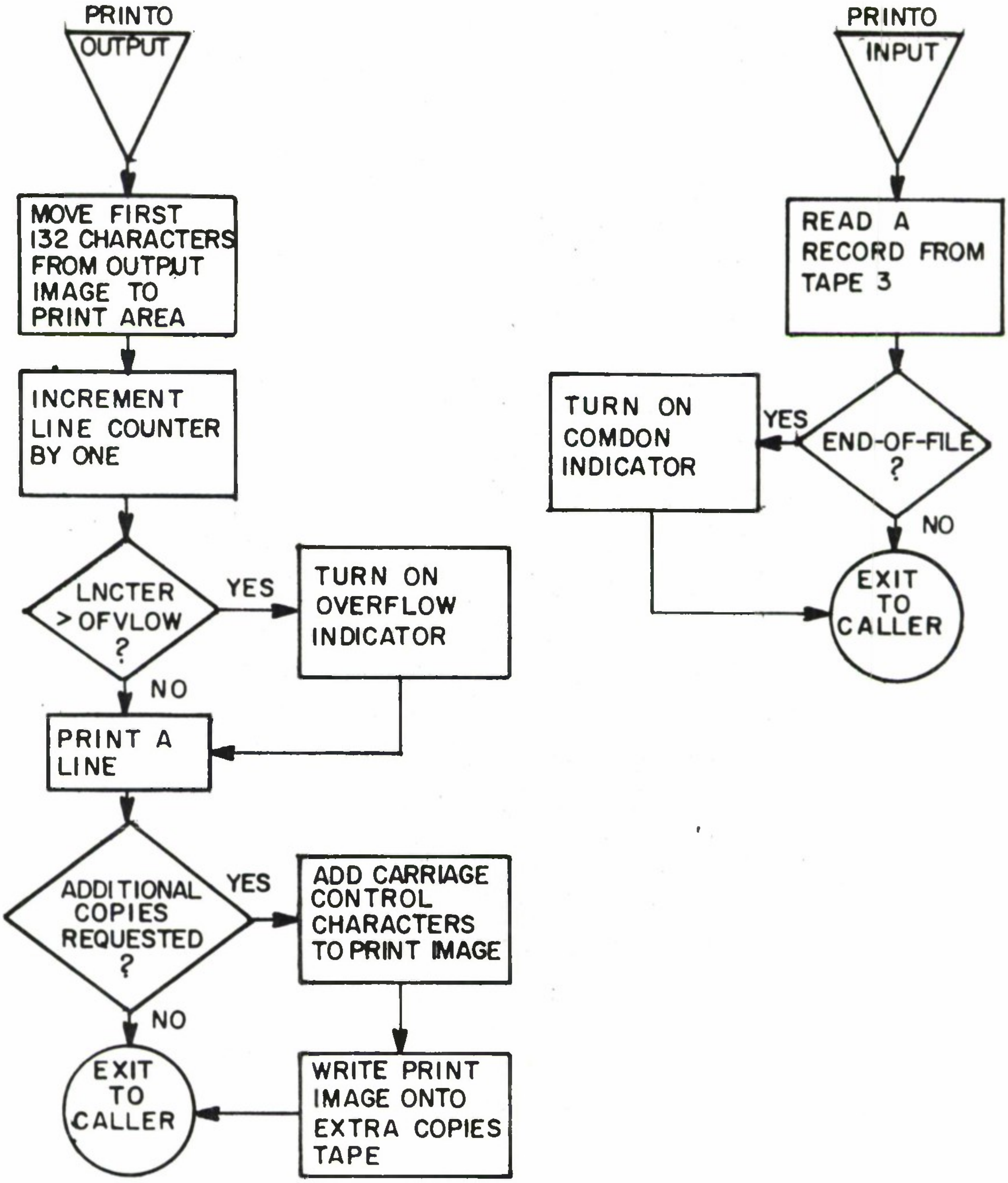
```
B       OUTPUT
(Normal return)
```

Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communications Zone.

If entry is made at OUTPUT, the first 132 characters of the output image are printed. The line counter is incremented, and if it exceeds the value in OVFLOW, the overflow indicator is turned on.

If additional copies of the printed report were requested, the print image is also written onto the extra copies tape. Three characters are added to the record on tape to indicate any blank lines or page ejects which should be accomplished before this line is printed. (This spacing information will be interpreted by the PRTTAP program which generates additional copies of the printed report from the extra copies tape.)


PRINTO
OUTPUT
MOVE FIRST 132 CHARACTERS FROM OUTPUT IMAGE TO PRINT AREA
INCREMENT LINE COUNTER BY ONE
LNCTER > OFVLOW ?
YES
TURN ON OVERFLOW INDICATOR
NO
PRINT A LINE
ADDITIONAL COPIES REQUESTED ?
YES
ADD CARRIAGE CONTROL CHARACTERS TO PRINT IMAGE
NO
EXIT TO CALLER
WRITE PRINT IMAGE ONTO EXTRA COPIES TAPE
PRINTO
INPUT
READ A RECORD FROM TAPE 3
END-OF-FILE ?
YES
TURN ON COMDON INDICATOR
NO
EXIT TO CALLER

IB-26,268

CARDO

Purpose

Punch a card from the output image or read a record from the Serial File Tape.

Calling Sequence

CARDO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

```
B     INPUT

(normal return)
```

If a card is to be punched, the call is

```
B     OUTPUT

(normal return)
```

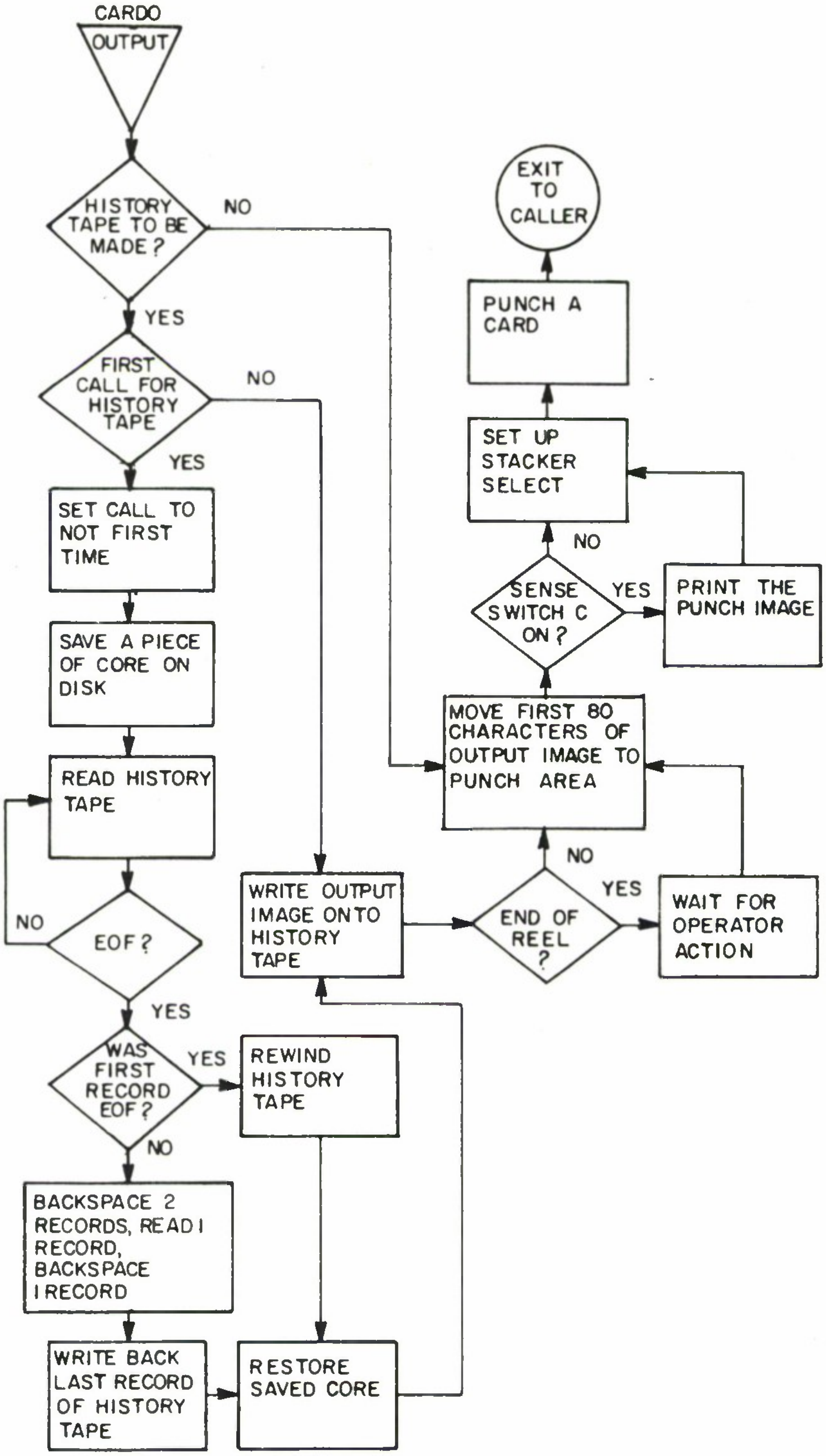
Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communications Zone.

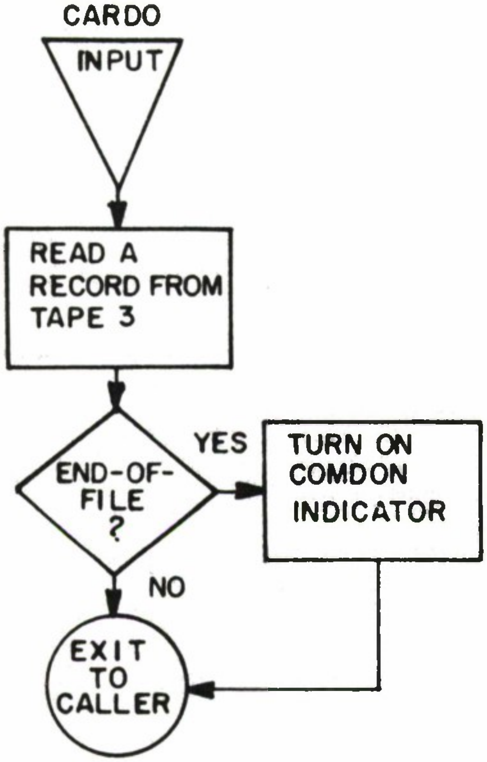
If entry is made at OUTPUT, a check is made to see if a history tape was requested. If a history tape was requested, the first entry at OUTPUT causes the history tape to be spaced until an end-of-file is detected, and then positioned so that records may be added to the file. The output image is then written on the history tape.

If Sense Switch C is on, the output image is printed.

Next the stacker select character is acquired from the Communication Zone, and the first 8∅ characters of the output image are punched in a card.


CARDO
OUTPUT
HISTORY TAPE TO BE MADE ?
NO
YES
FIRST CALL FOR HISTORY TAPE
NO
YES
SET CALL TO NOT FIRST TIME
SAVE A PIECE OF CORE ON DISK
READ HISTORY TAPE
NO
EOF ?
YES
WAS FIRST RECORD EOF ?
YES
NO
REWIND HISTORY TAPE
BACKSPACE 2 RECORDS, READ I RECORD, BACKSPACE I RECORD
WRITE BACK LAST RECORD OF HISTORY TAPE
RESTORE SAVED CORE
WRITE OUTPUT IMAGE ONTO HISTORY TAPE
END OF REEL ?
NO
YES
WAIT FOR OPERATOR ACTION
MOVE FIRST 80 CHARACTERS OF OUTPUT IMAGE TO PUNCH AREA
SENSE SWITCH C ON ?
YES
NO
PRINT THE PUNCH IMAGE
SET UP STACKER SELECT
PUNCH A CARD
EXIT TO CALLER

18-26,269


CARDO
INPUT
READ A RECORD FROM TAPE 3
END-OF-FILE ?
YES
TURN ON COMDON INDICATOR
NO
EXIT TO CALLER

TAPEO

Purpose

Write an output image onto magnetic tape, write a Header record on magnetic tape, write a Trailer record on magnetic tape or read a record from the Serial File Tape.

Calling Sequence

TAPEO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

```
B    INPUT
(normal return)
```

If the output image is to be written on tape, the call is

```
B    OUTPUT
(normal return)
```

If a Header is to be written on tape, the call is

```
B    HEADER
DCW  'NNN'
```

where NNN is the number of characters of the output image to be used as a Header record. If a Trailer record is to be written on tape, the call is

```
B    TRAILER
DCW  'NNN'
```

where NNN is the number of characters of the output image to be used as a Trailer record.

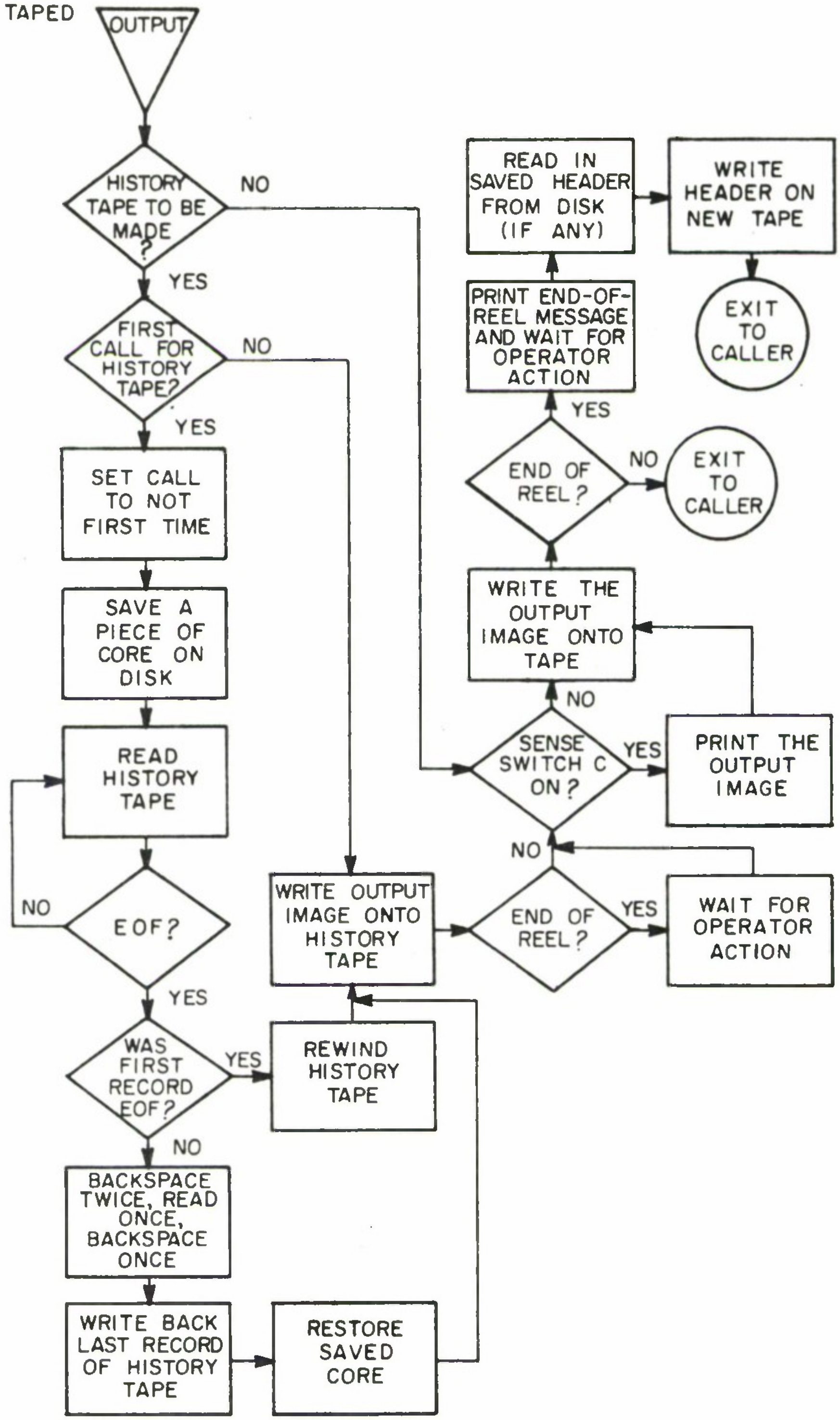
Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communication Zone.

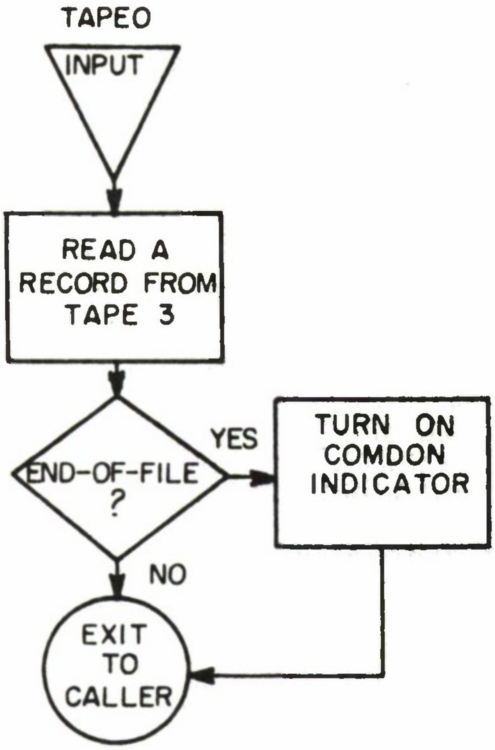
If entry is made at OUTPUT, a check is made to see if a history tape was requested. If a history tape was requested, the first entry at OUTPUT causes the history tape to be spaced until an end-of-file is detected, and then positioned so that records may be added to the file. The output image (up to 1ØØØ characters) is then written on the history tape. If Sense Switch C is on, the first 132 characters of the output image are printed. The output image is then written on the output tape.

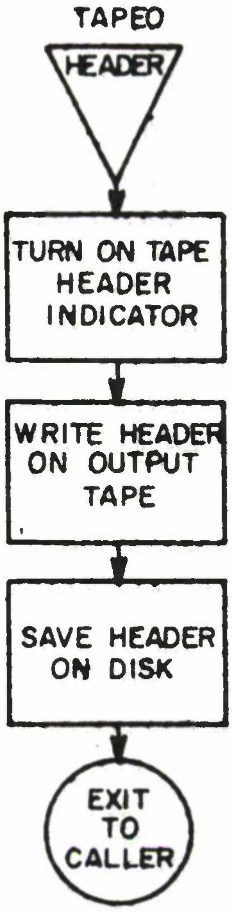
If entry is made at HEADER, the Header indicator is turned on, the first NNN characters of the output image are written on the output tape, and are also saved on the system disk (if a subsequent entry at OUTPUT results in an end-of-reel condition on the output tape, the saved Header will be written as the first record of the next reel).

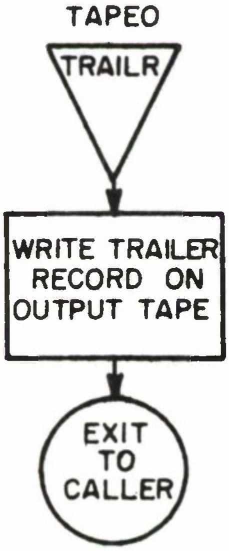
If entry is made at TRAILER, the first NNN characters of the output image are written on the output tape.


TAPED
OUTPUT
HISTORY TAPE TO BE MADE ?
NO
YES
FIRST CALL FOR HISTORY TAPE?
NO
YES
SET CALL TO NOT FIRST TIME
SAVE A PIECE OF CORE ON DISK
READ HISTORY TAPE
EOF?
NO
YES
WAS FIRST RECORD EOF?
YES
NO
BACKSPACE TWICE, READ ONCE, BACKSPACE ONCE
WRITE BACK LAST RECORD OF HISTORY TAPE
RESTORE SAVED CORE
REWIND HISTORY TAPE
WRITE OUTPUT IMAGE ONTO HISTORY TAPE
END OF REEL ?
YES
WAIT FOR OPERATOR ACTION
NO
SENSE SWITCH C ON ?
YES
PRINT THE OUTPUT IMAGE
NO
WRITE THE OUTPUT IMAGE ONTO TAPE
END OF REEL ?
NO
EXIT TO CALLER
YES
PRINT END-OF-REEL MESSAGE AND WAIT FOR OPERATOR ACTION
READ IN SAVED HEADER FROM DISK (IF ANY)
WRITE HEADER ON NEW TAPE
EXIT TO CALLER

1B-26,188


TAPEO
INPUT
READ A RECORD FROM TAPE 3
END-OF-FILE ?
YES
TURN ON COMDON INDICATOR
NO
EXIT TO CALLER


TAPEO
HEADER
TURN ON TAPE HEADER INDICATOR
WRITE HEADER ON OUTPUT TAPE
SAVE HEADER ON DISK
EXIT TO CALLER


TAPEO
TRAILR
WRITE TRAILER RECORD ON OUTPUT TAPE
EXIT TO CALLER

REALO

Purpose

Interpretively execute an output program to produce the desired report.

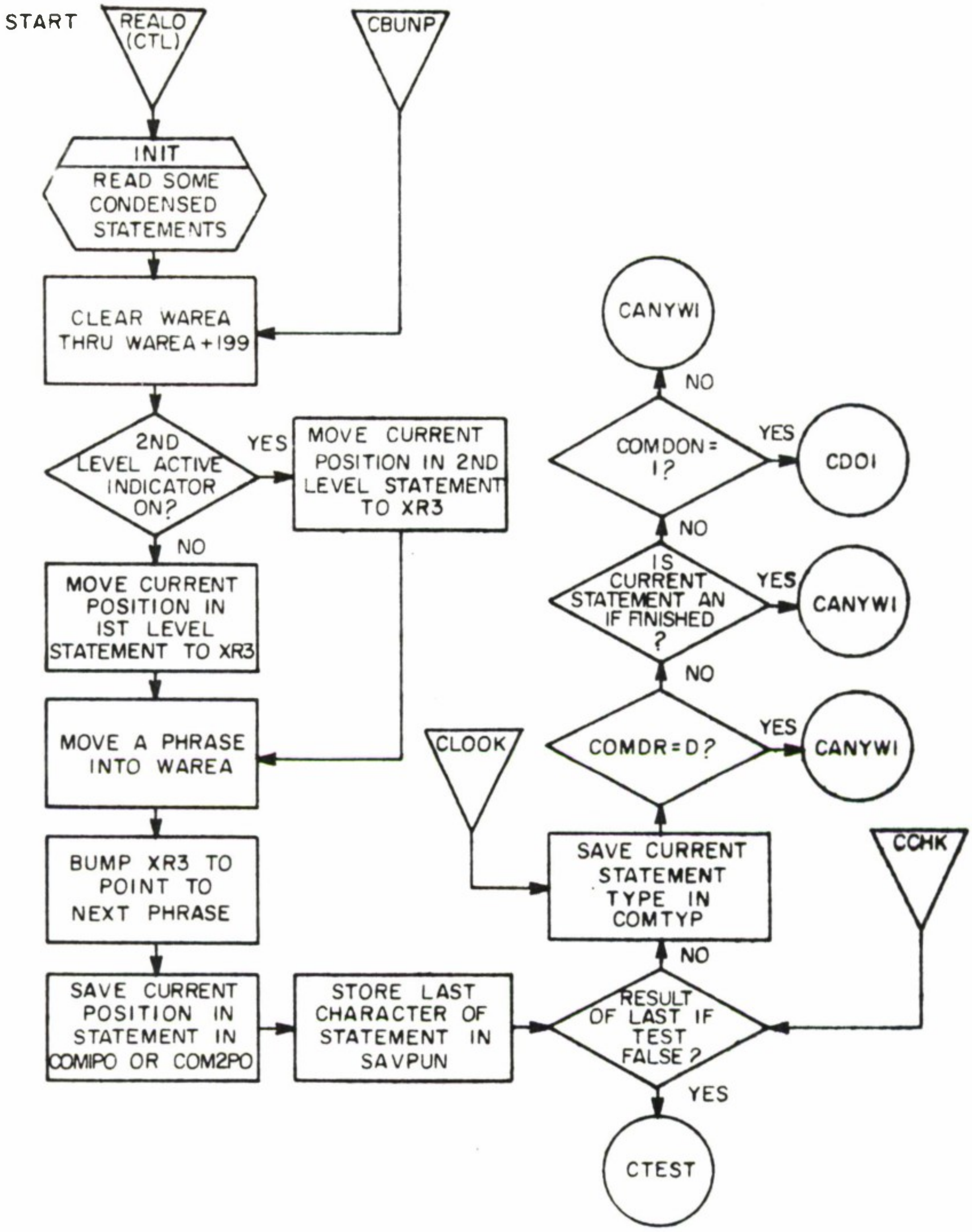
Calling Sequence

REALO is called internally from RGPOUT by

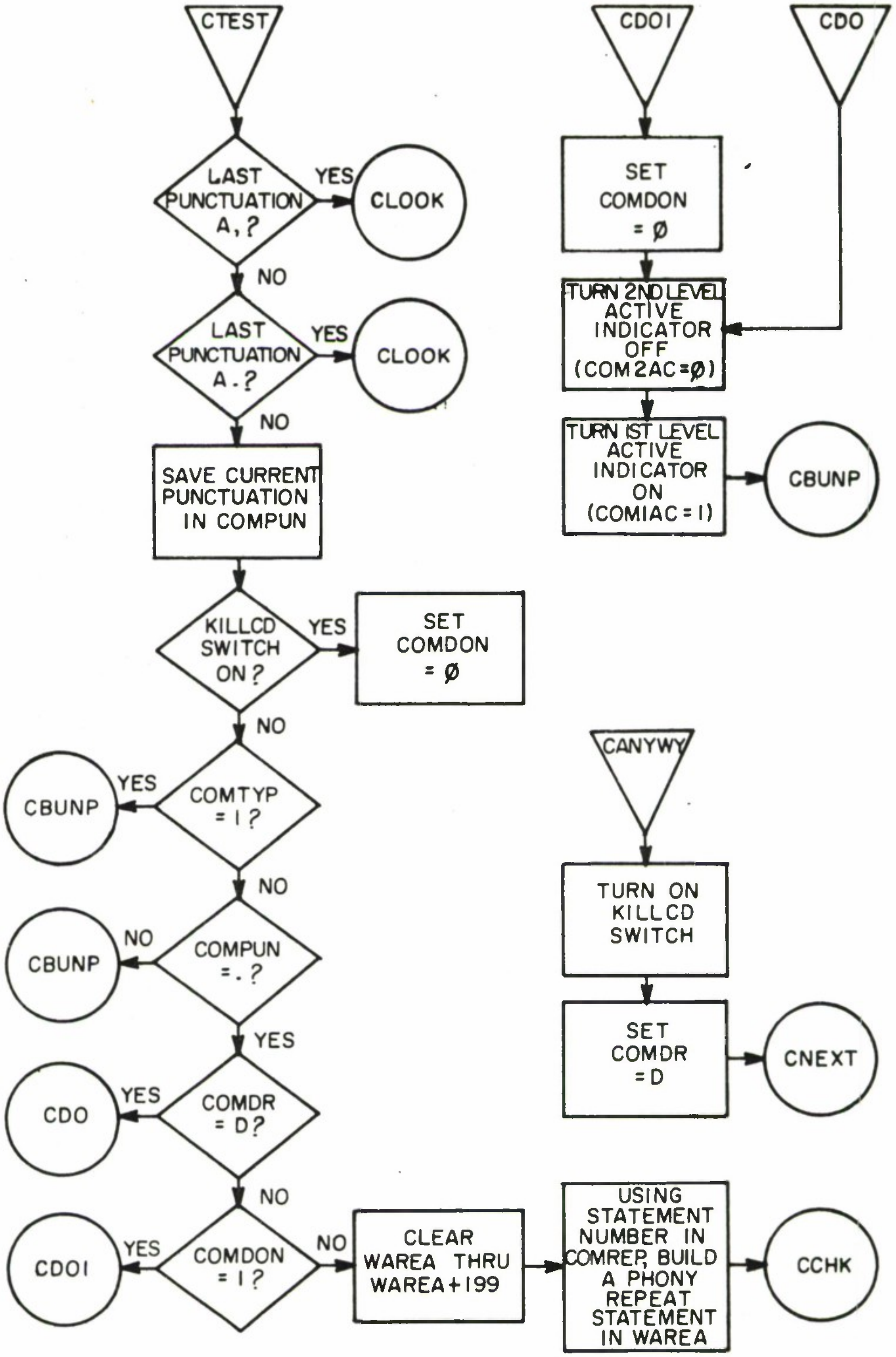
```
B     CTL
(normal return)
```

Description

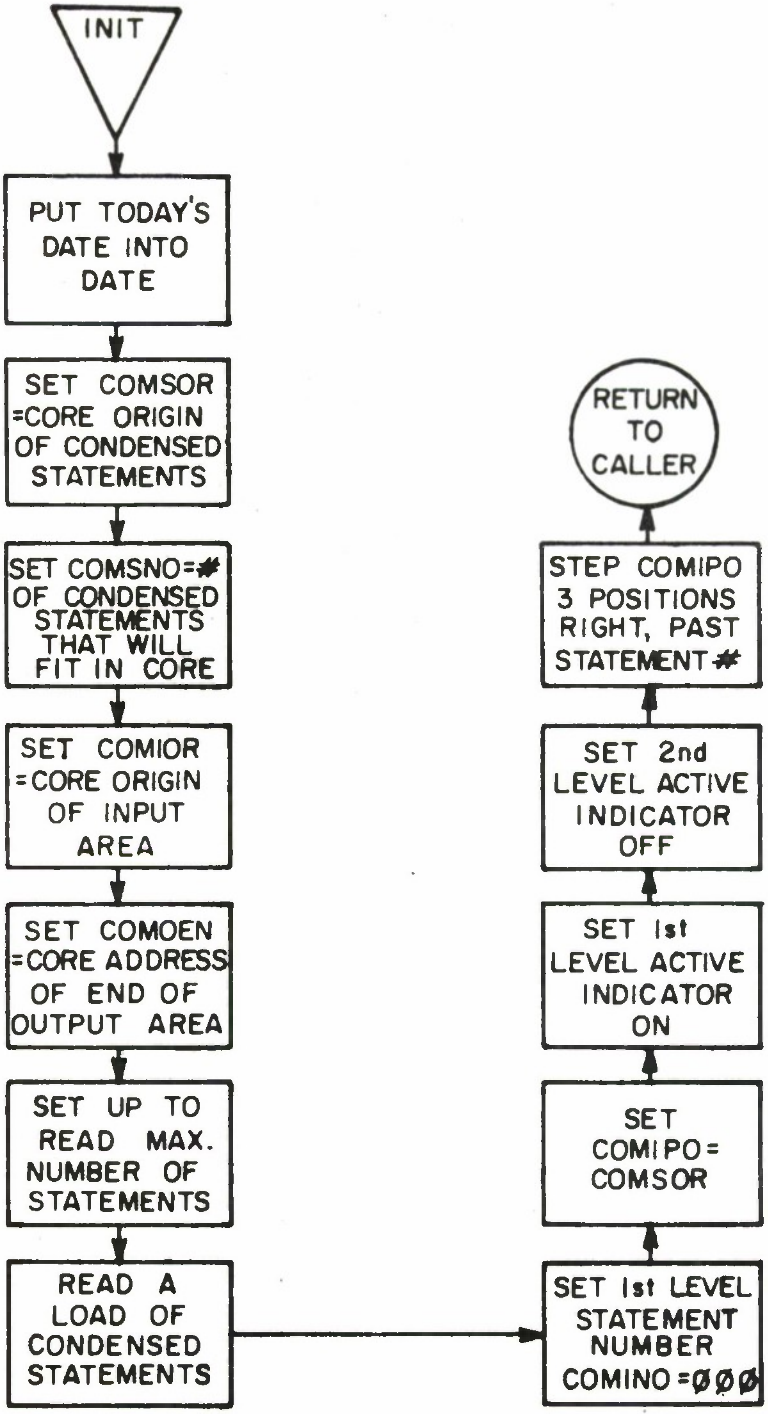
An initialize routine is first executed to read into core as much of the condensed output program as will fit. Statements are kept in core just as they appeared on disk, in 245 character records. Each statement, as it is required, is moved to a buffer at WAREA from where it is actually executed. The current statement is examined one phrase at a time. Each phrase begins with a number from 1 to 6 indicating the phrase type. Recognition of the type causes a transfer to the appropriate TYPE routine, where the phrase is completely identified and executed. Section 6.4.4 under LOD shows the phrases grouped by type, and the format of the condensed phrases. Execution of an END phrase causes control to be passed from REALO back to RGPOUT.


START
REALO (CTL)
CBUNP
INIT
READ SOME CONDENSED STATEMENTS
CLEAR WAREA THRU WAREA + 199
2ND LEVEL ACTIVE INDICATOR ON?
YES
MOVE CURRENT POSITION IN 2ND LEVEL STATEMENT TO XR3
NO
MOVE CURRENT POSITION IN IST LEVEL STATEMENT TO XR3
MOVE A PHRASE INTO WAREA
BUMP XR3 TO POINT TO NEXT PHRASE
SAVE CURRENT POSITION IN STATEMENT IN COMIPO OR COM2PO
STORE LAST CHARACTER OF STATEMENT IN SAVPUN
CANYWI
NO
COMDON = 1?
YES
CDOI
NO
IS CURRENT STATEMENT AN IF FINISHED ?
YES
CANYWI
NO
COMDR = D?
YES
CANYWI
CLOOK
SAVE CURRENT STATEMENT TYPE IN COMTYP
CCHK
NO
RESULT OF LAST IF TEST FALSE ?
YES
CTEST

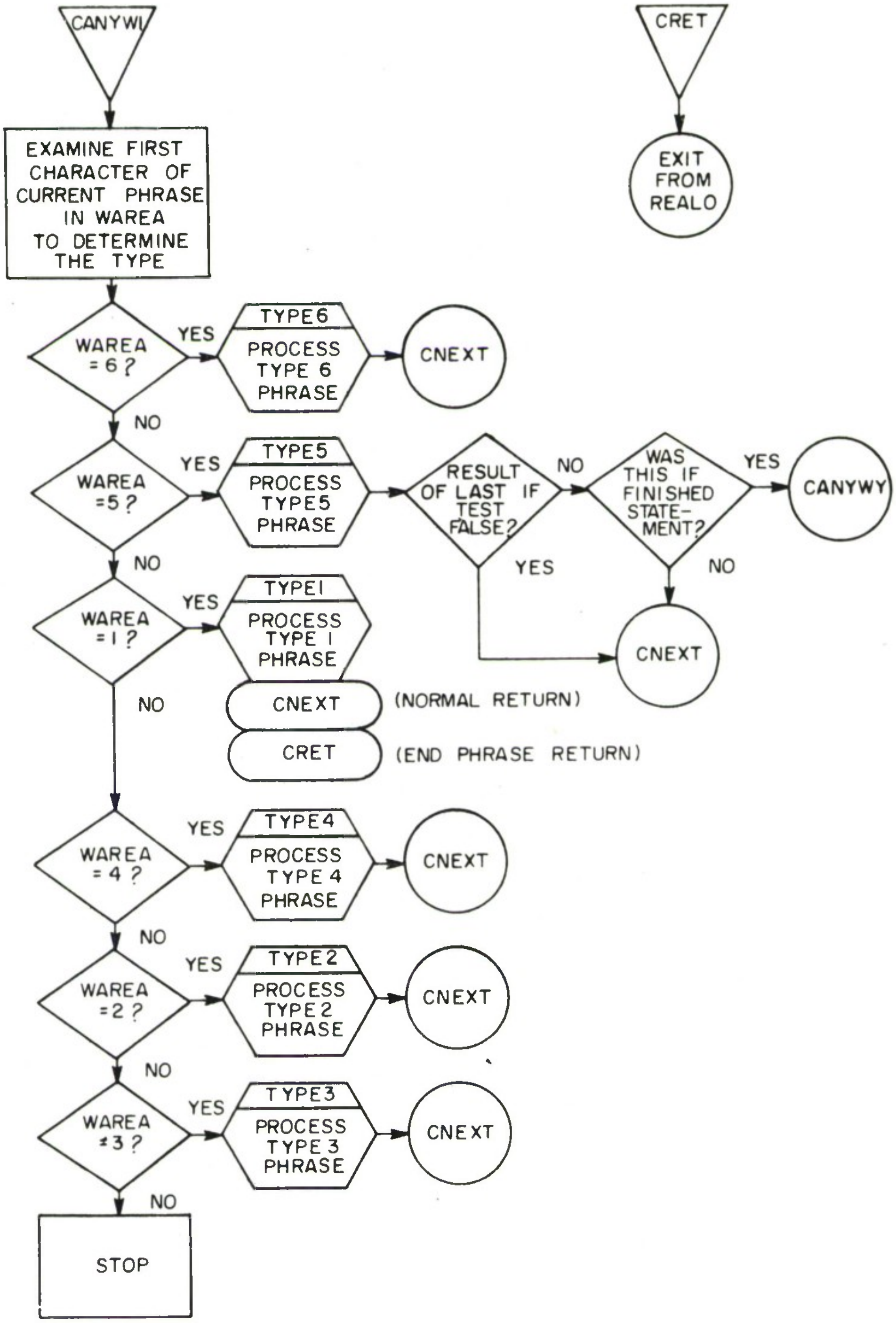
IB-26,212


CTEST
LAST PUNCTUATION A, ?
YES
CLOOK
NO
LAST PUNCTUATION A . ?
YES
CLOOK
NO
SAVE CURRENT PUNCTUATION IN COMPUN
KILLCD SWITCH ON ?
YES
SET COMDON = Ø
NO
COMTYP = 1 ?
YES
CBUNP
NO
COMPUN = . ?
NO
CBUNP
YES
COMDR = D?
YES
CDO
NO
COMDON = 1 ?
YES
CDO1
NO
CLEAR WAREA THRU WAREA+199
USING STATEMENT NUMBER IN COMREP, BUILD A PHONY REPEAT STATEMENT IN WAREA
CCHK
CDO1
SET COMDON = Ø
CDO
TURN 2ND LEVEL ACTIVE INDICATOR OFF (COM2AC=Ø)
TURN 1ST LEVEL ACTIVE INDICATOR ON (COM1AC = 1)
CBUNP
CANYWY
TURN ON KILLCD SWITCH
SET COMDR = D
CNEXT

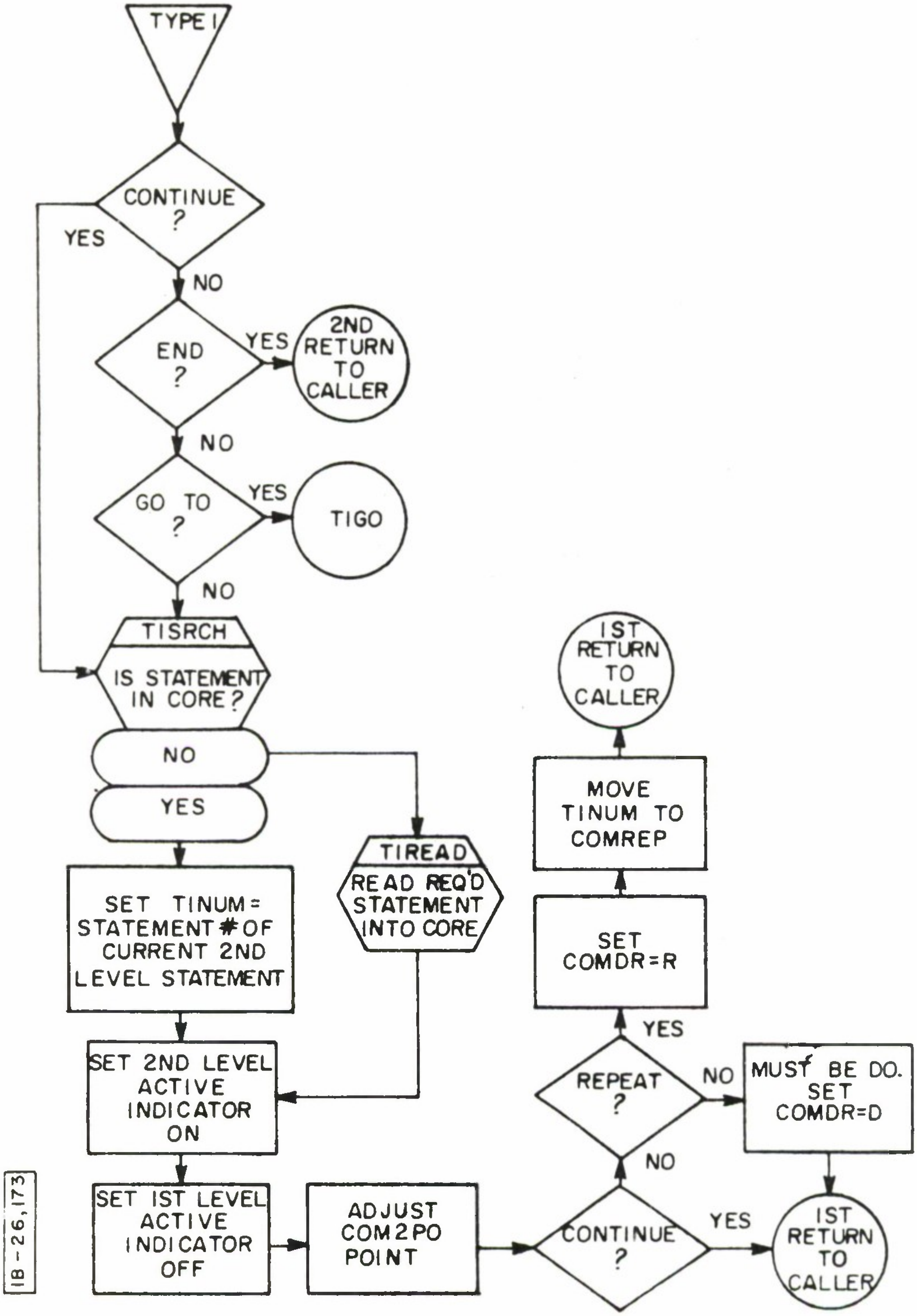
IB-26,202

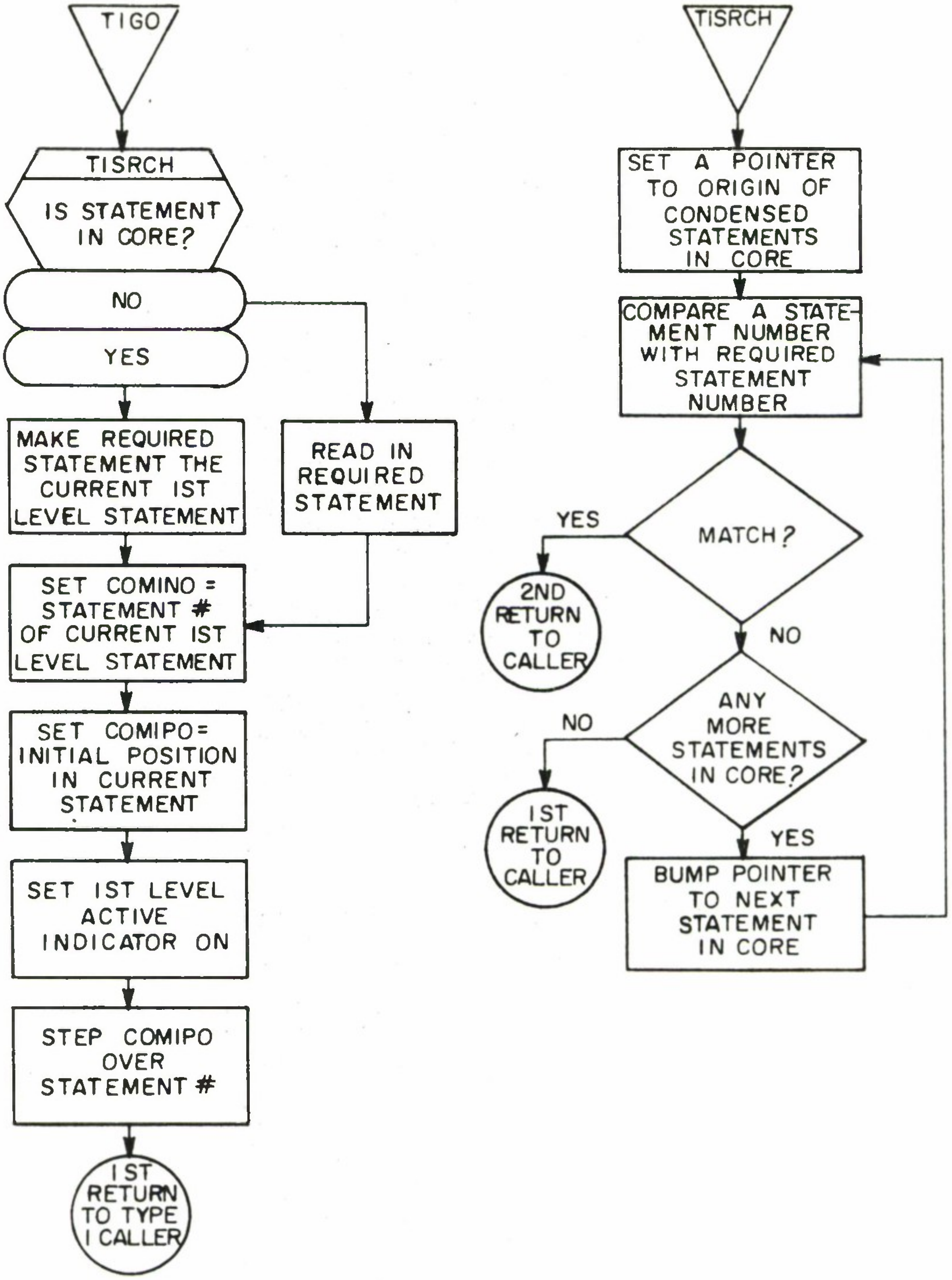

INIT
PUT TODAY'S DATE INTO DATE
SET COMSOR =CORE ORIGIN OF CONDENSED STATEMENTS
SET COMSNO=# OF CONDENSED STATEMENTS THAT WILL FIT IN CORE
SET COMIOR =CORE ORIGIN OF INPUT AREA
SET COMOEN =CORE ADDRESS OF END OF OUTPUT AREA
SET UP TO READ MAX. NUMBER OF STATEMENTS
READ A LOAD OF CONDENSED STATEMENTS
SET 1st LEVEL STATEMENT NUMBER COMINO =ØØØ
SET COMIPO = COMSOR
SET 1st LEVEL ACTIVE INDICATOR ON
SET 2nd LEVEL ACTIVE INDICATOR OFF
STEP COMIPO 3 POSITIONS RIGHT, PAST STATEMENT #
RETURN TO CALLER

IB-26,201

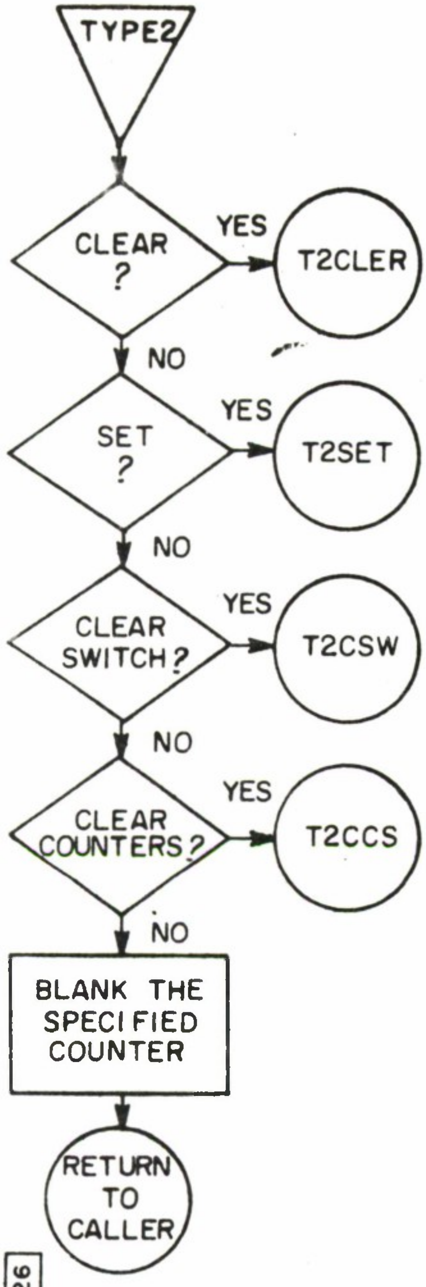

CANYWI
CRET
EXIT FROM REALO
EXAMINE FIRST CHARACTER OF CURRENT PHRASE IN WAREA TO DETERMINE THE TYPE
WAREA =6?
YES
TYPE6
PROCESS TYPE 6 PHRASE
CNEXT
NO
WAREA =5?
YES
TYPE5
PROCESS TYPE5 PHRASE
RESULT OF LAST IF TEST FALSE?
NO
WAS THIS IF FINISHED STATE-MENT?
YES
CANYWY
YES
NO
CNEXT
NO
WAREA =1?
YES
TYPE1
PROCESS TYPE 1 PHRASE
CNEXT
(NORMAL RETURN)
CRET
(END PHRASE RETURN)
NO
WAREA =4?
YES
TYPE4
PROCESS TYPE 4 PHRASE
CNEXT
NO
WAREA =2?
YES
TYPE2
PROCESS TYPE2 PHRASE
CNEXT
NO
WAREA ≠3?
YES
TYPE3
PROCESS TYPE 3 PHRASE
CNEXT
NO
STOP

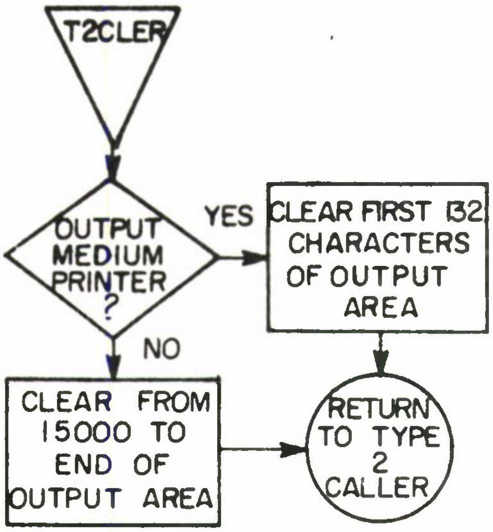
18-26,172

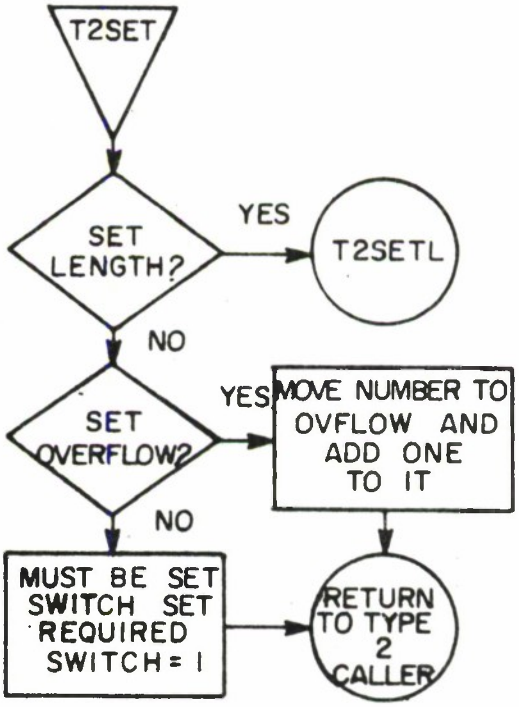

TYPE I
CONTINUE ?
YES
NO
END ?
YES
2ND RETURN TO CALLER
NO
GO TO ?
YES
TIGO
NO
TISRCH
IS STATEMENT IN CORE ?
NO
YES
SET TINUM = STATEMENT # OF CURRENT 2ND LEVEL STATEMENT
TIREAD
READ REQ'D STATEMENT INTO CORE
SET 2ND LEVEL ACTIVE INDICATOR ON
SET IST LEVEL ACTIVE INDICATOR OFF
ADJUST COM2PO POINT
CONTINUE ?
YES
IST RETURN TO CALLER
NO
REPEAT ?
YES
SET COMDR=R
MOVE TINUM TO COMREP
IST RETURN TO CALLER
NO
MUST BE DO. SET COMDR=D
IST RETURN TO CALLER
IB-26,173


TIGO
TISRCH
IS STATEMENT IN CORE?
NO
YES
MAKE REQUIRED STATEMENT THE CURRENT IST LEVEL STATEMENT
READ IN REQUIRED STATEMENT
SET COMINO = STATEMENT # OF CURRENT IST LEVEL STATEMENT
SET COMIPO= INITIAL POSITION IN CURRENT STATEMENT
SET IST LEVEL ACTIVE INDICATOR ON
STEP COMIPO OVER STATEMENT #
I ST RETURN TO TYPE I CALLER
TISRCH
SET A POINTER TO ORIGIN OF CONDENSED STATEMENTS IN CORE
COMPARE A STATEMENT NUMBER WITH REQUIRED STATEMENT NUMBER
MATCH?
YES
2ND RETURN TO CALLER
NO
ANY MORE STATEMENTS IN CORE?
NO
I ST RETURN TO CALLER
YES
BUMP POINTER TO NEXT STATEMENT IN CORE

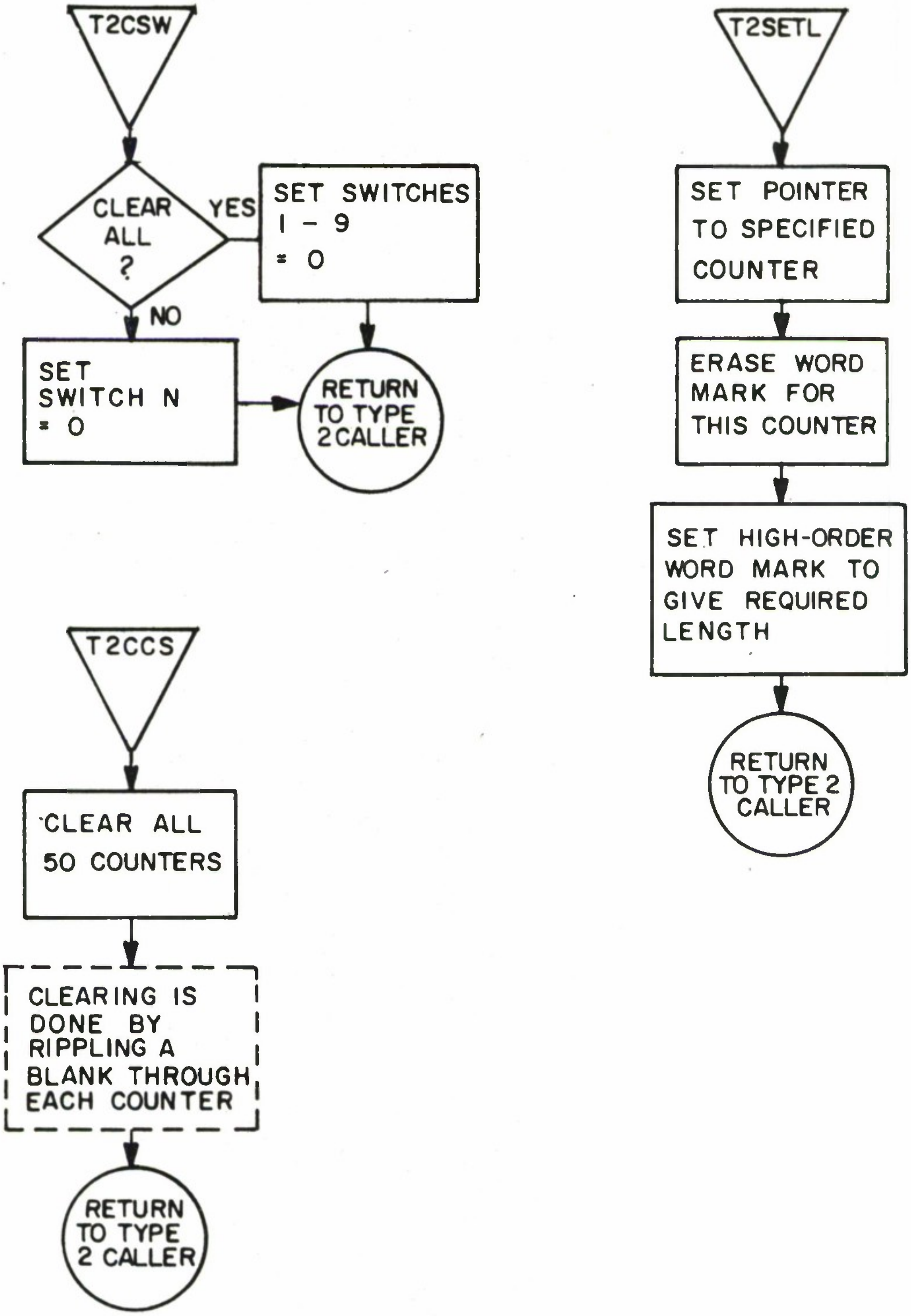
IB-26,225


TYPE2
CLEAR ?
YES
T2CLER
NO
SET ?
YES
T2SET
NO
CLEAR SWITCH ?
YES
T2CSW
NO
CLEAR COUNTERS ?
YES
T2CCS
NO
BLANK THE SPECIFIED COUNTER
RETURN TO CALLER

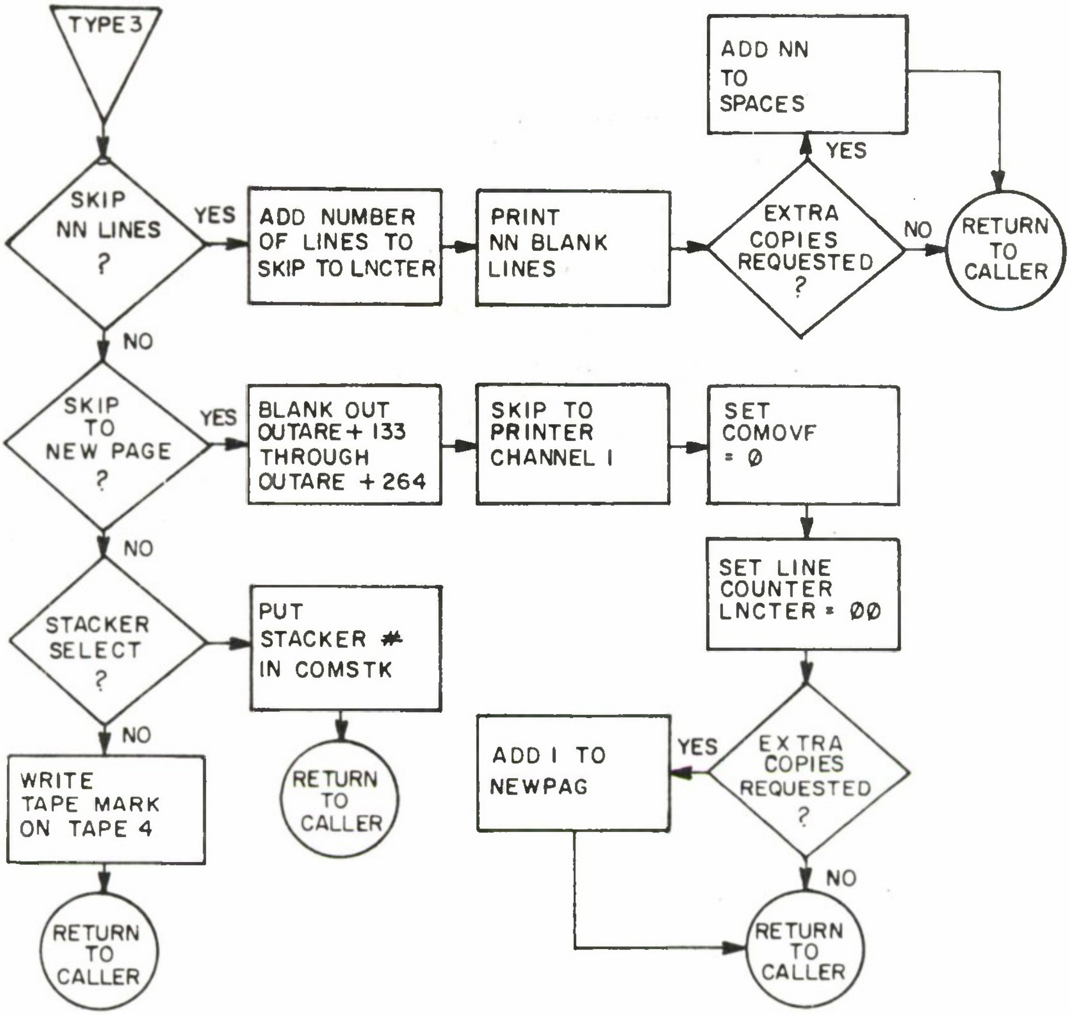

T2CLER
OUTPUT MEDIUM PRINTER ?
YES
CLEAR FIRST 132 CHARACTERS OF OUTPUT AREA
NO
CLEAR FROM 15000 TO END OF OUTPUT AREA
RETURN TO TYPE 2 CALLER


T2SET
SET LENGTH?
YES
T2SETL
NO
SET OVERFLOW?
YES
MOVE NUMBER TO OVFLOW AND ADD ONE TO IT
NO
MUST BE SET SWITCH SET REQUIRED SWITCH = 1
RETURN TO TYPE 2 CALLER

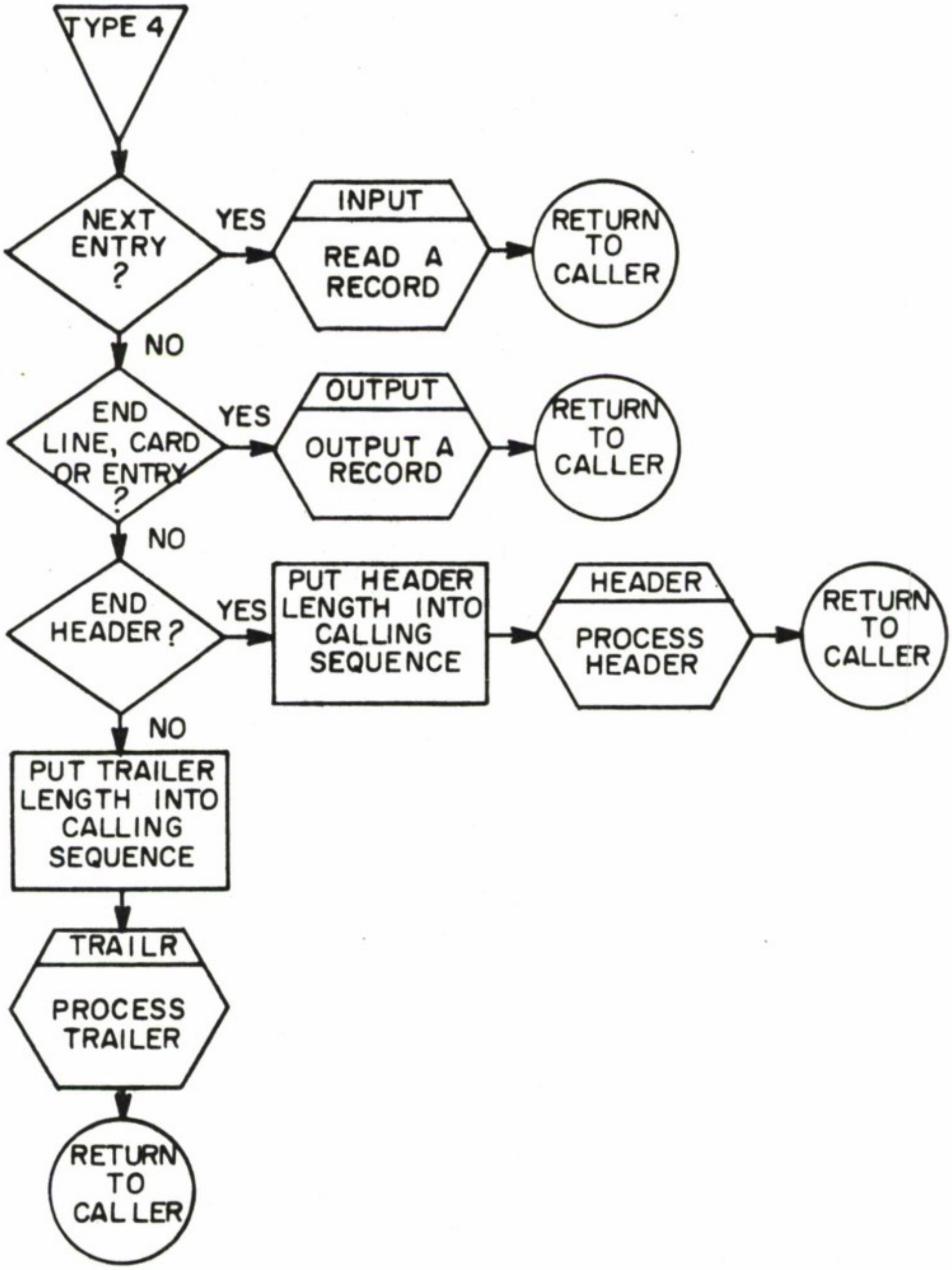
IB-26,226


T2CSW
CLEAR ALL ?
YES
NO
SET SWITCHES 1 – 9 = 0
SET SWITCH N = 0
RETURN TO TYPE 2CALLER
T2SETL
SET POINTER TO SPECIFIED COUNTER
ERASE WORD MARK FOR THIS COUNTER
SET HIGH-ORDER WORD MARK TO GIVE REQUIRED LENGTH
RETURN TO TYPE 2 CALLER
T2CCS
CLEAR ALL 50 COUNTERS
CLEARING IS DONE BY RIPPLING A BLANK THROUGH EACH COUNTER
RETURN TO TYPE 2 CALLER

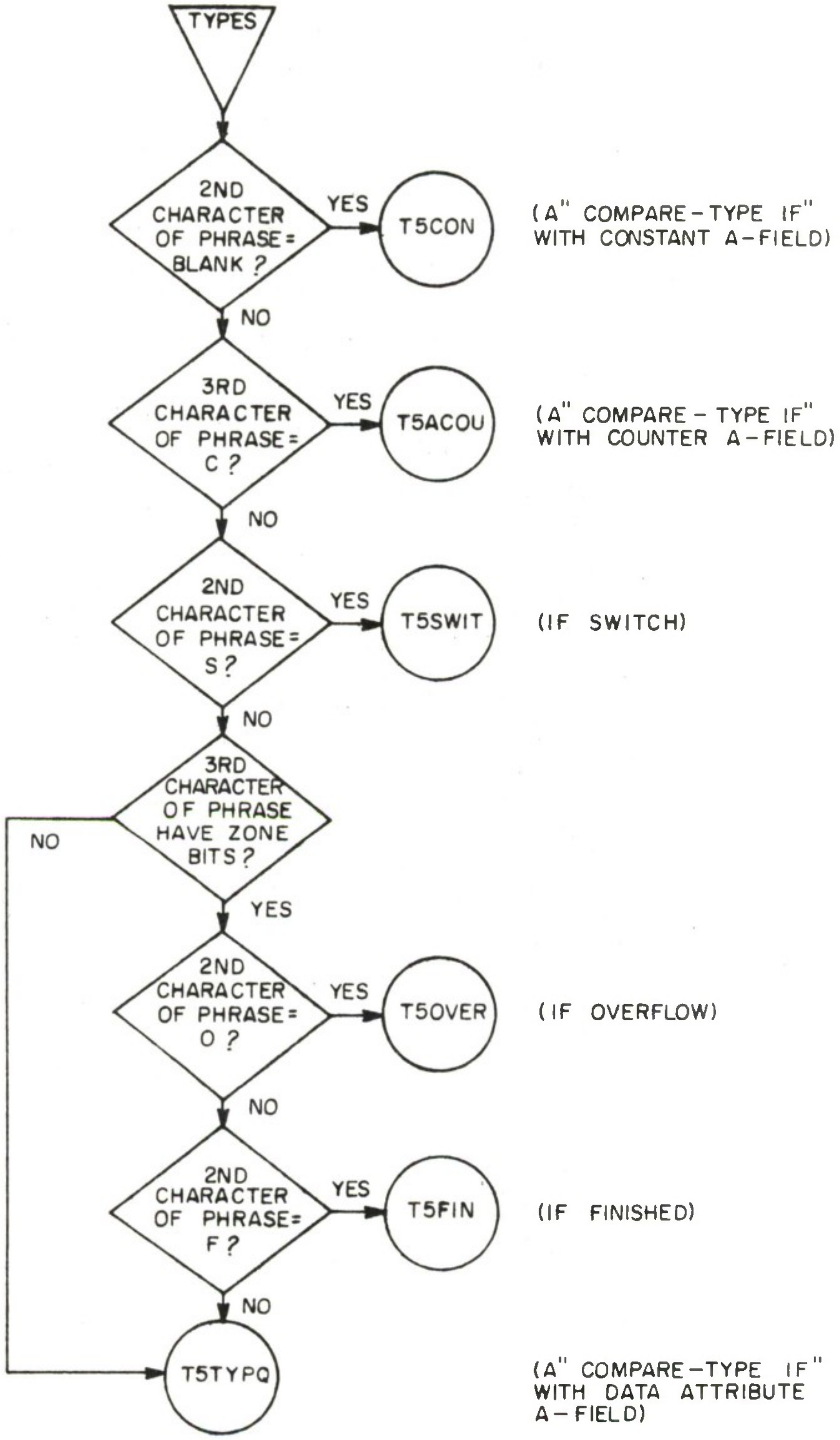
IB–26,271


TYPE 3
SKIP NN LINES ?
YES
ADD NUMBER OF LINES TO SKIP TO LNCTER
PRINT NN BLANK LINES
EXTRA COPIES REQUESTED ?
YES
ADD NN TO SPACES
NO
RETURN TO CALLER
NO
SKIP TO NEW PAGE ?
YES
BLANK OUT OUTARE + 133 THROUGH OUTARE + 264
SKIP TO PRINTER CHANNEL 1
SET COMOVF = Ø
SET LINE COUNTER LNCTER = ØØ
EXTRA COPIES REQUESTED ?
YES
ADD 1 TO NEWPAG
NO
RETURN TO CALLER
NO
STACKER SELECT ?
PUT STACKER # IN COMSTK
RETURN TO CALLER
NO
WRITE TAPE MARK ON TAPE 4
RETURN TO CALLER

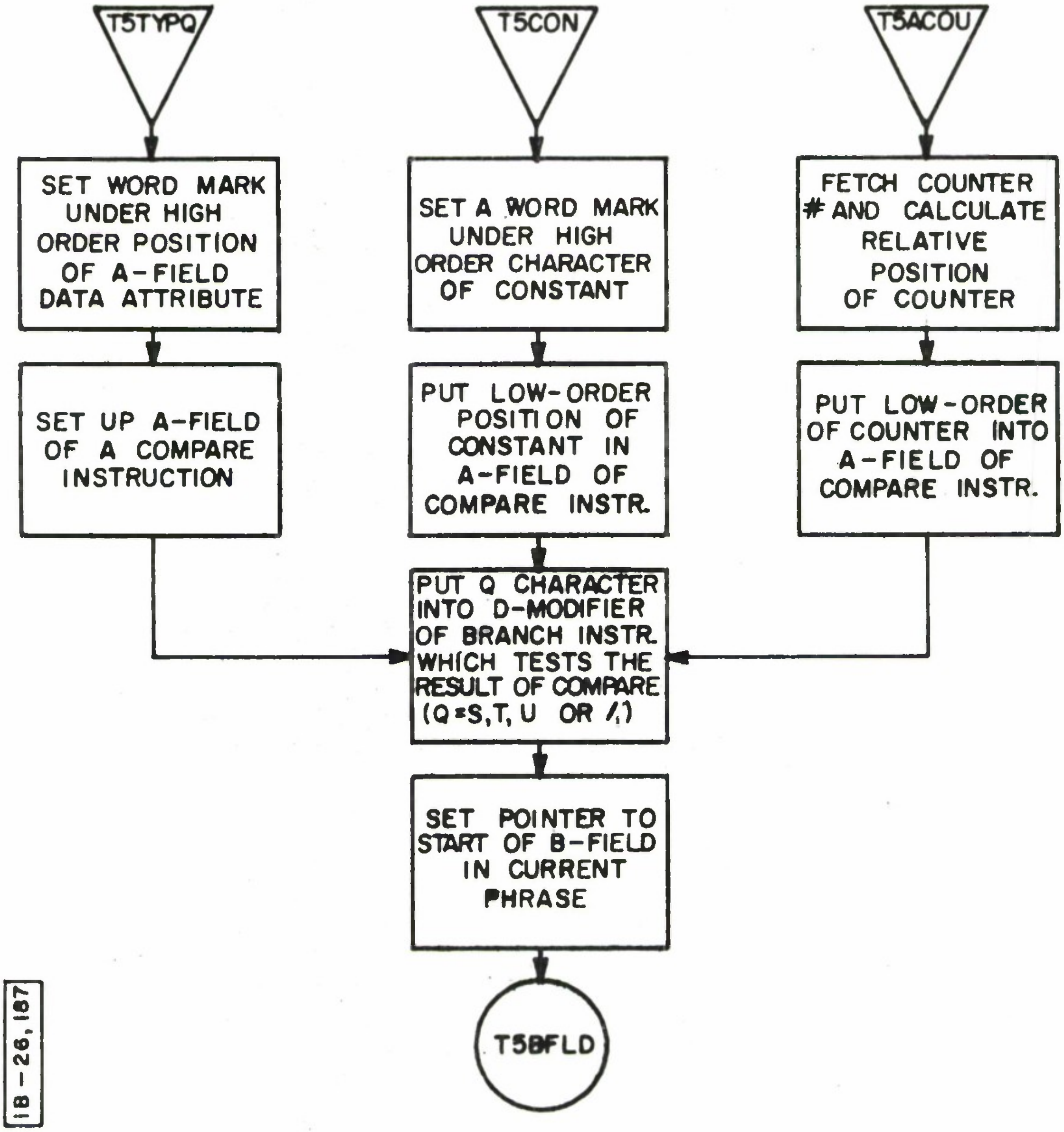
IB-26,272


TYPE 4
NEXT ENTRY ?
YES
INPUT
READ A RECORD
RETURN TO CALLER
NO
END LINE, CARD OR ENTRY ?
YES
OUTPUT
OUTPUT A RECORD
RETURN TO CALLER
NO
END HEADER ?
YES
PUT HEADER LENGTH INTO CALLING SEQUENCE
HEADER
PROCESS HEADER
RETURN TO CALLER
NO
PUT TRAILER LENGTH INTO CALLING SEQUENCE
TRAILR
PROCESS TRAILER
RETURN TO CALLER

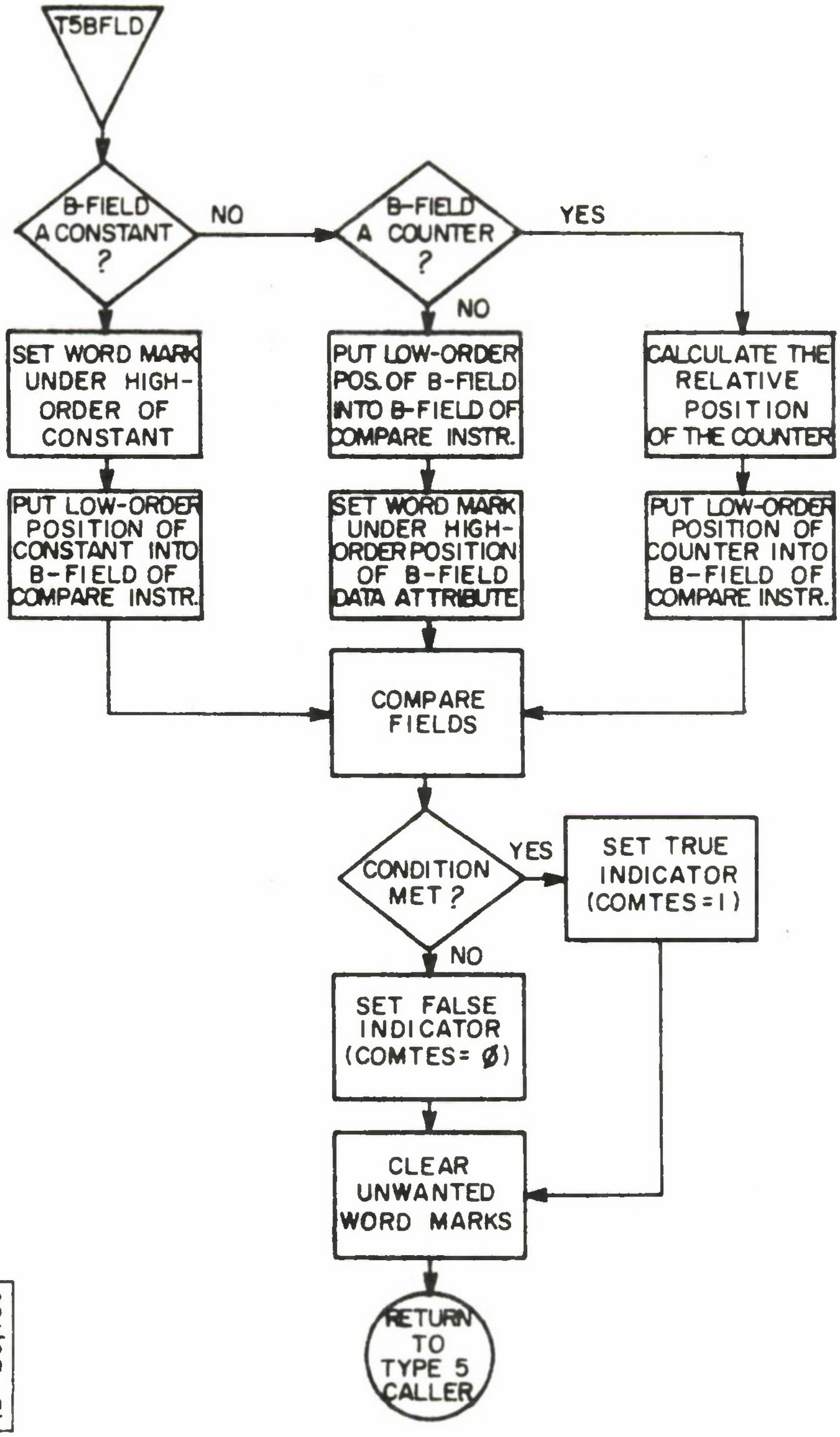
IB-26,218


TYPES
2ND CHARACTER OF PHRASE = BLANK ?
YES
T5CON
(A" COMPARE-TYPE IF" WITH CONSTANT A-FIELD)
NO
3RD CHARACTER OF PHRASE = C ?
YES
T5ACOU
(A" COMPARE - TYPE IF" WITH COUNTER A-FIELD)
NO
2ND CHARACTER OF PHRASE = S ?
YES
T5SWIT
(IF SWITCH)
NO
3RD CHARACTER OF PHRASE HAVE ZONE BITS ?
NO
YES
2ND CHARACTER OF PHRASE = O ?
YES
T5OVER
(IF OVERFLOW)
NO
2ND CHARACTER OF PHRASE = F ?
YES
T5FIN
(IF FINISHED)
NO
T5TYPQ
(A" COMPARE-TYPE IF" WITH DATA ATTRIBUTE A-FIELD)

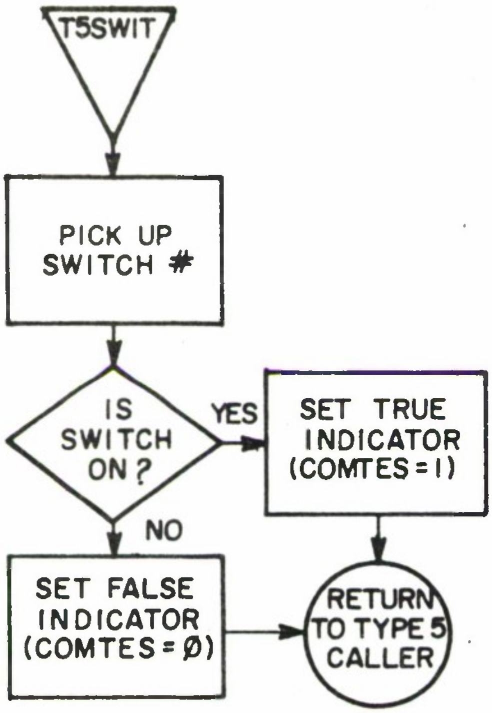
1B-26,217

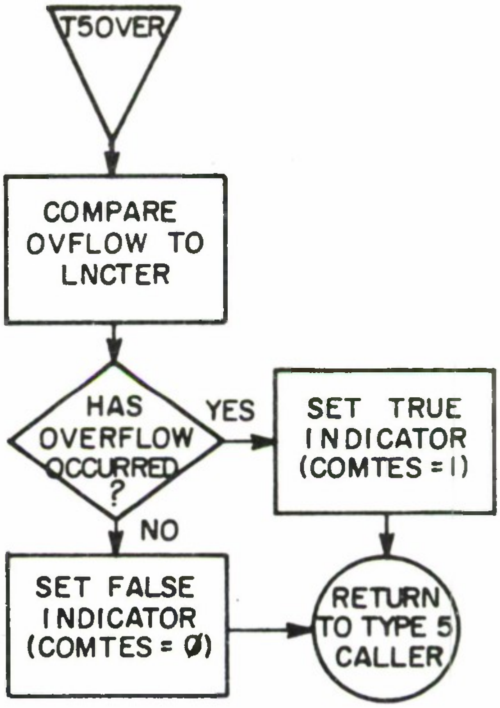

T5TYPQ
T5CON
T5ACOU
SET WORD MARK UNDER HIGH ORDER POSITION OF A-FIELD DATA ATTRIBUTE
SET A WORD MARK UNDER HIGH ORDER CHARACTER OF CONSTANT
FETCH COUNTER # AND CALCULATE RELATIVE POSITION OF COUNTER
SET UP A-FIELD OF A COMPARE INSTRUCTION
PUT LOW-ORDER POSITION OF CONSTANT IN A-FIELD OF COMPARE INSTR.
PUT LOW-ORDER OF COUNTER INTO A-FIELD OF COMPARE INSTR.
PUT Q CHARACTER INTO D-MODIFIER OF BRANCH INSTR. WHICH TESTS THE RESULT OF COMPARE (Q=S,T,U OR /)
SET POINTER TO START OF B-FIELD IN CURRENT PHRASE
T5BFLD

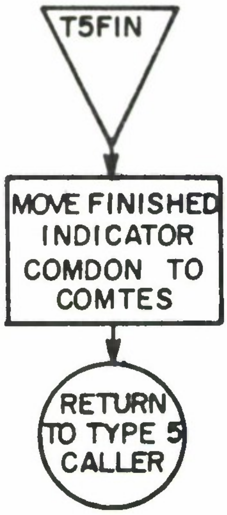
IB-26,187


T5BFLD
B-FIELD A CONSTANT ?
NO
B-FIELD A COUNTER ?
YES
NO
SET WORD MARK UNDER HIGH-ORDER OF CONSTANT
PUT LOW-ORDER POS. OF B-FIELD INTO B-FIELD OF COMPARE INSTR.
CALCULATE THE RELATIVE POSITION OF THE COUNTER
PUT LOW-ORDER POSITION OF CONSTANT INTO B-FIELD OF COMPARE INSTR.
SET WORD MARK UNDER HIGH-ORDER POSITION OF B-FIELD DATA ATTRIBUTE
PUT LOW-ORDER POSITION OF COUNTER INTO B-FIELD OF COMPARE INSTR.
COMPARE FIELDS
CONDITION MET ?
YES
SET TRUE INDICATOR (COMTES=1)
NO
SET FALSE INDICATOR (COMTES= Ø)
CLEAR UNWANTED WORD MARKS
RETURN TO TYPE 5 CALLER

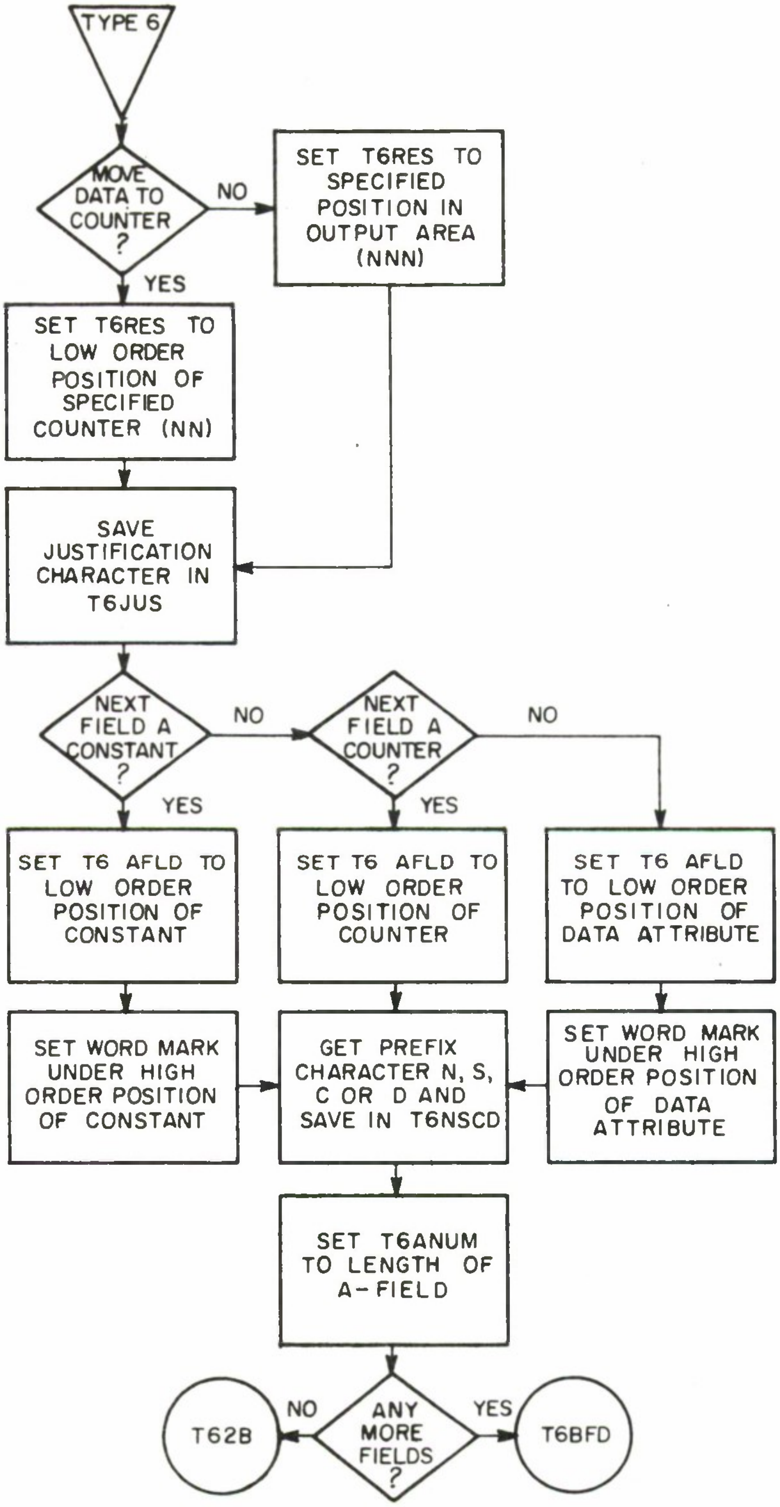
IB-26,186


T5SWIT
PICK UP SWITCH #
IS SWITCH ON ?
YES
SET TRUE INDICATOR (COMTES = 1)
NO
SET FALSE INDICATOR (COMTES = Ø)
RETURN TO TYPE 5 CALLER

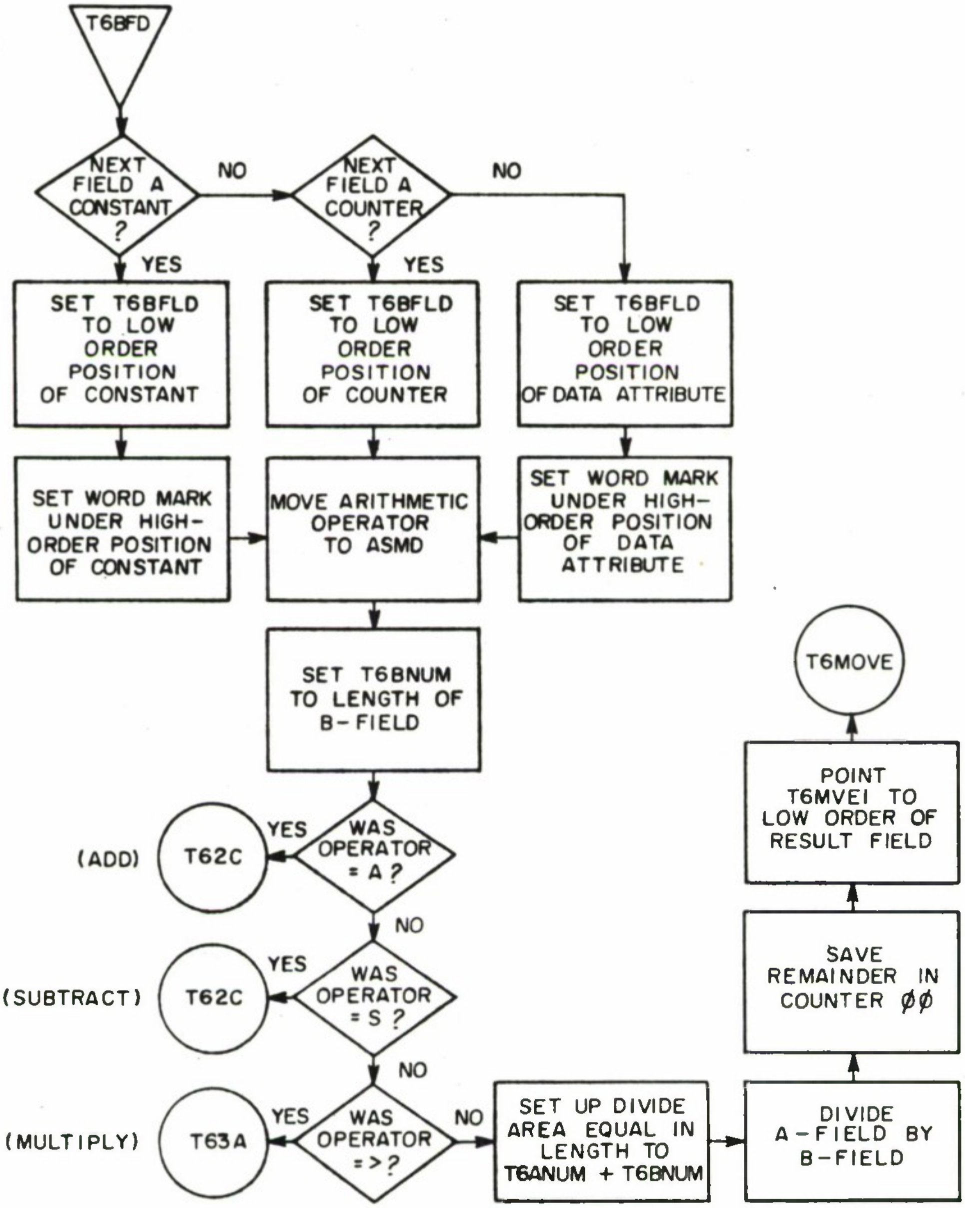

T5OVER
COMPARE OVFLOW TO LNCTER
HAS OVERFLOW OCCURRED ?
YES
SET TRUE INDICATOR (COMTES = 1)
NO
SET FALSE INDICATOR (COMTES = Ø)
RETURN TO TYPE 5 CALLER


T5FIN
MOVE FINISHED INDICATOR COMDON TO COMTES
RETURN TO TYPE 5 CALLER

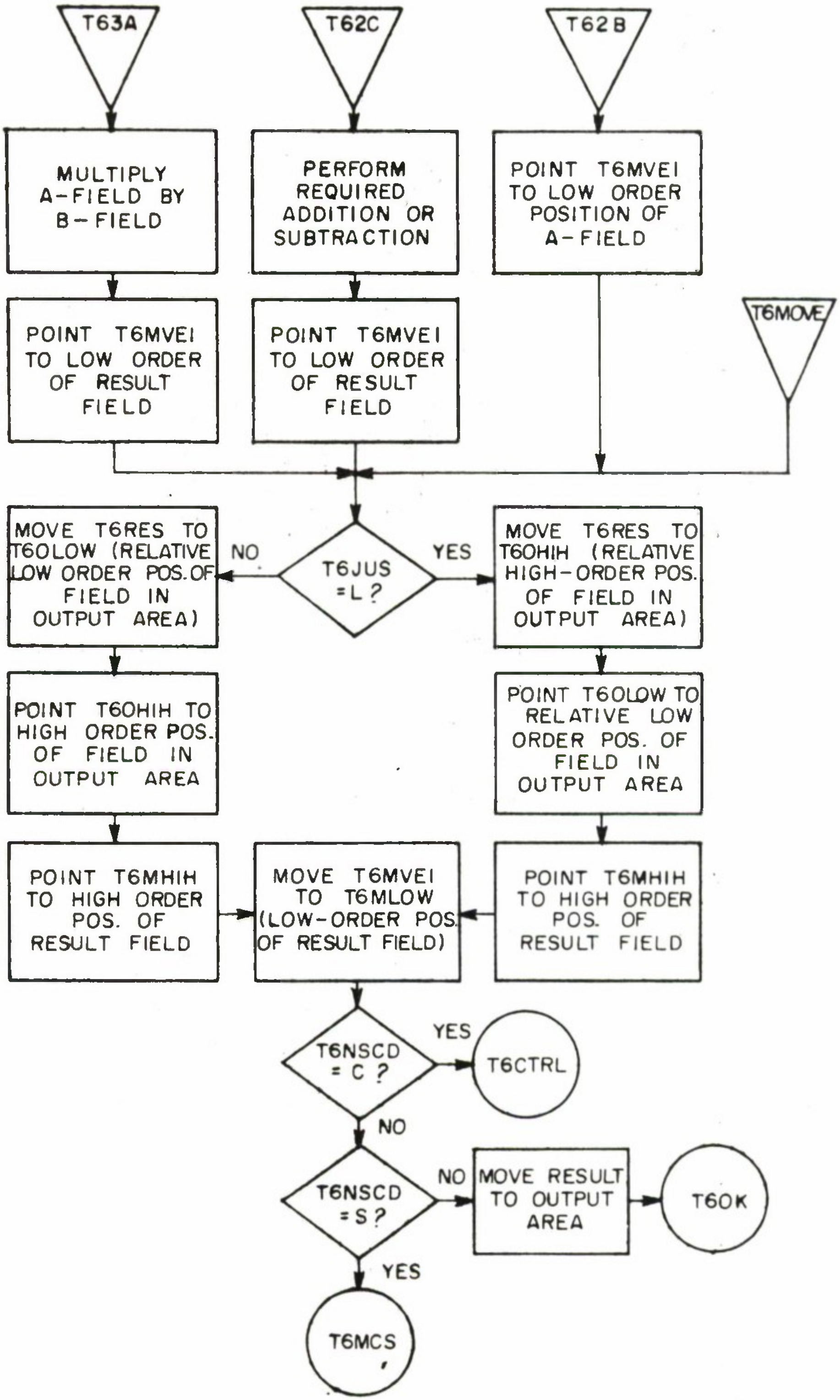
18-26,185


TYPE 6
MOVE DATA TO COUNTER ?
NO
YES
SET T6RES TO SPECIFIED POSITION IN OUTPUT AREA (NNN)
SET T6RES TO LOW ORDER POSITION OF SPECIFIED COUNTER (NN)
SAVE JUSTIFICATION CHARACTER IN T6JUS
NEXT FIELD A CONSTANT ?
NO
YES
NEXT FIELD A COUNTER ?
NO
YES
SET T6 AFLD TO LOW ORDER POSITION OF CONSTANT
SET T6 AFLD TO LOW ORDER POSITION OF COUNTER
SET T6 AFLD TO LOW ORDER POSITION OF DATA ATTRIBUTE
SET WORD MARK UNDER HIGH ORDER POSITION OF CONSTANT
GET PREFIX CHARACTER N, S, C OR D AND SAVE IN T6NSCD
SET WORD MARK UNDER HIGH ORDER POSITION OF DATA ATTRIBUTE
SET T6ANUM TO LENGTH OF A-FIELD
ANY MORE FIELDS ?
NO
YES
T62B
T6BFD

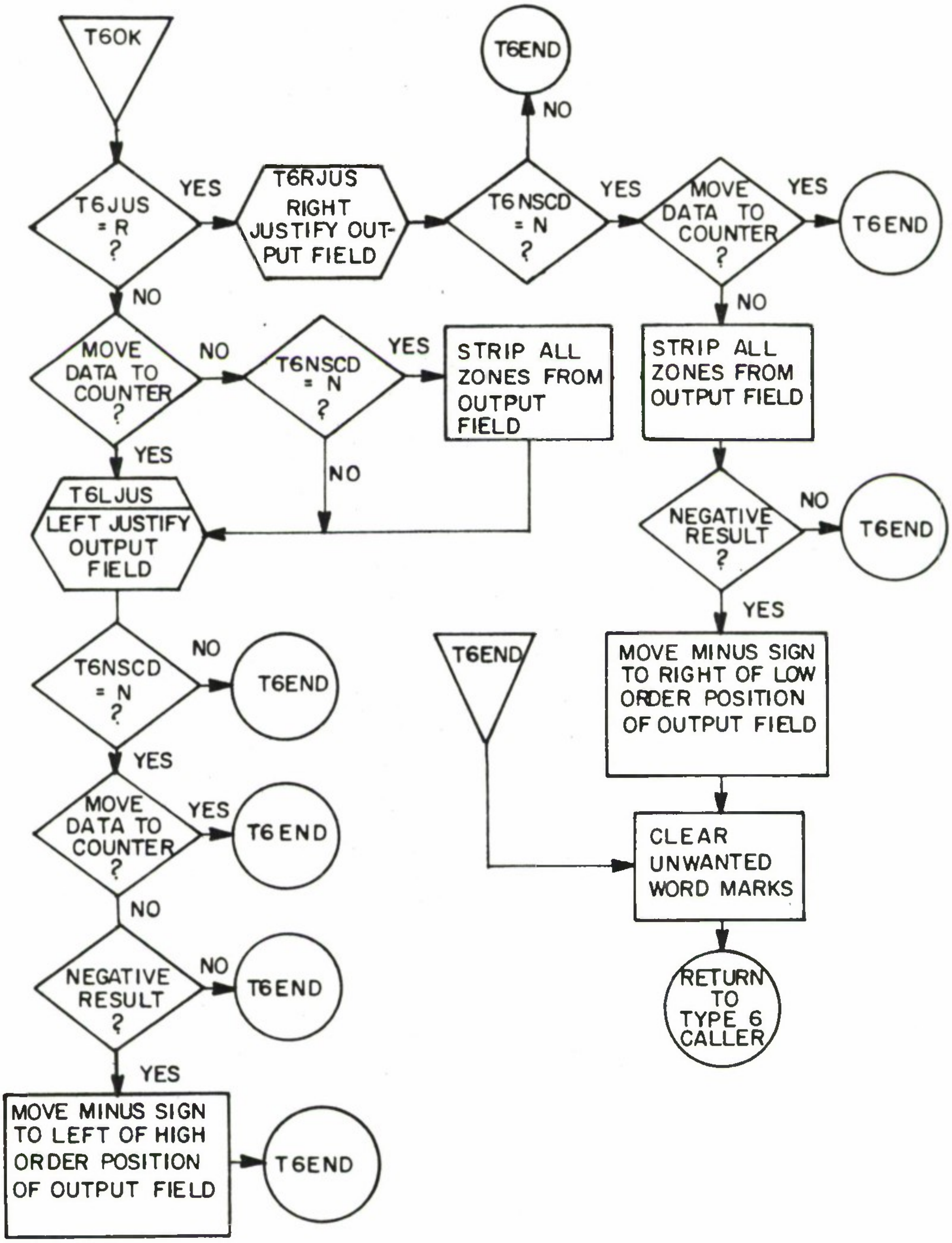
1B-26,184


T6BFD
NEXT FIELD A CONSTANT ?
NO
NEXT FIELD A COUNTER ?
NO
YES
YES
SET T6BFLD TO LOW ORDER POSITION OF CONSTANT
SET T6BFLD TO LOW ORDER POSITION OF COUNTER
SET T6BFLD TO LOW ORDER POSITION OF DATA ATTRIBUTE
SET WORD MARK UNDER HIGH-ORDER POSITION OF CONSTANT
MOVE ARITHMETIC OPERATOR TO ASMD
SET WORD MARK UNDER HIGH-ORDER POSITION OF DATA ATTRIBUTE
SET T6BNUM TO LENGTH OF B-FIELD
T6MOVE
(ADD)
T62C
YES
WAS OPERATOR = A ?
NO
POINT T6MVEI TO LOW ORDER OF RESULT FIELD
(SUBTRACT)
T62C
YES
WAS OPERATOR = S ?
NO
SAVE REMAINDER IN COUNTER ØØ
(MULTIPLY)
T63A
YES
WAS OPERATOR = > ?
NO
SET UP DIVIDE AREA EQUAL IN LENGTH TO T6ANUM + T6BNUM
DIVIDE A-FIELD BY B-FIELD

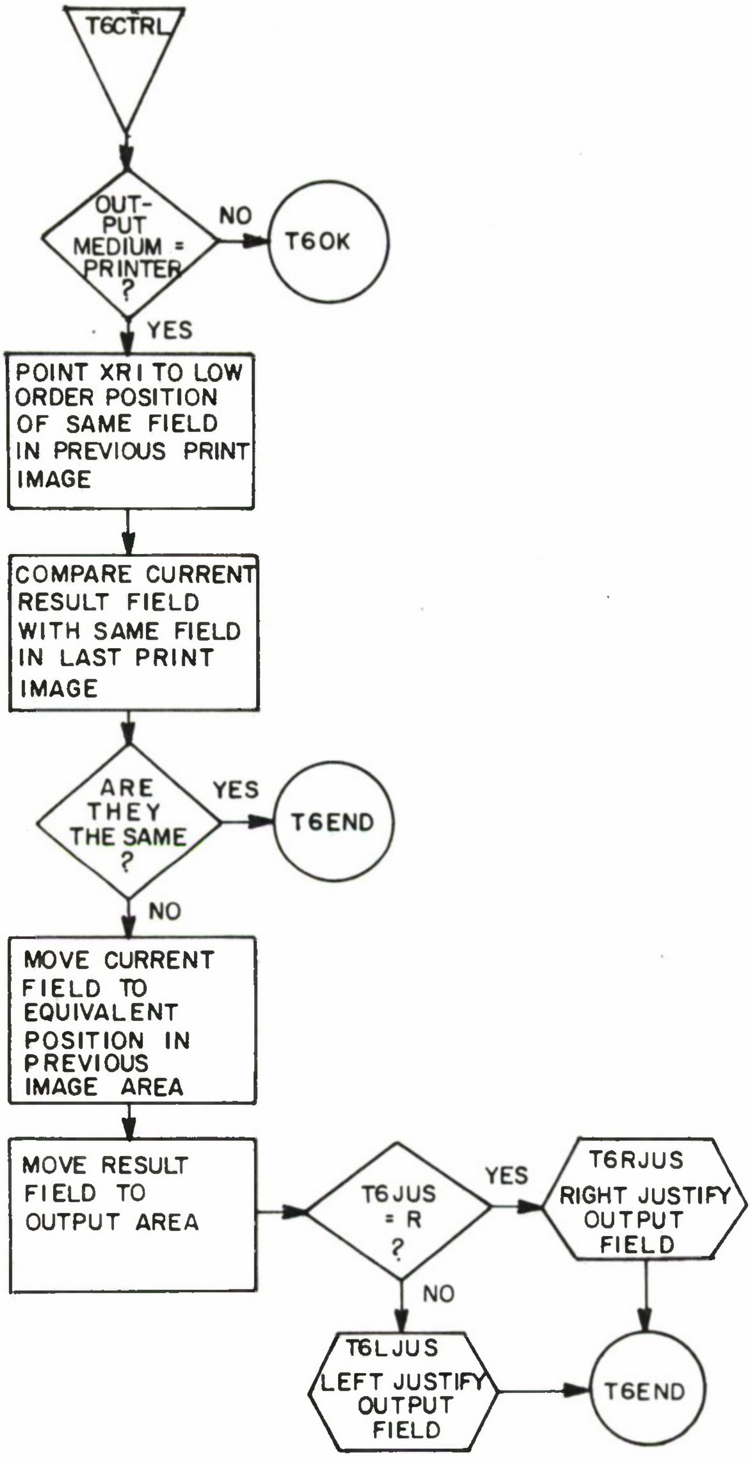
18-26,232


T63A
T62C
T62B
MULTIPLY A-FIELD BY B-FIELD
PERFORM REQUIRED ADDITION OR SUBTRACTION
POINT T6MVEI TO LOW ORDER POSITION OF A-FIELD
T6MOVE
POINT T6MVEI TO LOW ORDER OF RESULT FIELD
POINT T6MVEI TO LOW ORDER OF RESULT FIELD
T6JUS =L ?
NO
YES
MOVE T6RES TO T6OLOW (RELATIVE LOW ORDER POS. OF FIELD IN OUTPUT AREA)
MOVE T6RES TO T6OHIH (RELATIVE HIGH-ORDER POS. OF FIELD IN OUTPUT AREA)
POINT T6OHIH TO HIGH ORDER POS. OF FIELD IN OUTPUT AREA
POINT T6OLOW TO RELATIVE LOW ORDER POS. OF FIELD IN OUTPUT AREA
POINT T6MHIH TO HIGH ORDER POS. OF RESULT FIELD
MOVE T6MVEI TO T6MLOW (LOW-ORDER POS. OF RESULT FIELD)
POINT T6MHIH TO HIGH ORDER POS. OF RESULT FIELD
T6NSCD = C ?
YES
T6CTRL
NO
T6NSCD = S ?
NO
MOVE RESULT TO OUTPUT AREA
T6OK
YES
T6MCS

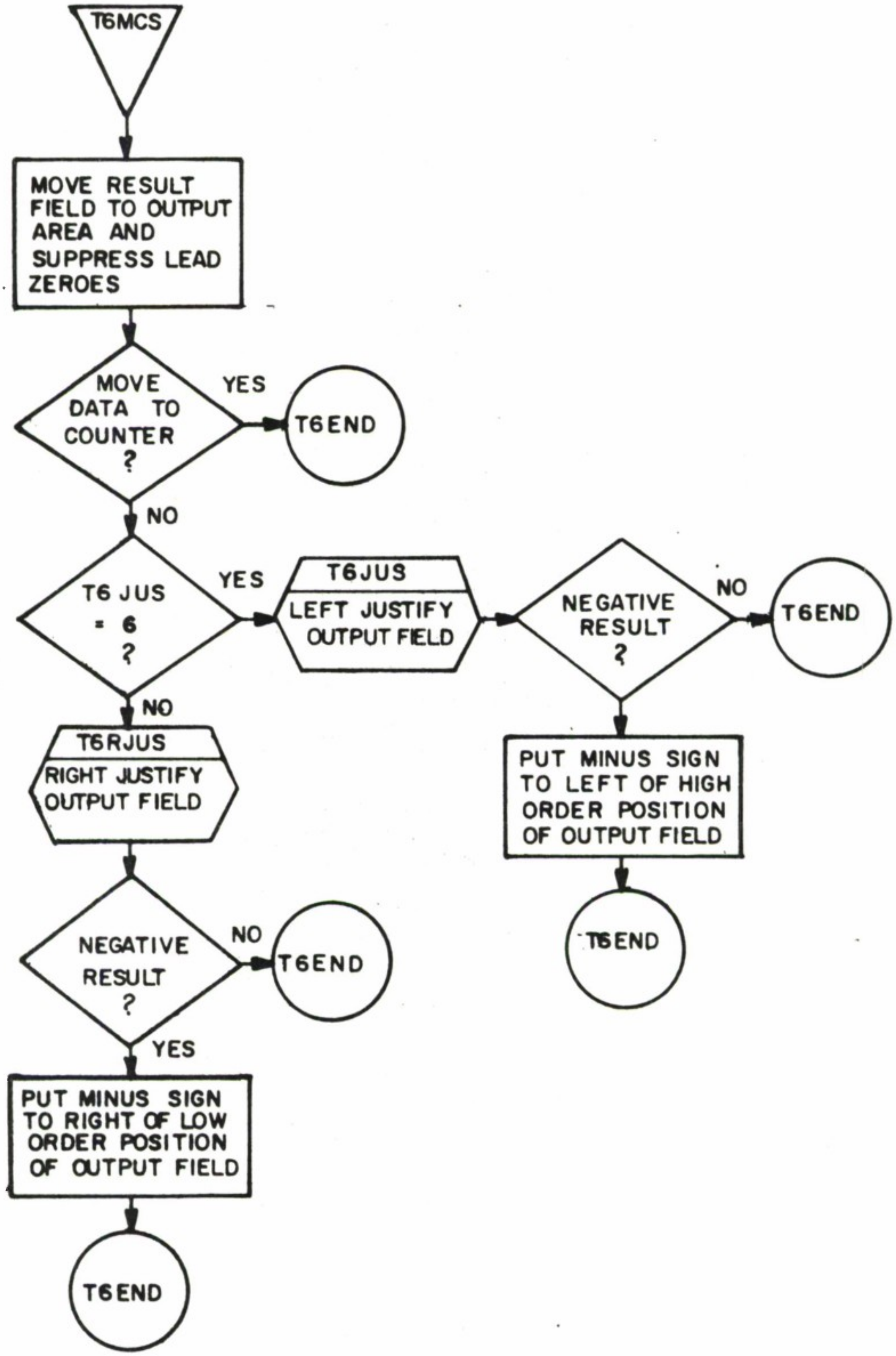
IB-26,231


T6OK
T6JUS = R ?
YES
NO
T6RJUS
RIGHT JUSTIFY OUTPUT FIELD
T6NSCD = N ?
NO
T6END
YES
MOVE DATA TO COUNTER ?
YES
T6END
NO
MOVE DATA TO COUNTER ?
NO
T6NSCD = N ?
YES
STRIP ALL ZONES FROM OUTPUT FIELD
NO
YES
STRIP ALL ZONES FROM OUTPUT FIELD
T6LJUS
LEFT JUSTIFY OUTPUT FIELD
NEGATIVE RESULT ?
NO
T6END
YES
T6NSCD = N ?
NO
T6END
YES
T6END
MOVE MINUS SIGN TO RIGHT OF LOW ORDER POSITION OF OUTPUT FIELD
MOVE DATA TO COUNTER ?
YES
T6END
NO
CLEAR UNWANTED WORD MARKS
NEGATIVE RESULT ?
NO
T6END
RETURN TO TYPE 6 CALLER
YES
MOVE MINUS SIGN TO LEFT OF HIGH ORDER POSITION OF OUTPUT FIELD
T6END

IB-26,273


T6CTRL
OUT-PUT MEDIUM = PRINTER ?
NO
T6OK
YES
POINT XRI TO LOW ORDER POSITION OF SAME FIELD IN PREVIOUS PRINT IMAGE
COMPARE CURRENT RESULT FIELD WITH SAME FIELD IN LAST PRINT IMAGE
ARE THEY THE SAME ?
YES
T6END
NO
MOVE CURRENT FIELD TO EQUIVALENT POSITION IN PREVIOUS IMAGE AREA
MOVE RESULT FIELD TO OUTPUT AREA
T6JUS = R ?
YES
T6RJUS
RIGHT JUSTIFY OUTPUT FIELD
NO
T6LJUS
LEFT JUSTIFY OUTPUT FIELD
T6END

IB-26,274


T6MCS
MOVE RESULT FIELD TO OUTPUT AREA AND SUPPRESS LEAD ZEROES
MOVE DATA TO COUNTER ?
YES
T6END
NO
T6 JUS = 6 ?
YES
T6JUS
LEFT JUSTIFY OUTPUT FIELD
NEGATIVE RESULT ?
NO
T6END
PUT MINUS SIGN TO LEFT OF HIGH ORDER POSITION OF OUTPUT FIELD
T6END
NO
T6RJUS
RIGHT JUSTIFY OUTPUT FIELD
NEGATIVE RESULT ?
NO
T6END
YES
PUT MINUS SIGN TO RIGHT OF LOW ORDER POSITION OF OUTPUT FIELD
T6END

18-26,275

REMARK

Purpose

Print or punch remarks which were stored on disk during the input phase of a REPGEN run.

Calling Sequence

REMARK is loaded and called from RGPOUT by

```
B    REMARK
(normal return)
```

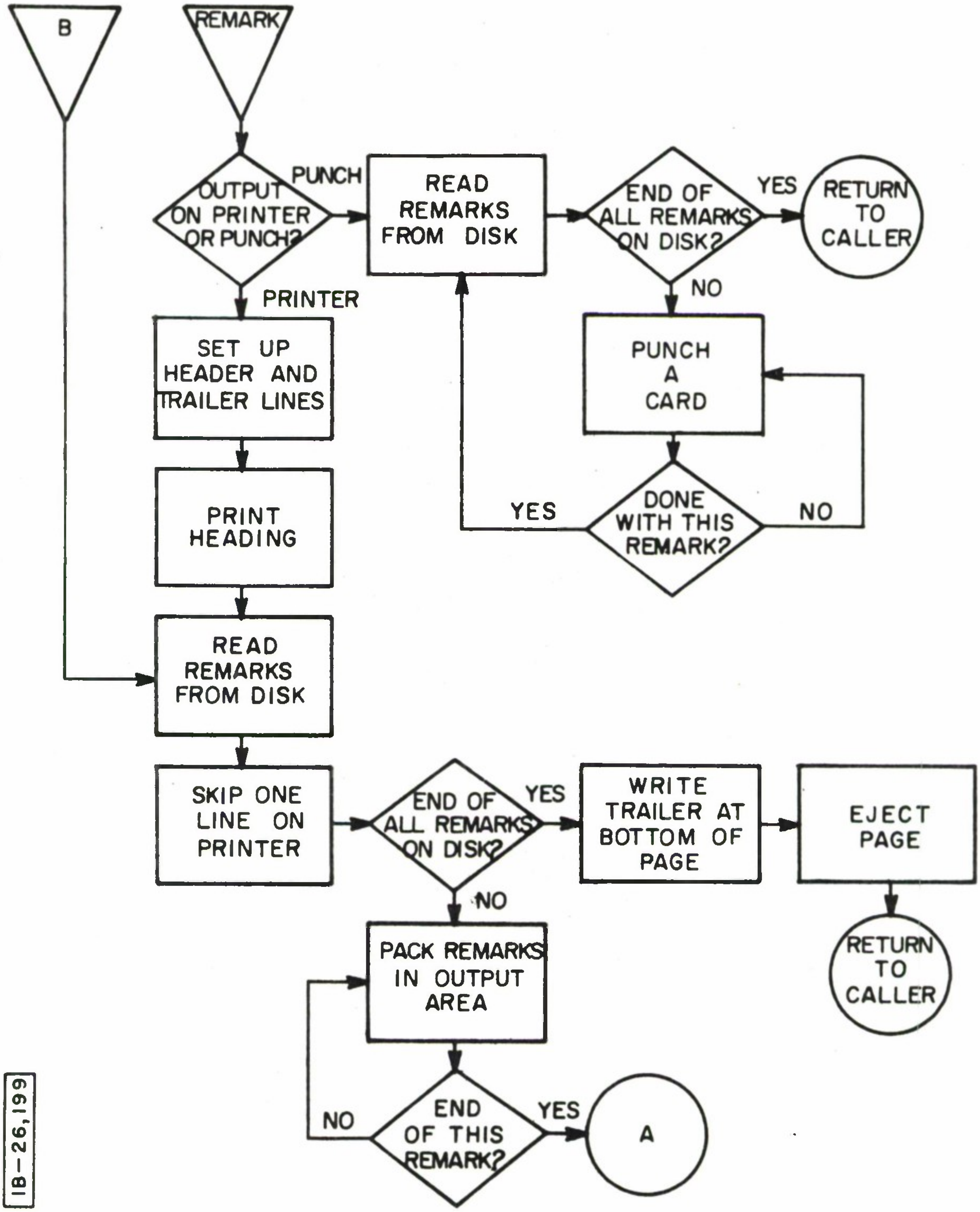
Description

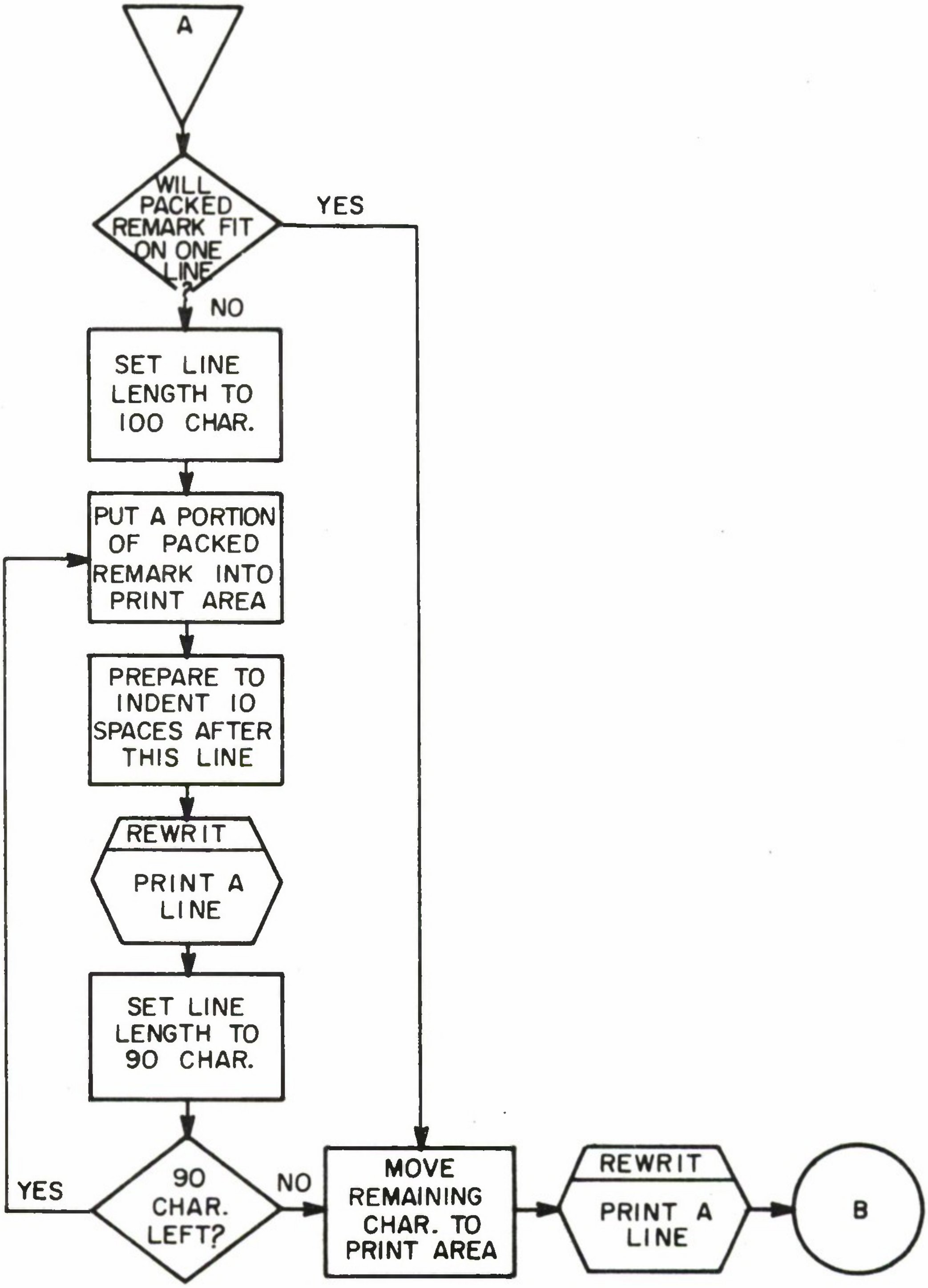
During the input or file generation phase of REPGEN, any input records defined as remarks are put into a remarks area on the system disk. Remarks normally would be punched into cards, and a given remark may not exceed five cards in length.

If the output medium is punched cards, the REMARK processor simply reads back the remarks stored on disk and punches them into cards.

If the output medium is the printer, REMARK prints a heading and reads a remark from the disk. The remark is packed into a print line and printed. If it overflows to more than one line, subsequent lines are indented ten spaces until the entire remark is printed. The printer is spaced once, the next remark is read and the process is repeated until all remarks have been printed.

The REMARK routine is called and executed after REALO has finished executing an output program.


B
REMARK
OUTPUT ON PRINTER OR PUNCH?
PUNCH
PRINTER
READ REMARKS FROM DISK
END OF ALL REMARKS ON DISK?
YES
RETURN TO CALLER
NO
PUNCH A CARD
DONE WITH THIS REMARK?
YES
NO
SET UP HEADER AND TRAILER LINES
PRINT HEADING
READ REMARKS FROM DISK
SKIP ONE LINE ON PRINTER
END OF ALL REMARKS ON DISK?
YES
WRITE TRAILER AT BOTTOM OF PAGE
EJECT PAGE
RETURN TO CALLER
NO
PACK REMARKS IN OUTPUT AREA
END OF THIS REMARK?
NO
YES
A
IB-26,199


A
WILL PACKED REMARK FIT ON ONE LINE ?
YES
NO
SET LINE LENGTH TO 100 CHAR.
PUT A PORTION OF PACKED REMARK INTO PRINT AREA
PREPARE TO INDENT 10 SPACES AFTER THIS LINE
REWRIT
PRINT A LINE
SET LINE LENGTH TO 90 CHAR.
YES
90 CHAR. LEFT?
NO
MOVE REMAINING CHAR. TO PRINT AREA
REWRIT
PRINT A LINE
B

IB-26,200

REPEET

Purpose

A stored query GETTAC statement is retrieved, expanded and executed.

Calling Sequence

REPEET is called internally by REPGEN. The call is placed in SDA ØØØ3 as

```
REPEETbNNN
```

where NNN is the saved statement number assigned by SAVER. REPGEN then issues a

```
B    COPS
DCW  'REPEET'
```

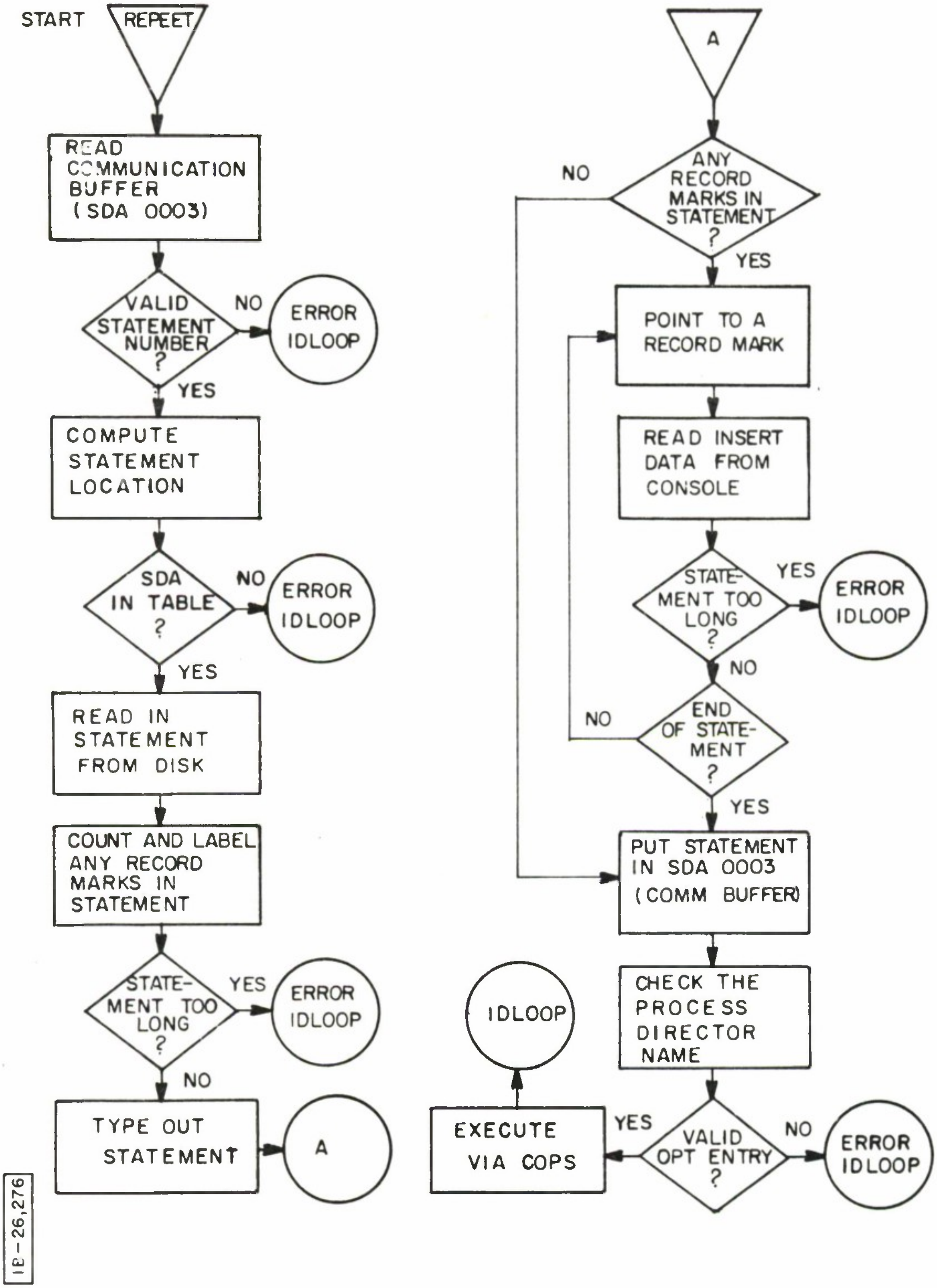
REPEET may be called externally by typing

```
REPEETbNNN
```

on the console typewriter.

Description

The statement number is validity checked. The saved statement SDA table is read in from disk and the SDA corresponding to the given statement number is extracted. The saved statement is read into core from disk. The statement is scanned for record marks. Each record mark is labeled by a record mark number. The statement is then typed out and data insertions are made via typewriter for each record mark in the statement. The edited statement is written to the system communication area and the GETTAC processor is called via COPS.


START
REPEET
READ COMMUNICATION BUFFER (SDA 0003)
VALID STATEMENT NUMBER ?
NO
ERROR IDLOOP
YES
COMPUTE STATEMENT LOCATION
SDA IN TABLE ?
NO
ERROR IDLOOP
YES
READ IN STATEMENT FROM DISK
COUNT AND LABEL ANY RECORD MARKS IN STATEMENT
STATE-MENT TOO LONG ?
YES
ERROR IDLOOP
NO
TYPE OUT STATEMENT
A
A
ANY RECORD MARKS IN STATEMENT ?
NO
YES
POINT TO A RECORD MARK
READ INSERT DATA FROM CONSOLE
STATE-MENT TOO LONG ?
YES
ERROR IDLOOP
NO
END OF STATE-MENT ?
NO
YES
PUT STATEMENT IN SDA 0003 (COMM BUFFER)
CHECK THE PROCESS DIRECTOR NAME
VALID OPT ENTRY ?
NO
ERROR IDLOOP
YES
EXECUTE VIA COPS
IDLOOP
IB-26,276

REKALL

Purpose

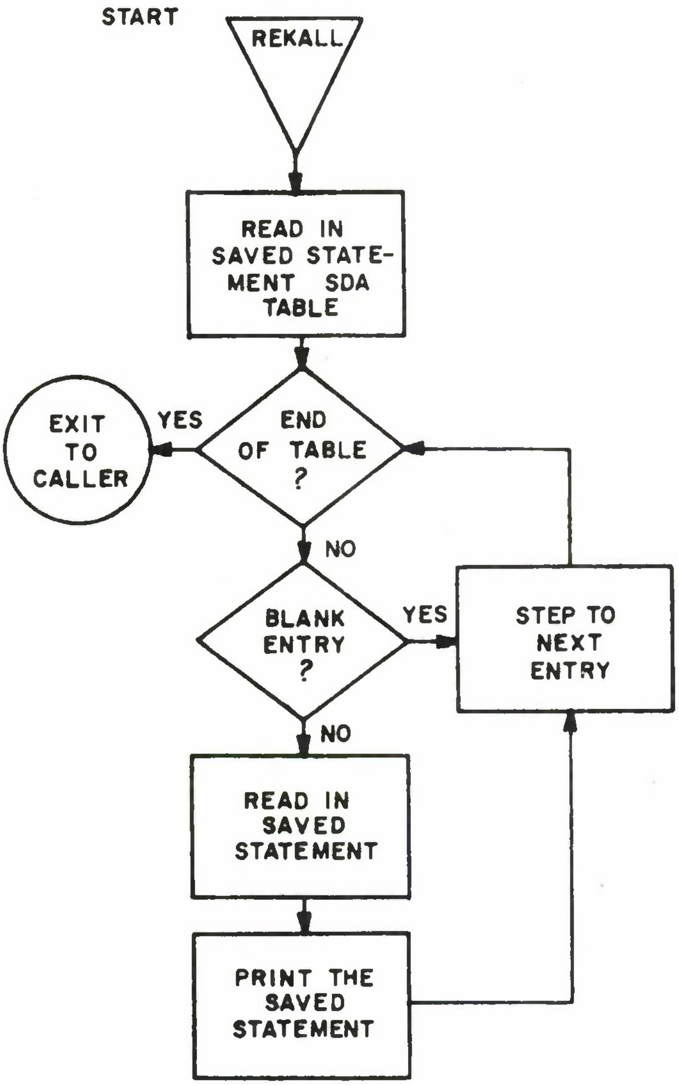
Print out all statments previously stored on disk by the program SAVER.

Calling Sequence

Enter REKALL on the console typewriter. Return will be to IDLOOP.

Description

REKALL reads the saved statement SDA table into core. Each non-blank entry points to a saved statement. Each saved statement is read in turn from the disk and is printed along with its assigned statement number. A total of 5ØØ statements may be saved.


START
REKALL
READ IN SAVED STATEMENT SDA TABLE
END OF TABLE ?
YES
EXIT TO CALLER
NO
BLANK ENTRY ?
YES
STEP TO NEXT ENTRY
NO
READ IN SAVED STATEMENT
PRINT THE SAVED STATEMENT

1A-26,233

DELEET

Purpose

Delete from the system disk a Query Language statement previously stored there by the SAVER program.

Calling Sequence

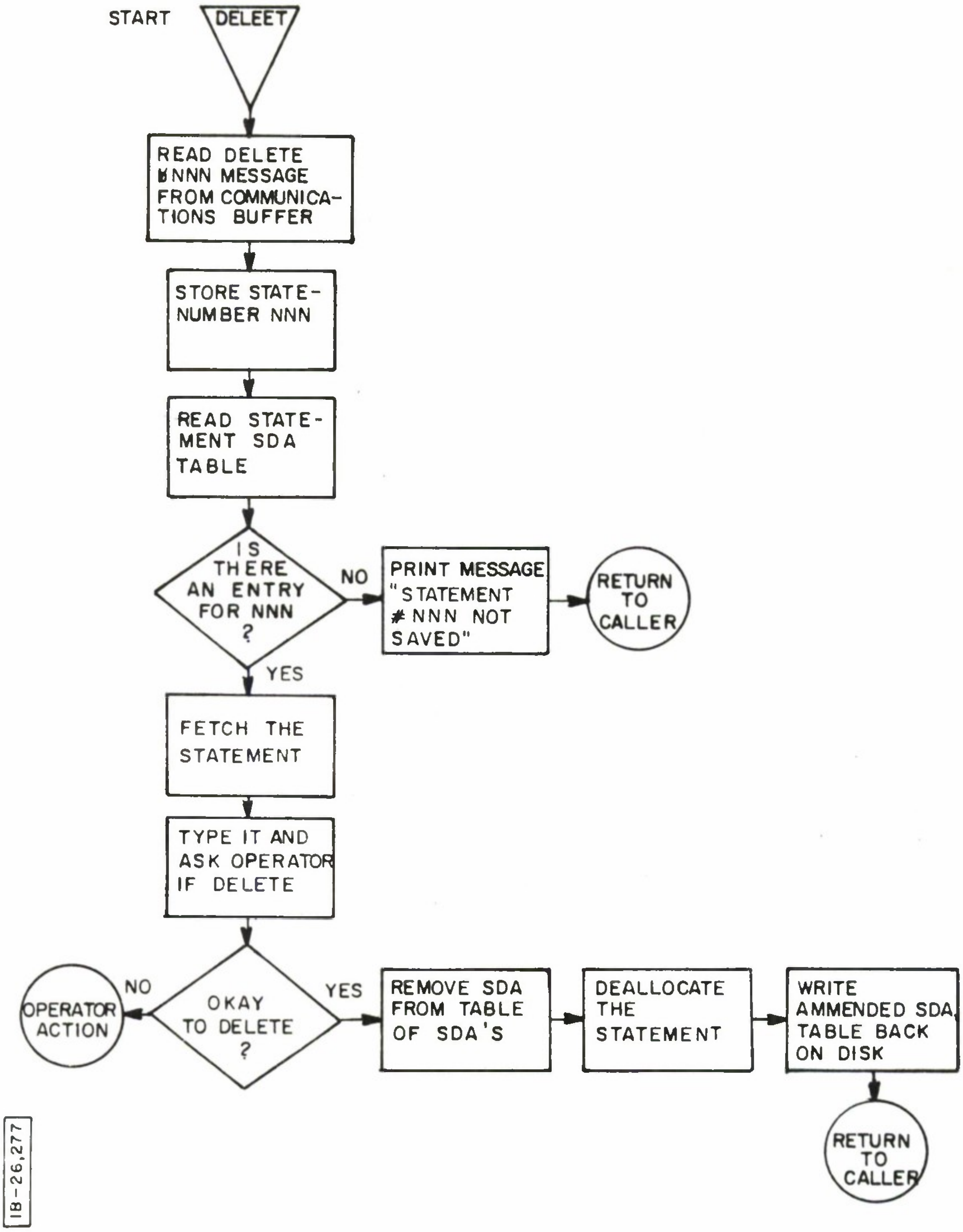
Enter on the console typewriter

DELEETᵬNNN

where NNN is the saved statement number assigned by SAVER.

Description

The DELEETᵬNNN message is retrieved from SDA 0003. The 3 character number referencing the statement to be deleted is stored. The table of 4 character SDA's which references where each saved statement is stored is read into core. The first entry in the table references statement number 001, the second entry references statement number 002, and so on. Blank entries indicate that statement number has not been saved and is available. The statement SDA to be deleted is then located. If it is blank, a message will be printed on the console indicating that this statement cannot be deleted because it has not been saved. If an SDA is found in the table entry corresponding to the statement number to be deleted, the saved statement is typed out on the I/O typewriter for the operator's approval. If okay, the SDA is deleted from the statement table and also deallocated so that the statement itself is effectively deleted. The new SDA statement table is now rewritten on the disk and the program returns to IDLOOP.


START
DELEET
READ DELETE ЖNNN MESSAGE FROM COMMUNICATIONS BUFFER
STORE STATE-NUMBER NNN
READ STATEMENT SDA TABLE
IS THERE AN ENTRY FOR NNN ?
NO
PRINT MESSAGE "STATEMENT #NNN NOT SAVED"
RETURN TO CALLER
YES
FETCH THE STATEMENT
TYPE IT AND ASK OPERATOR IF DELETE
OKAY TO DELETE ?
NO
OPERATOR ACTION
YES
REMOVE SDA FROM TABLE OF SDA'S
DEALLOCATE THE STATEMENT
WRITE AMMENDED SDA TABLE BACK ON DISK
RETURN TO CALLER
1B-26,277

PRTCRD

Purpose

PRTCRD is used to print a report from cards which were generated by RPTCRD.

Calling Sequence

Enter PRTCRD on the console typewriter. Return will be to IDLOOP.

Description

The print area is cleared, the printer is spaced to a new page, and a card is read.

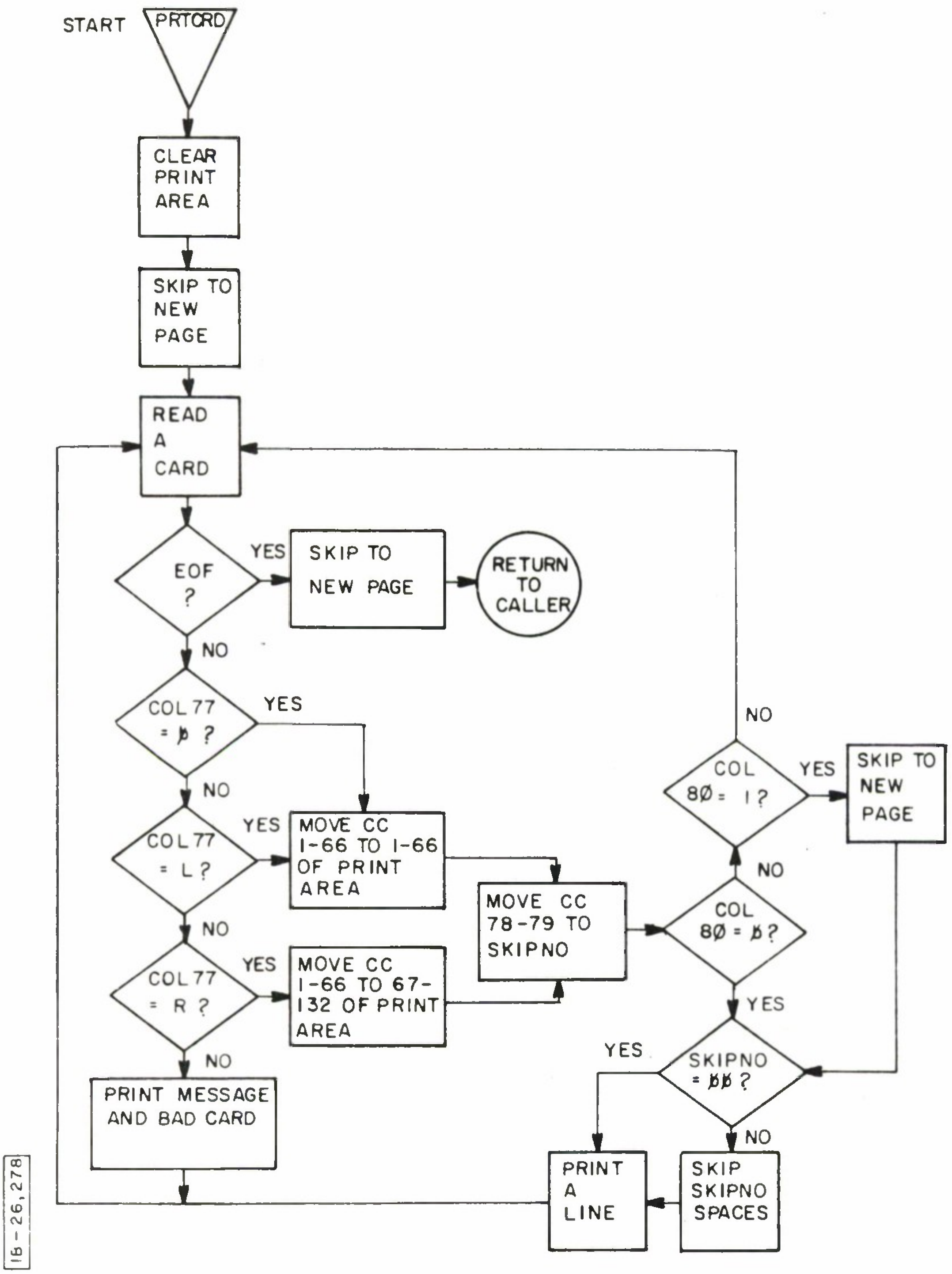
Each input card contains half of a print image in columns 1-66. Columns 77-80 contain information identifying which half of the print image the card represents and any page or line spacing to be done before printing. Columns 77-80 are interpreted as follows:

Column 77 = L or b̸ ; This is left half of print image.
R ; This is right half of print image.

Columns 78-79 = nn ; Number of times to space before printing.
Blank specifies single spacing.

Column 80 = b̸ ; Print after putting this card in buffer and performing required spacing.
1 ; Same as b̸, except skip to new page first.
* ; Put this card in buffer but do not print or space.

As each card is read, columns 1-66 are placed into the left or right half of a print image. If column 80 is a blank or 1, required spacing is performed and a line is printed. If column 80 is *, no printing or spacing is done, and the next card is immediately read.


START
PRTCRD
CLEAR PRINT AREA
SKIP TO NEW PAGE
READ A CARD
EOF ?
YES
SKIP TO NEW PAGE
RETURN TO CALLER
NO
COL 77 = ƀ ?
YES
NO
COL 77 = L ?
YES
MOVE CC 1-66 TO 1-66 OF PRINT AREA
NO
COL 77 = R ?
YES
MOVE CC 1-66 TO 67-132 OF PRINT AREA
NO
PRINT MESSAGE AND BAD CARD
MOVE CC 78-79 TO SKIPNO
COL 8Ø = ƀ ?
NO
COL 8Ø = 1 ?
YES
SKIP TO NEW PAGE
NO
YES
SKIPNO = ƀƀ ?
YES
NO
PRINT A LINE
SKIP SKIPNO SPACES
18-26,278

PRTTAP

Purpose

PRTTAP is used to print additional copies of a report from a REPGEN extra-copies tape.

Calling Sequence

PRTTAP is called internally from RGPOUT. RGPOUT puts

PRTTAPn

where n is the number of copies desired, into SDA 0003 and then calls

```
B     COPS
DCW   'PRTTAP'
```

The program can also be called externally by entering

PRTTAPn

on the console typewriter. Return will be to IDLOOP.

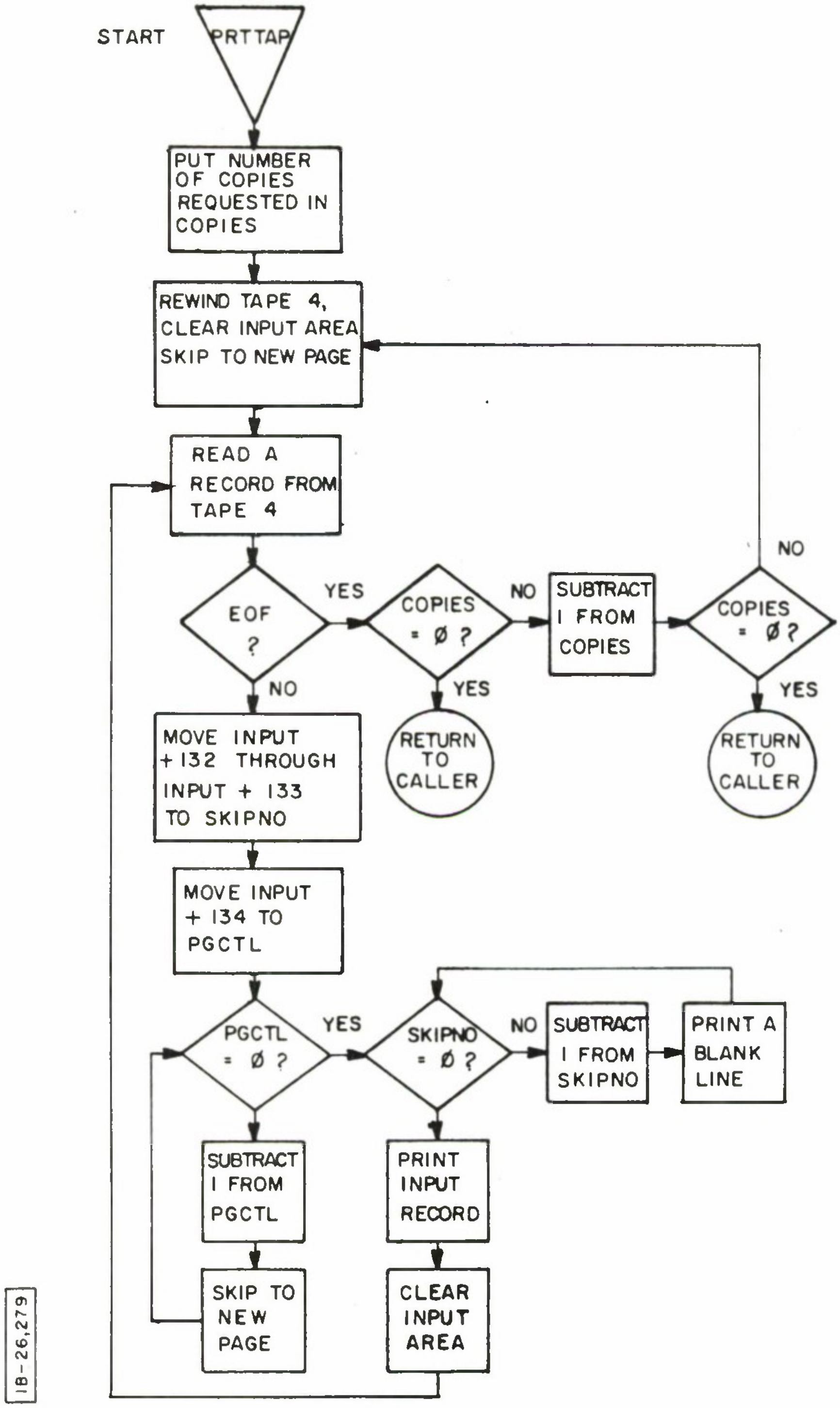
Description

The input to PRTTAP is tape 4, with one file of 135 character records written in even parity. The first 132 characters of each record represent a print image. The last three characters of each record contain information regarding page and line spacing to be accomplished before printing the image.

The format of each record is as follows:

```
Character    1-132 = Print image.
           133-134 = Number of times to space before
                     printing.
                     Blank specifies single spacing
               135 = Page control.
                     1 means skip to new page before
                     printing; blank means do not skip.
```

PRTTAP reads a record and first examines the page control character. If it is a 1, a skip to channel 1 is executed. Next, if any spacing is indicated it is accomplished by printing the appropriate number of blank lines. The line is then printed. When an end-of-file is encountered, the tape is rewound, and the number of copies is decremented by one. If it is not blank, another run is made through the tape.



START
PRTTAP
PUT NUMBER OF COPIES REQUESTED IN COPIES
REWIND TAPE 4, CLEAR INPUT AREA SKIP TO NEW PAGE
READ A RECORD FROM TAPE 4
EOF ?
YES
NO
COPIES = Ø ?
NO
YES
RETURN TO CALLER
SUBTRACT 1 FROM COPIES
COPIES = Ø ?
NO
YES
RETURN TO CALLER
MOVE INPUT +132 THROUGH INPUT + 133 TO SKIPNO
MOVE INPUT + 134 TO PGCTL
PGCTL = Ø ?
YES
SKIPNO = Ø ?
NO
SUBTRACT 1 FROM SKIPNO
PRINT A BLANK LINE
SUBTRACT 1 FROM PGCTL
SKIP TO NEW PAGE
PRINT INPUT RECORD
CLEAR INPUT AREA
1B-26,279

RPTCRD

Purpose

RPTCRD is used to punch cards from a REPGEN extra-copies tape in a format which will allow the cards to be processed by PRTCRD.

Calling Sequence

Enter RPTCRD on the console typewriter. Return will be to IDLOOP.

Description

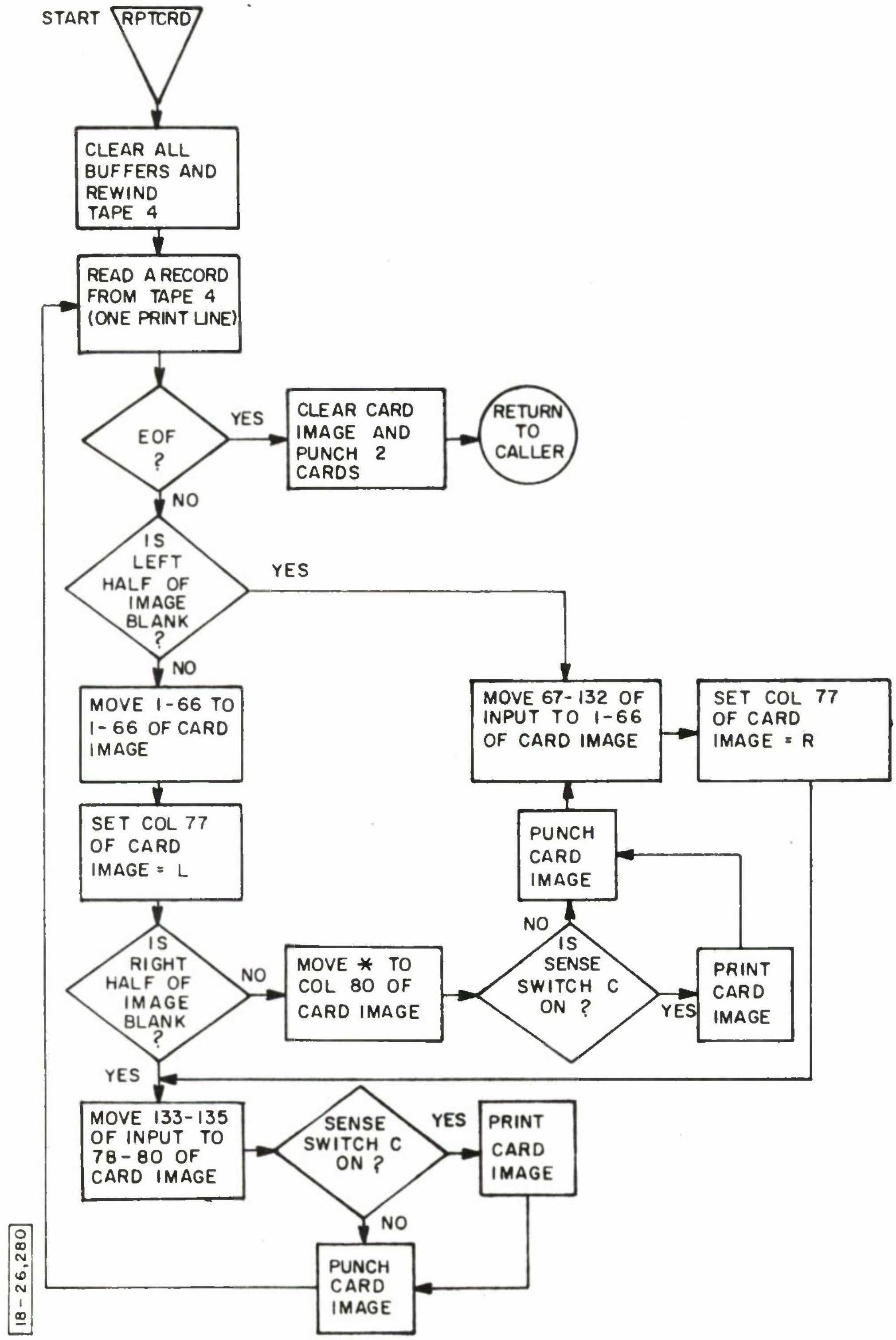
Input to RPTCRD is tape 4, with one file of 135 character records written in even parity.

Each input record from tape 4 is treated as a print line, where the first 132 characters constitute a print image, and the last three characters are for carriage control. (See write-up of PRTTAP for discussion of the carriage control characters.)

RPTCRD will divide each print image into a left and right half, and punch each half into the first 66 columns of a card using columns 77-80 of the cards for carriage control information. These cards are in proper format to be processed by PRTCRD, which will use these cards to generate a report on the printer. If Sense Switch C is on, each card will be printed as well as punched.

The format of the output cards and the meaning of the data entered in columns 77-80 is the same as that which was presented in section 6.18.3 to describe the input cards to PRTCRD.

When an end-of-file is encountered on tape 4, two blank cards are punched and control returns to IDLOOP.


START
RPTCRD
CLEAR ALL BUFFERS AND REWIND TAPE 4
READ A RECORD FROM TAPE 4 (ONE PRINT LINE)
EOF ?
YES
CLEAR CARD IMAGE AND PUNCH 2 CARDS
RETURN TO CALLER
NO
IS LEFT HALF OF IMAGE BLANK ?
YES
NO
MOVE 1-66 TO 1-66 OF CARD IMAGE
MOVE 67-132 OF INPUT TO 1-66 OF CARD IMAGE
SET COL 77 OF CARD IMAGE = R
SET COL 77 OF CARD IMAGE = L
PUNCH CARD IMAGE
NO
IS RIGHT HALF OF IMAGE BLANK ?
NO
MOVE * TO COL 80 OF CARD IMAGE
IS SENSE SWITCH C ON ?
YES
PRINT CARD IMAGE
YES
MOVE 133-135 OF INPUT TO 78-80 OF CARD IMAGE
SENSE SWITCH C ON ?
YES
PRINT CARD IMAGE
NO
PUNCH CARD IMAGE
18-26,280

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* The MITRE Corporation Bedford, Massachusetts	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP N/A

3. REPORT TITLE

PROGRAM/CODING SPECIFICATION FOR THE AFICCS REPORT GENERATION (REPGEN) CAPABILITY

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

N/A

5. AUTHOR(S) *(First name, middle initial, last name)*

Norman B. Sutherland

6. REPORT DATE January 1969	7a. TOTAL NO. OF PAGES 170	7b. NO. OF REFS 0
8a. CONTRACT OR GRANT NO. AF 19(628)-5165 b. PROJECT NO. 512V c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) ESD-TR-68-134 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* MTR-564	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES N/A	12. SPONSORING MILITARY ACTIVITY Director of Planning and Technology, Electronic Systems Division, Air Force Systems Command, L. G. Hanscom Field, Bedford, Mass.

13. ABSTRACT

The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 141Ø computer under the Air Force Integrated Command and Control System (AFICCS). The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, magnetic tape or on the system disk. This document contains detailed information regarding the structure and operation of the REPGEN programs, including table descriptions, core maps, program flow-charts and abstracts.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
REPORT GENERATOR						
AFICCS REPORT GENERATION CAPABILITY						
REPORT GENERATOR PROGRAM SPECIFICATIONS - (1410)						