TECHNICAL DOCUMENTATION

for

UNICODE

Automatic Programming System for Univac Scientific 1103A and 1105

Volume I

April , 1961

PX 1790

Remington Rand Univac®
DIVISION OF SPERRY RAND CORPORATION
UNIVAC PARK, ST. PAUL 16, MINNESOTA

VOLUME I

VOLUME II

VOLUME III

VOLUME I

TABLE OF CONTENTS

3. UNICODE SYSTEM TAPE PACKAGE

4. UNICODE SAMPLE CODING

5. UNICODE CARD INPUT

6. STATISTICAL MISCELLANY

I. INTRODUCTION

I. INTRODUCTION

This collection of notes, flow charts and annotated coding has been gathered together to form a documentation of UNICODE, the automatic programming system for Univac Scientific computers 1103A and 1105. The material supplements that contained in the UNICODE Manual (U-1451, Rev. 3). It is not needed to operate the UNICODE System but it would be useful to anyone desiring to modify the System.

Following the flow charts and annotated coding of UNICODE Service Routines which are given in the General section, the coding of the Librarian is given in Use Compiler form. The use of the Librarian to build up an Installation Library is explained in Chapter 11 of the UNICODE Manual. More complete coverage is given here to the Permanent Library Routines described in the same chapter.

The notes on the System Tape Package explain the use of this bioctal tape in updating or changing the UNICODE Master Tape. Flow charts and annotated coding of Routines 1-5 are also presented.

UNICODE coding samples given in Section II are a matrix inversion program, a system of converting floating-point numbers to fixed-point, and a linear programming application.

Use of UNICODE with card input is also discussed in the General section. Flow charts and coding are given for a routine which will convert a card-to-tape converted tape to a UNICODE input tape. The latter routine may readily be added to the UNICODE System.

A page of corrections to the UNICODE Manual is included in a miscellaneous statistical section. How the compiler can be altered to produce two- and three-core Object Programs is also explained at the end of the General section.

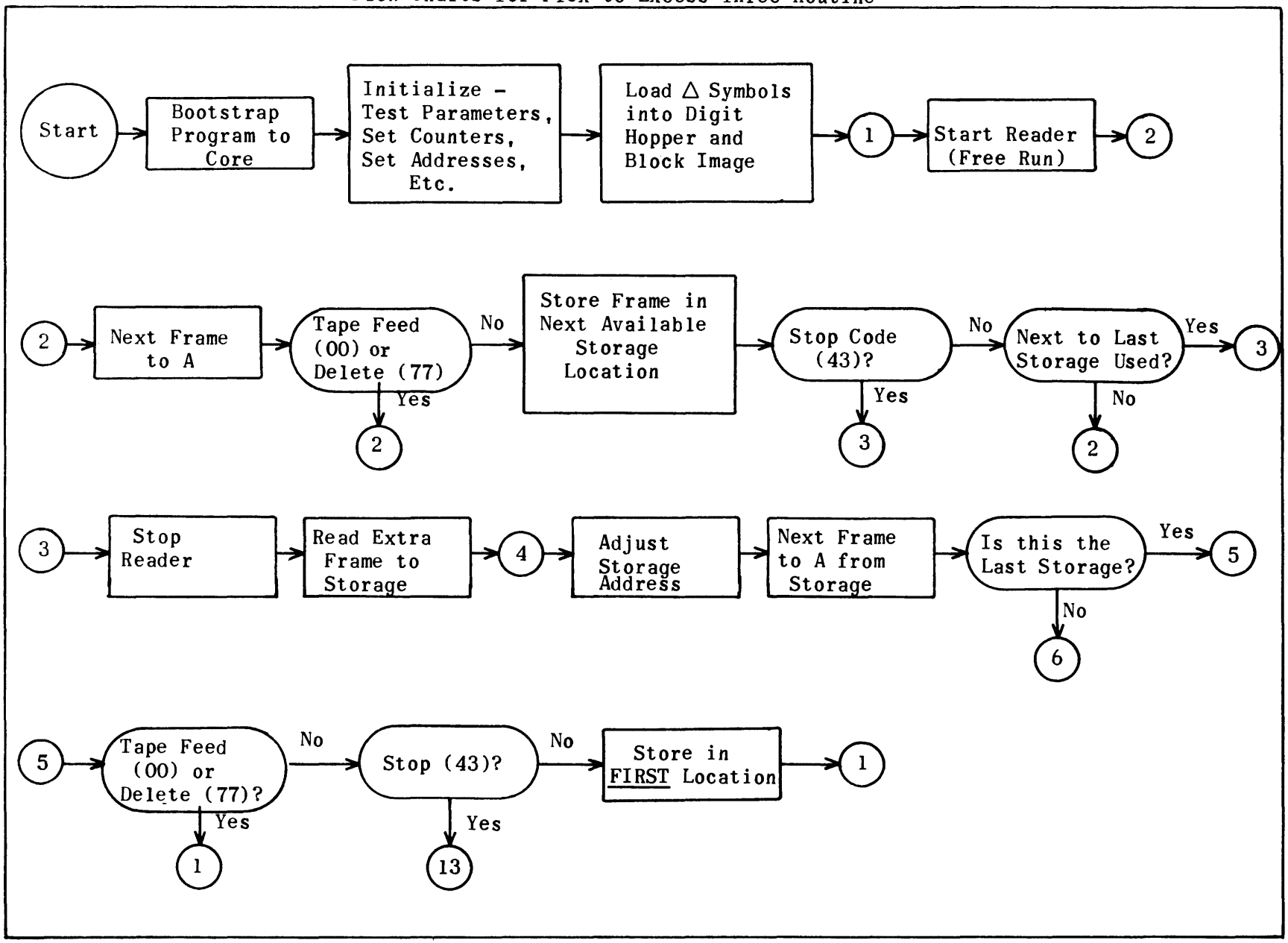
The remainder of this manual contains notes and flow charts, when available, and the annotated coding of the routines used for compilation from the beginning processes of correction and translation to the final listing, by sentence, of the Object Program produced.

II. GENERAL

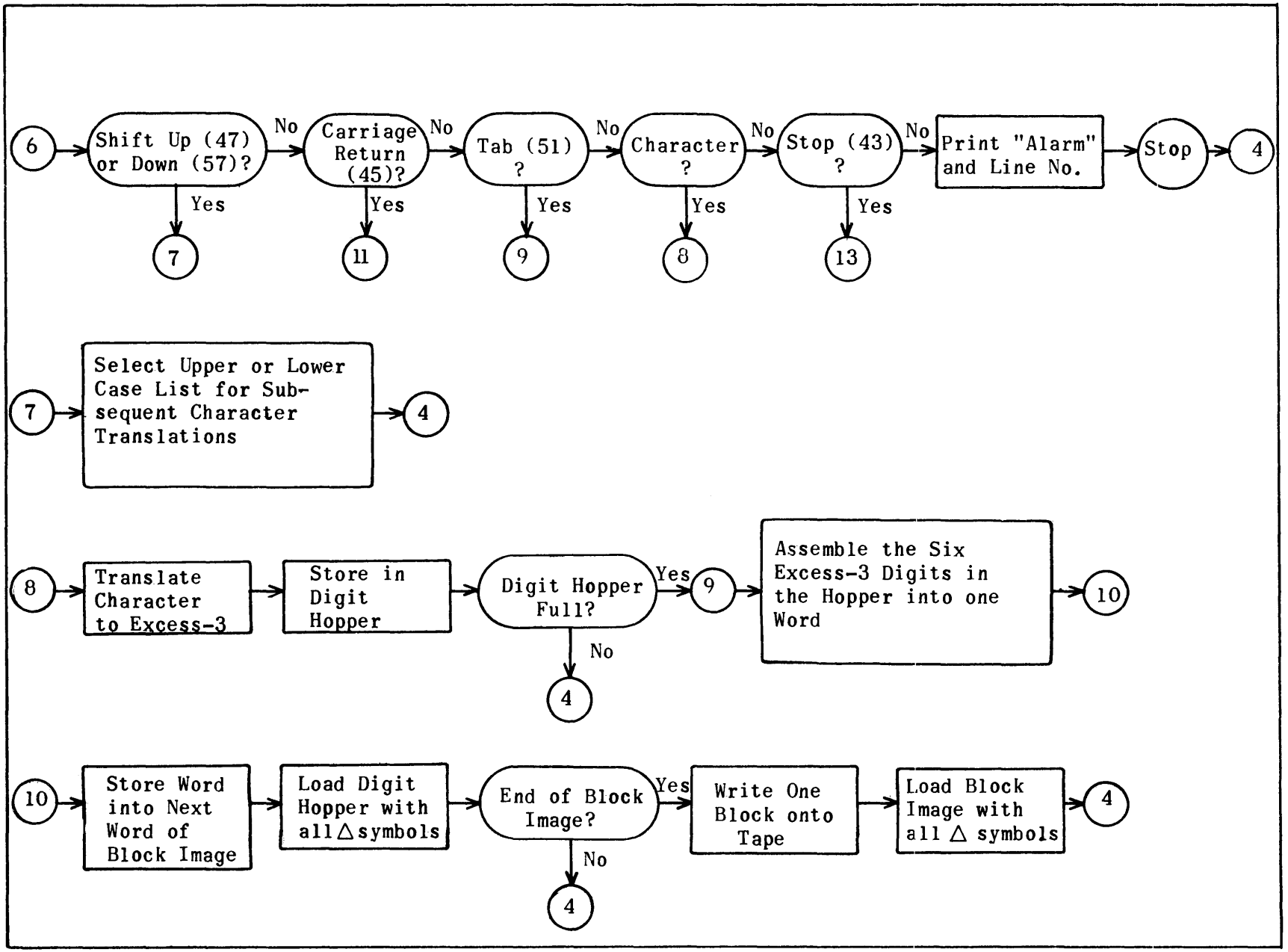
1. UNICODE SERVICE ROUTINES

A. Flex to Excess-Three Routine

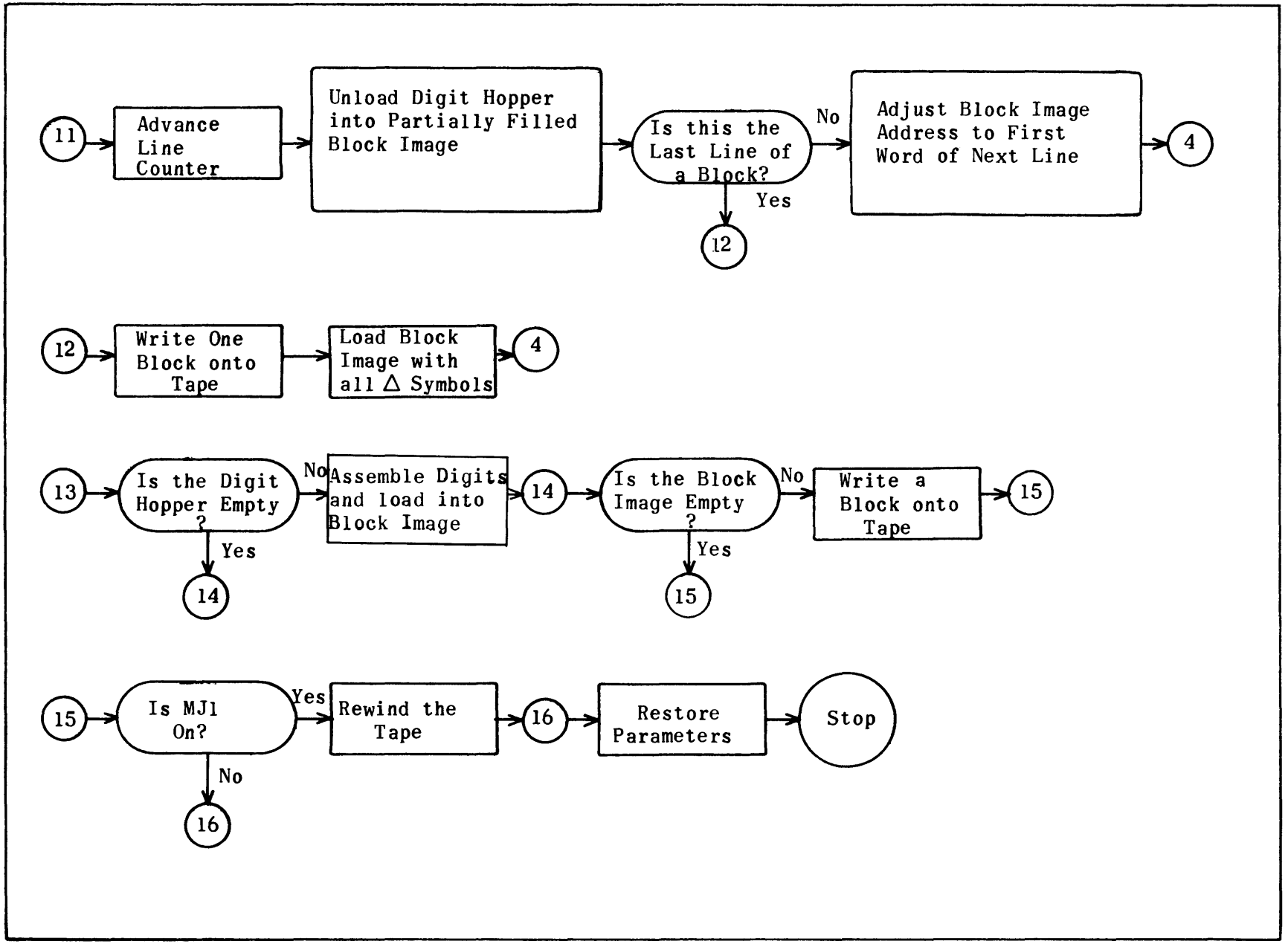
Flow Charts for Flex to Excess-Three Routine



Flow Charts for Flex to Excess-Three Routine (Cont.)



Flow Charts for Flex to Excess-Three Routine (Cont.)



Flex to Excess-Three Routine

Regions

```
RE  BS76300
RE  FC76303
RE  IN1
RE  RD42
RE  CD56
RE  ER102
RE  CS132
RE  WD141
RE  WB156
RE  LC165
RE  UC172
RE  CR177
RE  EX206
RE  MS223
RE  FA255
RE  EA301
RE  FB325
RE  EB400
RE  WS453
RE  SD464
RE  BI472
RE  SF662
```

Flex to Excess-Three Routine (1105 Version)

```
          IA   BS
       0  TP   FC14   0
       1  RP   30454  IN      Bootstrap routine to core
       2  TP   FC     IN
          CA   BS3

          IA   FC
IN     0  LT   0      WS2     Store A
       1  TP   A      WS3
       2  TP   Q      WS4     Store Q
       3  SP   WS3    0       (A_R)⟶A
       4  ZJ   IN5    IN6     If = 0⟶alarm
       5  TJ   EA6*   IN15    If not too large⟶IN15
       6  PR   0      FB32    "P"
       7  SP   WS2    44
      10  SA   WS3    0       Restore A & Q
      11  TP   WS4    Q
      12  MS   0      IN13    Stop
      13  MJ   0      IN      Recycle
      14  MJ   0      IN      Not used
      15  SP   A      14      Position Uniservo No. in A
      16  TP   MS5    Q       Mask⟶Q
      17  QS   A      MS3     Uniservo No.⟶ef words for writing
      20  QS   A      MS4       and rewinding
      21  LQ   MS6    Q3      TCU mask⟶Q
      22  SP   WS4    36      TCU No.⟶A
      23  TJ   MS7    IN6
      24  TJ   Q      IN26    If illegal⟶IN6
      25  MJ   0      IN6     If OK⟶IN26
      26  QS   A      MS3     TCU No.⟶ef word for writing, and re-
      27  QS   A      MS4       winding
      30  TP   MS6    Q       Buffer No. mask⟶Q
      31  SP   WS4    33      Buffer No.⟶A
      32  QS   A      MS2     Buffer No. ef word to "write buffer"
      33  SP   WS4    32      Buffer No.⟶A to mod MJ4 command
      34  QS   A      WB      Modify MJ4 command
      35  RP   10006  1N37    Load the digit hopper with
      36  TP   EA23   SD        all (0 0 1)'s
      37  RP   10170  RD      Load the block image with
      40  TP   MS10   BI        all Δ 's
RD     0  EF   0      MS      Start reader
       1  ER   0      A       Frame⟶A
       2  RJ   RD2    RD1     Discard 1st frame
       3  EJ   EA10   RD1     Ignores (00 & 77)
       4  EJ   MS13   RD1
       5  TP   A      SF      Store frame in image
       6  EJ   EA     RD11    If stop,⟶RD11
       7  RA   RD5    EA23    Advance storage address of frame
```

```
        10   TJ   MS14    RD1          If not next to last, recycle
        11   EF   0       MS1          Stop reader
        12   ER   0       7777         Last frame ⟶7777
        13   TV   CD7     RD5          Start of image ⟶RD5
CD       0   TU   MS20    CD12         Starting address⟶CD12
         1   RA   CD12    MS11         Advance by one
         2   TJ   CD3     CD12         Test &⟶ CD12
         3   SP   7777    0            Last frame ⟶A
         4   EJ   EA10    RD  }
         5   EJ   MS13    RD  }        Ignores
         6   EJ   EA      EX           Stop
         7   TP   A       SF           Store in lst location
        10   RA   RD5     EA23         Advance address
        11   MJ   0       RD           Return for another block
        12   SP   30000   0            Frame⟶ A
        13   EJ   EA11    LC           If shift down⟶LC
        14   EJ   EB23    UC           If shift up ⟶UC
        15   MJ   0       ER           To ER if not a shift on lst frame
        16   RJ   CD15    CD1          Clear the v address of CD14
        17   EJ   EB25    CR           If carriage return⟶ CR
        20   EJ   EB21    WD           If tab⟶WD
        21   RP   40000   CD23 }       If a digit, character, or symbol,
        22   EJ   30000   CS1  }         ⟶CS1
        23   EJ   EA      EX           If a stop,⟶EX; if illegal,⟶ER
ER       0   PR   0       EA11         "lc"
         1   PR   0       EB25         "cr"
         2   SP   MS17    52  }
         3   PR   0       A   }        "Error sp"
         4   SS   A       6   }
         5   ZJ   ER3     ER6 }
         6   TP   WS      Q   }
         7   TP   EB5     WS5 }
        10   TV   MS31    ER13}
        11   TP   Q       A   }        Last 3 decimal digits of line
        12   DV   EB42    Q   }        No.⟶storage (line No. = 1 + No.
        13   TP   A       30000}       of cr executed so far)
        14   RS   ER13    EA23}
        15   IJ   WS5     ER11}
        16   TP   EB5     WS5 }
        17   TU   MS31    ER20}
        20   SP   30000   0   }        Print last three decimal digits
        21   AT   MS12    ER22}        of line No.
        22   0    0       0   }
        23   RA   ER20    MS11}
        24   IJ   WS5     ER20}
        25   MS   0       ER26         Stop
        26   MJ   30000   EX6          If MJ3 is on⟶exit
        27   MJ   0       CD1          If not, ignore the error
CS       0   TP   30000   SD           Dummy
         1   SP   Q       17           j, n-r ⟶(A_R)_u
         2   ST   30000   A            Subtract j,n, leaving -r
         3   AT   CS      CS4          Add to dummy instr. & store
```

```
         4   0   0       0          Execute the storage of excess-three
                                      symbol
         5   RA  CS      EA23       Advance storage address
         6   IJ  WS1     CD1        Recycle until digit hopper is full;
                                      when full ⟶ WD
WD       0   SP  SD      6   ┐
         1   RP  20004   WD3 ├      Form word in A_R
         2   SA  SD1     6   ├
         3   SA  SD5     0   ┘
         4   TP  A       BI         Store in block image
         5   RP  10006   WD7 ┐      Δ ⟶ digit hopper
         6   TP  EA23    SD  ┘
         7   TV  WD6     CS         Start at top of digit hopper (in CS)
        10   TP  FB22    WS1        Reset index counter
        11   RA  WD4     EA23       Advance block image add
        12   RJ  WD12    WD13       On shut
        13   TJ  MS15    CD1        If not at end of block, recycle
        14   TV  WB6     WD4        Restore v of WD4 to start a new block
                                      WRITING A BLOCK
WB       0   MJ  40000   WB         Test buffer activity
         1   EF  0       MS2        ef for "write buffer"
         2   RP  10170   WB4 ┐      Load buffer from image
         3   EW  10000   BI  ┘
         4   EF  0       MS3        ef for "write one block"
         5   RP  10170   CD1 ┐      Load image with all Δ's and exit to CD1
         6   TP  MS10    BI  ┘
LC       0   TU  MS21    CD21┐
         1   TU  MS22    CD22├      Arrange to take lower case lists
         2   TU  MS23    CS  ├        (FB and EB)
         3   TU  MS24    CS2 ┘
         4   MJ  0       CD16
UC       0   TU  MS25    CD21┐
         1   TU  MS26    CD22├      Arrange to select upper case lists
         2   TU  MS27    CS  ├        (FA and EA)
         3   TU  MS30    CS2 ┘
         4   MJ  0       CD16
CR       0   RA  WS      EA23       Advance counter
         1   RJ  WD12    WD         Store one word and advance storage
         2   TP  MS16    A          Initial word ⟶ A    TP  A  BI
         3   AT  FB35    A   ┐      Add No. of words in a line until
         4   TJ  WD4     CR3 ┘        word 4 is no longer >
         5   TP  A       WD4        Store in WD4
         6   MJ  0       WD13       Check for end of block
EX       0   SP  WS1     0          Index for hopper
         1   EJ  FB22    EX3        If empty ⟶ EX3
         2   RJ  WD12    WD         If not, dump into block image
         3   SP  WD4     0          Command to store next word into BI
         4   EJ  MS16    EX6        If block image is empty ⟶ EX6
         5   RJ  WB5     WB         If not, dump the block
         6   MJ  10000   EX10┐      If MJ1 is on, rewind without
         7   MJ  0       EX11├        interlock
        10   EF  0       MS4 ┘
```

```
        11   TP   WS4     Q       )
        12   SP   WS2     44      } Restore Q and A
        13   SA   WS3     0       )
        14   MS   0       BS      Stop on entry
MS       0   10   2       0       (EF) Start reader
         1   10   1       0       (EF) Stop reader
         2   0    0       400     (EF) Write buffer
         3   0    646     0       (EF) Write one block
         4   0    200     0       (EF) Rewind w/o interlock
         5   0    1       70000   Uniservo No. mask
         6   0    30000   0       TCU and Buffer mask
         7   1    0       0       Min TCU No. (for tj)
        10   1    1010    10101   All Δ
        11   0    1       0       u advance
        12   PR   0       FA1     Print 0 (start of 10 dec. digits)
        13   0    0       77      Ignore
        14   TP   A       7777    For tj
        15   TP   A       BI170   For tj
        16   TP   A       BI      For the CR routine & the EX routine
        17   20   12120   31204   "error Δ "
        20   0    BI167   0       sf (-1) = BI167
        21   0    20053   0       )
        22   0    FB      0       } For LC routine
        23   0    EB53    0       )
        24   0    MS21    0       )
        25   0    20024   0       )
        26   0    FA      0       } For UC routine
        27   0    EA24    0       )
        30   0    MS25    0       )
        31   0    WS6     WS10    For ER routine
FA       0   0    0       4       space          UPPER CASE LIST
         1   0    0       37      0
         2   0    0       52      1
         3   0    0       74      2
         4   0    0       70      3
         5   0    0       64      4
         6   0    0       62      5
         7   0    0       66      6
        10   0    0       72      7
        11   0    0       60      8
        12   0    0       33      9
        13   0    0       56      -
        14   0    0       44      .
        15   0    0       22      D (')
        16   0    0       54      /
        17   0    0       27      X (*)
        20   0    0       13      G ( > )
        21   0    0       11      L ( < )
        22   0    0       46      (
        23   0    0       42      )
EA       0   0    0       43      )
         1   0    0       17      (
```

	2	0	0	37	<	
	3	0	0	16	>	
	4	0	0	56	*	
	5	0	0	64	/	
	6	0	0	15	/	
	7	0	0	62	.	
	10	0	0	0	-	
	11	0	0	57	9	
	12	0	0	36	8	
	13	0	0	75	7	Superscripts
	14	0	0	55	6	
	15	0	0	35	5	
	16	0	0	41	4	
	17	0	0	20	3	
	20	0	0	40	2	
	21	0	0	61	1	
	22	0	0	60	0	
	23	0	0	01	Δ	
FB	0	0	0	04	Δ	Lower case list
	1	0	0	37	0	
	2	0	0	52	1	
	3	0	0	74	2	
	4	0	0	70	3	
	5	0	0	64	4	Regular
	6	0	0	62	5	
	7	0	0	66	6	
	10	0	0	72	7	
	11	0	0	60	8	
	12	0	0	33	9	
	13	0	0	30	a	
	14	0	0	23	b	
	15	0	0	16	c	
	16	0	0	22	d	
	17	0	0	20	e	
	20	0	0	26	f	
	21	0	0	13	g	
	22	0	0	05	h	
	23	0	0	14	i	
	24	0	0	32	j	
	25	0	0	36	k	
	26	0	0	11	l	
	27	0	0	07	m	
	30	0	0	06	n	
	31	0	0	03	o	
	32	0	0	15	p	
	33	0	0	35	q	
	34	0	0	12	r	
	35	0	0	24	s	
	36	0	0	01	t	
	37	0	0	34	u	
	40	0	0	17	v	
	41	0	0	31	w	

	42	0	0	27	x	
	43	0	0	25	y	
	44	0	0	21	z	
	45	0	0	56	-	(lc)
	46	0	0	44	=	
	47	0	0	54	+	
	50	0	0	46	,	
	51	0	0	42	.	(lc)
	52	0	0	50	|	(abs mag)
EB	0	0	0	42	|	
	1	0	0	22	.	
	2	0	0	21	,	
	3	0	0	63	+	
	4	0	0	76	=	
	5	0	0	02	-	
	6	0	0	74	z	
	7	0	0	73	y	
	10	0	0	72	x	
	11	0	0	71	w	
	12	0	0	70	v	
	13	0	0	67	u	
	14	0	0	66	t	
	15	0	0	65	s	
	16	0	0	54	r	
	17	0	0	53	q	
	20	0	0	52	p	
	21	0	0	51	o	
	22	0	0	50	n	
	23	0	0	47	m	
	24	0	0	46	l	
	25	0	0	45	k	
	26	0	0	44	j	
	27	0	0	34	i	
	30	0	0	33	h	
	31	0	0	32	g	
	32	0	0	31	f	
	33	0	0	30	e	
	34	0	0	27	d	
	35	0	0	26	c	
	36	0	0	25	b	
	37	0	0	24	a	
	40	0	0	14	9	
	41	0	0	13	8	
	42	0	0	12	7	
	43	0	0	11	6	
	44	0	0	10	5	
	45	0	0	07	4	
	46	0	0	06	3	
	47	0	0	05	2	
	50	0	0	04	1	
	51	0	0	03	0	
	52	0	0	01	Δ	
WS	0	0	0	1	Line counter	
	1	0	0	5	IJ counter for symbols/wd	
		CA	FC454			

Changes in Flex to Excess-Three Routine
To Produce the 1103A Version

```
        IA    FC33
IN33    MJ    0       IN35        Bypasses setting of MJ4
        CA    FC34

        IA    FC155
WB  0   EF    0       MS3         Write one block
    1   RP    10170   WB5  }
    2   EW    10000   BI   }     170(8) external writes
    3   0     0       0    }
    4   0     0       0    }     Unused area
    5   RP    10170   CD1  }
    6   TP    MS10    BI   }     Load image with all Δ's
        CA    FC164
```

B. Compilation Service Routines

This group of routines is used to start compilation and to act as a go-between for the different phases of Unicode. To begin a new compilation, set PAK = 77000 and start. In order to continue compilation from allocation on, set PAK = 77004 and start.

The various phases of Unicode exit to 77010 to read in the next phase and continue compilation.

Among the routines is a read routine and a print routine. The read routine reads one block of tape number 1. The parameter is sent to BB2 and an RJ BB BB1 is executed. The parameter has the form:

Op	u	v
0	N	(address)

where N ($\leq 170_8$) is the number of words to store and the address is the location of the first word. Words are stored in ascending order. This routine will reread a block six times in case of parity or sprocket errors and give an error print-out if all six attempts to read fail.

The print routine prints stored flex codes according to the parameter in PC2. The parameter has the form:

Op	u	v
0	(address)	N

where N is the number of words of codes to print and the address is the location of the first word. Successive words are picked up in ascending order.

With a start at either 77000 or 77004 the first block of tape 1 is read and the sentinel (in words 3 to 6) is checked. The first six words on the system tape are as follows:

		Op	u	v	
read into	7230	MJ	0	7236	for 77000 start
	7231	MJ	0	7243	for 77004 start
	7232	67	50342	65127	U N I C O D
	7233	30	77657	36566	E 77 S Y S T
	7234	30	47776	62452	E M 77 T A P
	7235	30	77777	77777	E 77 77 77 77 77

If words 3 to 6 are not as above, tape 1 is rewound and the following print-out occurs on the typewriter:

MOUNT UNICODE SYSTEM TAPE ON
SERVO 1. START.

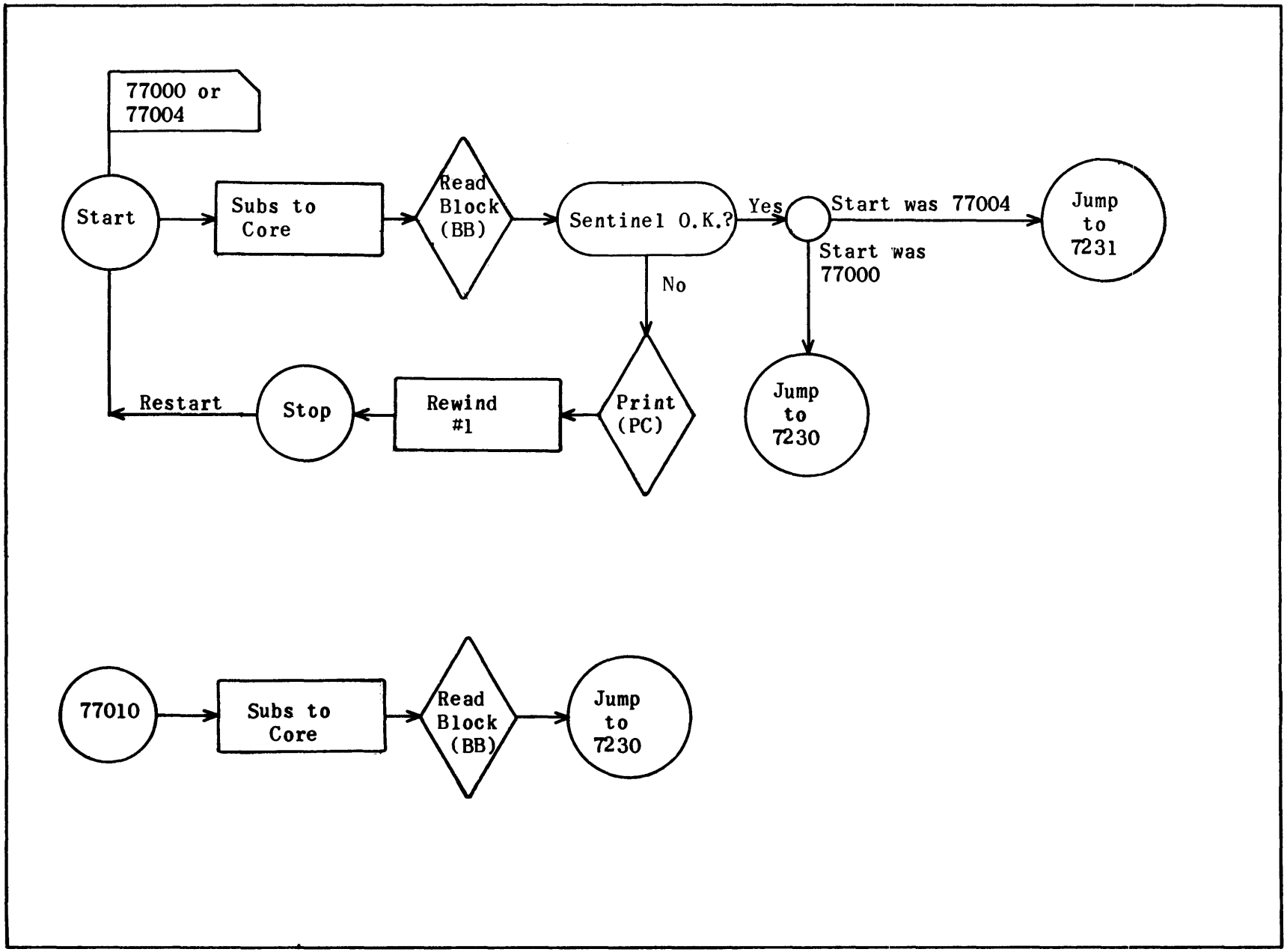
and the machine stops with PAK = 77000.

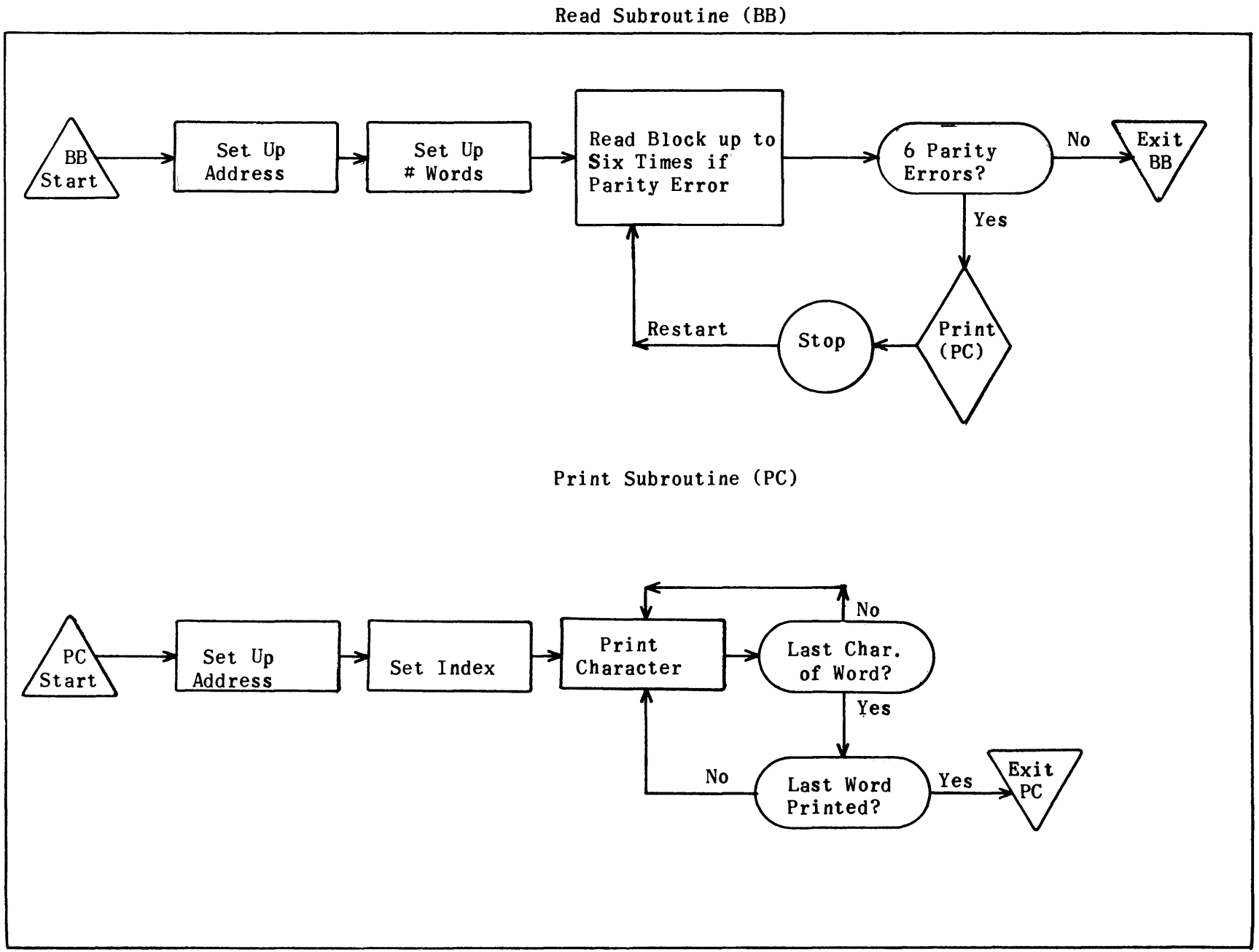
If the sentinel is correct the service routine jumps to

7230 in case of a 77000 start.
7231 in case of a 77004 start.

A jump to 77010 merely reads one block of Uniservo 1 into 7230-7417 and jumps to 7230.

In the sentinel checking region the read and the print routines are bootstrapped from drum to core before operating. The read and print routines are used by the coding in the first two blocks of the system tape and the two blocks of the Set-up Translation phase. The tape handler (TH) and the print text (UP) routines are not in the core at these times.


77000 or
77004
Start
Subs to
Core
Read
Block
(BB)
Sentinel O.K.?
Yes
Start was 77004
Jump
to
7231
Start was
77000
No
Jump
to
7230
Print
(PC)
Rewind
#1
Stop
Restart
77010
Subs to
Core
Read
Block
(BB)
Jump
to
7230


Read Subroutine (BB)
BB Start
Set Up Address
Set Up # Words
Read Block up to Six Times if Parity Error
6 Parity Errors?
No
Exit BB
Yes
Print (PC)
Stop
Restart
Print Subroutine (PC)
PC Start
Set Up Address
Set Index
Print Character
Last Char. of Word?
No
Yes
Last Word Printed?
No
Yes
Exit PC

Regional Assignments for Compilation Service Routine
(Both 1103A and 1105)

Drum		Core
ZA77000		
CA77014		
ZB77054	⟶	YC7070
BD77102	⟶	BB7116
PD77173	⟶	PC7207
		ZZ7230

Unicode Compilation Service Routines (1105)

		IA	ZA		
1st entrance	0	TP	ZA3	0	Set up zero
	1	RJ	ZA12	ZA12	Subs & 1st driver ⟶ core
	2	RJ	YC	YC1	⟶ 1st driver
	3	MJ	0	ZZ	⟶ 1st entrance of buffer
Entrance after stop	4	TP	ZA3	0	Set up zero
	5	RJ	ZA12	ZA12	Subs & 1st driver ⟶ core
	6	RJ	YC	YC1	⟶ 1st driver
	7	MJ	0	ZZ1	⟶ 2nd entrance of buffer
Merge & others	10	RJ	ZA12	ZA12	Subs ⟶ core
	11	MJ	0	YC23	2nd driver ⟶ core ⟶ 2nd driver
	12	RP	30140	(30000)	Drivers & subs ⟶ core
	13	TP	ZB	YC	
		CA	ZA14		

Constants (1105)

	IA	CA		
0	0	170	ZZ	
1	67	50342	65127	U N I C O D
2	30	77657	36566	E 77 S Y S T
3	30	47776	62452	E M 77 T A P
4	30	77777	77777	E 77 77 77 77 77
5	0	0	77777	V mask
6	0	0	1	
7	0	1	0	
10	0	CA15	12	Print parameter
11	0	CA27	11	
12	0	77777	0	u mask
13	2	200	10000	Rewind #1 TCU2
14	1	200	10000	Rewind #1 TCU1
15	47	12203	02204	↑ R E A D Δ
16	20	12120	31204	E R R O R Δ
17	46	15301	21401	(P A R I T
20	25	04031	20424	Y Δ O R Δ S
21	15	12031	63620	P R O C K E
22	01	42574	24704	T) ↓ . ↑ Δ
23	04	24013	01201	Δ S T A R T
24	04	26031	20412	Δ F O R Δ R
25	20	12203	02224	E R E A D S
26	57	42040	40404	↓ . Δ Δ Δ Δ
27	47	07033	40601	↑ M O U N T
30	04	34061	41603	Δ U N I C O
31	22	20042	42524	D E Δ S Y S
32	01	20070	40130	T E M Δ T A
33	15	20040	30604	P E Δ O N Δ
34	24	20121	70304	S E R V O Δ
35	57	52420	40447	↓ 1 . Δ Δ ↑
36	24	01301	20157	S T A R T ↓
37	42	04040	40404	. Δ Δ Δ Δ Δ
	CA	CA40		

```
            IA   ZB
YC      0   MJ   0       (30000)      Exit
        1   TP   CA      BB2  }       Read 1st blk.
        2   RJ   BB      BB1  }
        3   RP   30004   YC5  }
        4   CC   Z02     CA1  }
        5   SP   ZZ2     0    }       Check Sentinel
        6   RP   20003   YC10 }
        7   SA   ZZ3     0    }
       10   ZJ   YC11    YC           O.K.--> exit; no ↓
       11   MJ   20000   YC14         TCU2--> YC14; TCU1↓
       12   EF   0       CA14         Rewind #1 TCU1
       13   MJ   0       YC15
       14   EF   0       CA13         Rewind #1 TCU2
       15   TP   CA11    PC2  }       MOUNT UNICODE ON TAPE #1
       16   RJ   PC      PC1  }
       17   TV   YC      YC21 }       Set address
       20   RS   YC21    YC22 }
       21   MS   0       (30000)      Stop while tape changed
       22   0    0       3            Constant
       23   TP   CA      BB2  }       2nd driver
       24   RJ   BB      BB1  }
       25   MJ   0       ZZ           --> blk.
            CA   ZB26
```

Read One Block of Tape #1 (1105)

Parameter Format

Op	u	v
00	(No. words ≤ 170_8)	address

		IA	BD		
BB	0	MJ	0	(30000)	Exit
	1	MJ	0	BB3	Start
	2	0	(30000)	(30000)	Par. = 0 n address
	3	TP	BB56	Q	Mask → Q
	4	MJ	20000	BB10	TCU1 ↓ ; TCU2 → BB10
	5	TP	BB60	BB52	Set bypass #1
	6	RP	10005	BB13	Set TCU1
	7	QS	BB67	BB45	
	10	TP	BB57	BB52	Set bypass #2
	11	RP	10005	BB13	Set TCU2
	12	QS	BB70	BB45	
	13	EF	0	BB51	Set normal bias
	14	EF	0	BB52	Bypass #1 or #2
	15	TV	BB2	BB64	Set address
	16	TP	BB53	Q	Mask → Q
	17	QS	BB2	BB63	n of repeat
	20	QT	BB2	Q	n → Q
	21	TP	BB54	A	$170_8 \rightarrow A_u$
	22	ST	Q	A	$170_8 - n \rightarrow A_u$
	23	AT	BB55	BB65	$170_8 - n \rightarrow$ RP inst.
	24	RJ	BB65	BB62	Read block
	25	ER	0	A	IOA → A
	26	ZJ	BB27	BB43	Parity ↓ ; no →
	27	EF	0	BB50	Set high
	30	RJ	BB65	BB61	Reread
	31	ER	0	A	IOA → A
	32	ZJ	BB33	BB43	Parity ↓
	33	EF	0	BB47	Set low
	34	RJ	BB65	BB61	Reread
	35	ER	0	A	IOA → A
	36	ZJ	BB37	BB43	Parity ↓ ; no →
	37	EF	0	BB46	Move back one blk.
	40	TP	CA10	PC2	Print PARITY ERROR
	41	RJ	PC	PC1	
	42	MS	0	BB24	Stop for reread
	43	EF	0	BB51	Set normal
	44	MJ	0	BB	Exit
	45	2	602	10000	Read one block & stop
	46	2	14	10001	Move bwd one blk.
	47	2	1	70000	Low gain
	50	2	1	60000	High gain
	51	2	1	50000	Normal gain
	52	0	20000	04000	Set bypass

```
53   0    07777   0          Mask
54   0    170     0          120(10)
55   RP   0       (30000)    Set up RP
56   7    0       0          Mask
57   0    20000   04000      Bypass #2
60   0    10000   04000      Bypass #1
61   EF   0       BB46       Move bwd. one block  )
62   EF   0       BB45       Read one blk & stop  |
63   RP   1(0000) BB65   }   Read to core         } read sub.
64   ER   10000   (30000)}                        |
65   0    0       0          RP 0 exit } throw    |
66   ER   10000   A                    } away     )
67   1    0       0
70   2    0       0
     CA   BD71
```

Print (1105)

```
                IA  PD
PC      0       MJ  0        (30000)     Exit
        1       MJ  0        PC16        Start
        2       0   (30000)  0           Par. = 0  address n
        3       TP  PC2      Q
        4       QT  CA5      A45         Set index
        5       ST  CA6      PC20
        6       TU  PC2      PC7         Set address
        7       SP  (30000)  52
       10       PR  0        A           Print one word
       11       SS  A        6
       12       ZJ  PC10     PC13
       13       RA  PC7      CA7         Set for next word
       14       IJ  PC20     PC7         Finished ↓; no ⟶ PC7
       15       MJ  0        PC          Exit
       16       RP  4        PC3         Four carriage returns
       17       PR  0        PC4
       20       0   0        0           Index
                CA  PD21
```

		IA	ZA		
1st entrance	0	TP	ZA3	0	Set up zero
	1	RJ	ZA12	ZA12	Subs and 1st driver → core
	2	RJ	YC	YC1	→ 1st driver
	3	MJ	0	ZZ	→ 1st entrance of buffer
Entrance after stop	4	TP	ZA3	0	Set up zero
	5	RJ	ZA12	ZA12	Subs and 1st driver → core
	6	RJ	YC	YC1	→ 1st driver
	7	MJ	0	ZZ1	→ 2nd entrance of buffer
Merge and others	10	RJ	ZA12	ZA12	Subs → core
	11	MJ	0	YC20	2nd driver → core → 2nd driver
	12	RP	30140	(30000)	Driver & subs → core
	13	TP	ZB	YC	
		CA	ZA14		

Constants (1103A)

	IA	CA		
0	0	170	ZZ	ZZ prob. = 7230
1	67	50342	65127	U N I C O D
2	30	77657	36566	E 77 S Y S T
3	30	47776	62452	E M 77 T A P
4	30	77777	77777	E 77 77 77 77 77
5	0	0	77777	V mask
6	0	0	1	
7	0	1	0	
10	0	CA13	10	Print parameters
11	0	CA23	11	
12	0	77777	0	u mask
13	47	12203	02204	↑ R E A D Δ
14	20	12120	31204	E R R O R Δ
15	46	15301	21401	(P A R I T
16	25	42574	24704	Y) ↓ . ↑ Δ
17	04	24013	01201	Δ S T A R T
20	04	26031	20412	Δ F O R Δ R
21	20	12203	02224	E R E A D S
22	57	42040	40404	↓ . Δ Δ Δ Δ
23	47	07033	40601	↑ M O U N T
24	04	34061	41603	Δ U N I C O
25	22	20042	42524	D E Δ S Y S
26	01	20070	40130	T E M Δ T A
27	15	20040	30604	P E Δ O N Δ
30	24	20121	70304	S E R V O Δ
31	57	52420	40447	↓ 1 . Δ Δ ↑
32	24	01301	20157	S T A R T ↓
33	42	04040	40404	. Δ Δ Δ Δ Δ
34	2	200	10000	Rewind Uniservo 1
	CA	CA35		

FLEX CODES

1st and 2nd Drivers (1103A)

```
          IA  ZB
YC    0   MJ  0       (30000)     Exit
      1   TP  CA      BB2    }
      2   RJ  BB      BB1    }   Read 1st blk
      3   RP  30004   YC5    )
      4   CC  ZZ2     CA1    |
      5   SP  ZZ2     0      }   Check Sentinel
      6   RP  20003   YC10   |
      7   SA  ZZ3     0      )
     10   ZJ  YC11    YC          O.K. ---> exit;     no ↓
     11   EF  0       CA34        Rewind tape #1
     12   TP  CA11    PC2    }    UNICODE NOT ON TAPE #1
     13   RJ  PC      PC1    }
     14   TV  YC      YC16        Exit - 3
     15   RS  YC16    YC17
     16   MS  0       (30000)     Stop while tapes changed
     17   0   0       3           Constant
     20   TP  CA      BB2    }
     21   RJ  BB      BB1    }   2nd driver
     22   MJ  0       ZZ          ---> blk
          CA  ZB23
```

Read One Block of Tape #1 (1103A)

Parameter Format

Op	u	v
00	(No. words ≤ 170_8)	address
TP (par)	BB2	
RJ BB	BB1	

```
          IA   BD
BB    0   MJ   0       (30000)      Exit
      1   MJ   0       BB3          Start
      2   0    30000   30000        Par.
      3   MJ   0       BB4          Set normal bias
      4   TV   BB2     BB47         Set address
      5   TP   BB41    Q            Mask --> Q
      6   QS   BB2     BB46         n of RP
      7   QT   BB2     Q            n --> Q
     10   TP   BB42    A      }
     11   ST   Q       A      }     170(8) - n --> Au
     12   AT   BB43    BB50         170(8) - n --> RP inst.
     13   RJ   BB50    BB45         Read block
     14   ER   0       A            IOA --> A
     15   ZJ   BB16    BB32         Parity down; no --> BB32
     16   EF   0       BB37         High
     17   RJ   BB50    BB44         Reread
     20   ER   0       A            IOA --> A
     21   ZJ   BB22    BB32         Parity down; no --> BB32
     22   EF   0       BB36         Low
     23   RJ   BB50    BB44         Reread
     24   ER   0       A            IOA --> A
     25   ZJ   BB26    BB32         Parity down
     26   EF   0       BB35         Move bwd one blk.
     27   TP   CA10    PC2    }     Print parity error
     30   RJ   PC      PC1    }
     31   MS   0       BB13         Stop for reread
     32   EF   0       BB40         Set normal
     33   MJ   0       BB           Exit
     34   2    602     10000        Read one blk & stop
     35   2    14      10001        Move bwd. one blk.
     36   2    1       70000        Low
     37   2    1       60000        High
     40   2    1       50000        Normal
     41   0    07777   0            Mask
     42   0    170     0            120(10)
     43   RP   0       (30000)
     44   EF   0       BB35         Move bwd one blk.
     45   EF   0       BB34         Read one blk. & stop
     46   RP   1(0000) BB50   }      Read to core
     47   ER   10000   (30000)}     RP 0 exit } throw away
     50   0    0       0
     51   ER   10000   A
          CA   BD52
```

read sub.

Print (1103A)

```
          IA  PD
PC    0   MJ  0        (30000)      Exit
      1   MJ  0        PC16         Start
      2   0   (30000)  (0)          Par. = 0 address n
      3   TP  PC2      Q
      4   QT  CA5      A45          Set index
      5   ST  CA6      PC20
      6   TU  PC2      PC7          Set address
      7   SP  (30000)  52
     10   PR  0        A
     11   SS  A        6            Print one word
     12   ZJ  PC10     PC13
     13   RA  PC7      CA7          Set for next word
     14   IJ  PC20     PC7          Finished ↓ ; no → PC7
     15   MJ  0        PC           Exit
     16   RP  4        PC3
     17   PR  0        PC4
     20   0   0        0            Index
          CA  PD21
```

C. Object Program Service Routines

1. Flex Code Print-out

Stored at: 77250-77272

Operates at: 750-771

This routine does not preserve all of HSS.

Function

To print out on-line stored Flex codes.

Calling Sequence

```
α    TP   L (Parameter)   PR3
+1   RJ   PR2             PR
+2        Control returned here
```

Notes:

No editing is done - therefore all carriage return codes, etc., should be provided.

Parameter is of form 0 - u - v, where u = initial address of stored codes (6 per word)
v = number of words to be printed.

This routine is used by the Object Program Loader, Sections 1 and 2 of Initialization, and by some of the generated routines of the Object Program.

Regions for Flex Print Routine

```
        RE  PR77250          Storage address
        RE  RP750            Operating area
```

Flex Print-Out (N.B. - Stored at PR; Operates at RP)
(Core Requirement = 22 Locations)

```
        IA  PR
        RP  30021  RP2  }
        TP  PR2    RP   }   Send down to core
RP   0  MJ  0      (30000)  Exit                    )
     1  0   30000  (30000)  Parameter               |
     2  TV  RP1    RP20     Set up index            |
     3  TU  RP1    RP6                              |
     4  1J  RP20   RP6                              |
     5  MJ  0      RP                               |
     6  TP (30000) Q                                } Operating routine
     7  LQ  Q      6                                |
    10  PR  0      Q                                |
    11  QT  RP17   RP21                             |
    12  RS  Q      RP21                             |
    13  ZJ  RP7    RP14                             |
    14  RA  RP6    RP16                             |
    15  MJ  0      RP4                              |
    16  0   1      0                                |
    17  0   0      77                               |
    20  0   0      0                                )
        CA  PR23
```

2. Object Program Loaders

These routines, one for the 1103A and one for the 1105, perform the initial loading of data from magnetic tape on Uniservo 1 to High Speed Storage, and on recognition of a transfer block, transfer control to an address indicated on the tape.

The tape format on which these routines depend for proper loading of the Object Program is prepared by the Initialization Generation Phase according to the requirements of the problem.

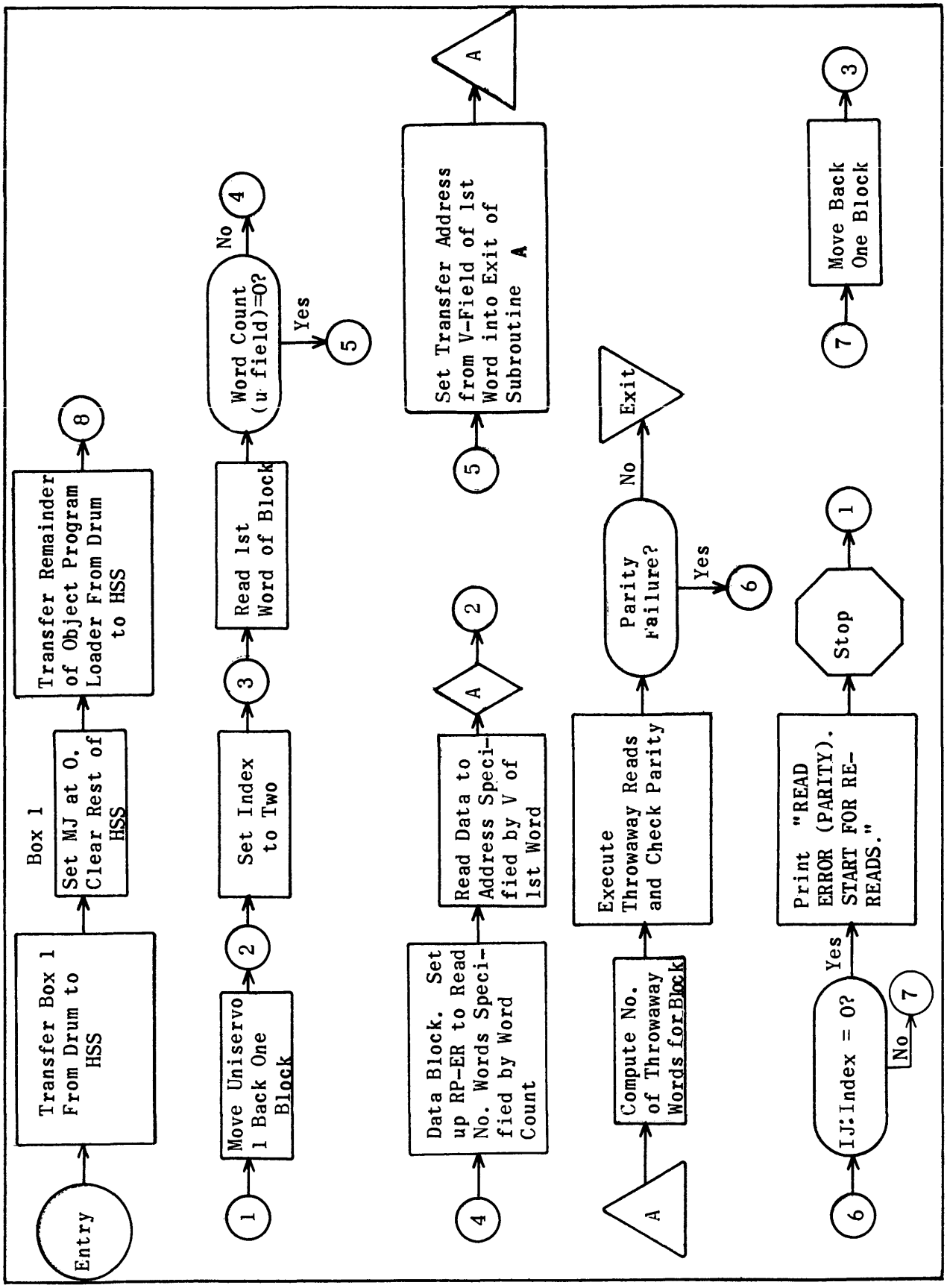
The first word of each block has special significance. It either contains the loading address and word count of the block or a transfer address.

In the former case, the u portion holds the word count (at most 167_8) and v holds the initial loading address, from which as many words as are specified by the word count are stored sequentially as read from tape. If the word count is less than 167_8, but not zero, the remainder of the block is disregarded.

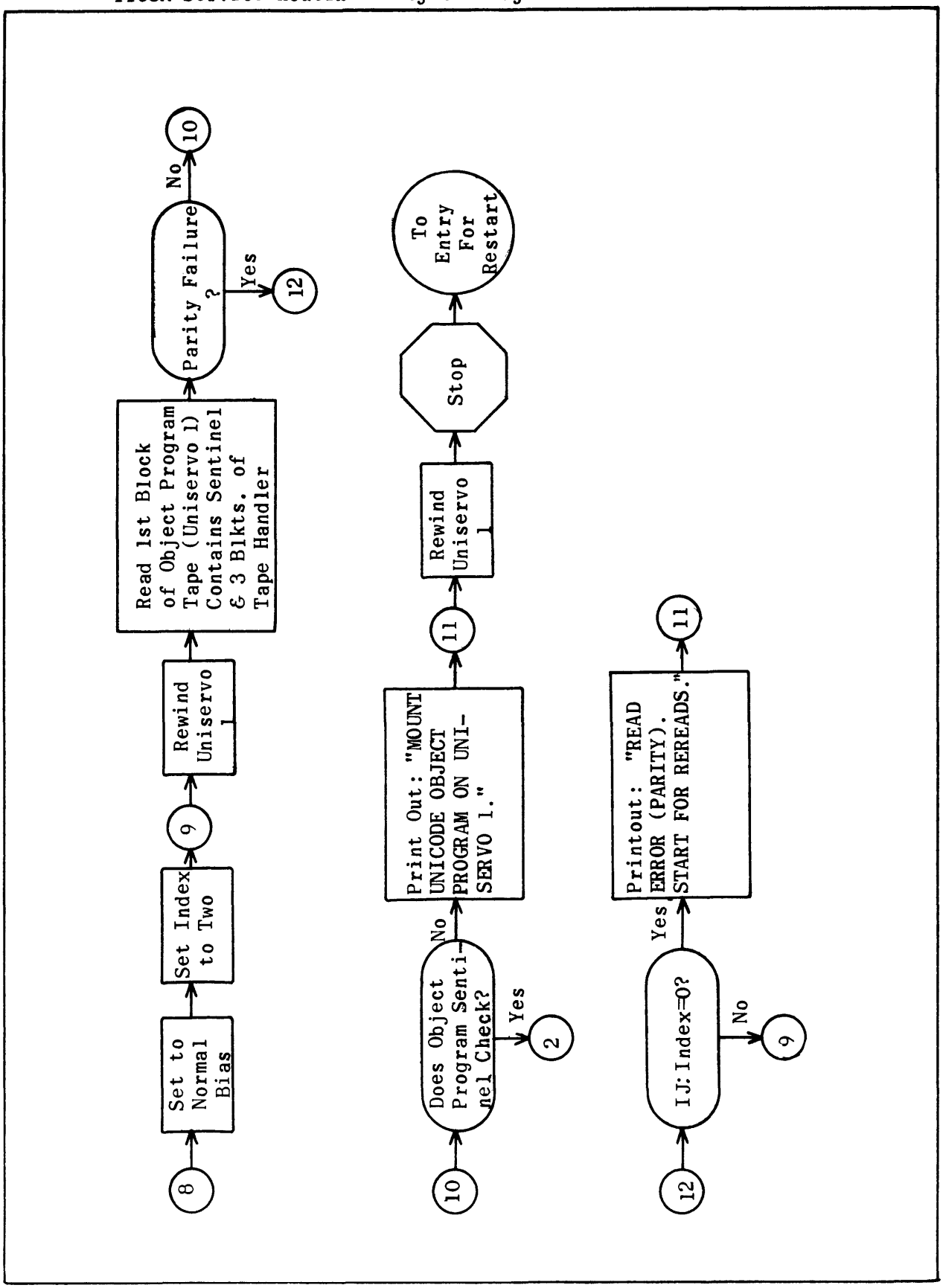
The transfer word is recognized as the first word of a block because its u field is zero (or a zero word count). In this case the remainder of the block is read but disregarded, and control is transferred to the location specified by the contents of the v field.

For further description of the loading of Object Programs, refer to the Initialization Generation write-up and its related diagram showing the layout of the Object Program tape.

1103A Service Routine - Object Program Loader Flow Chart



1103A Service Routine - Object Program Loader Flow Chart (Cont.)



Regions for 1103A Object Program Loader

```
        RE  DA77300              Storage address
        RE  PR77250              Flex print routine
        RE  LD1500   )
        RE  LE1564   }           Operating area
        RE  LT1624   )
        RE  SN114                Address for loading Sentinel block
```

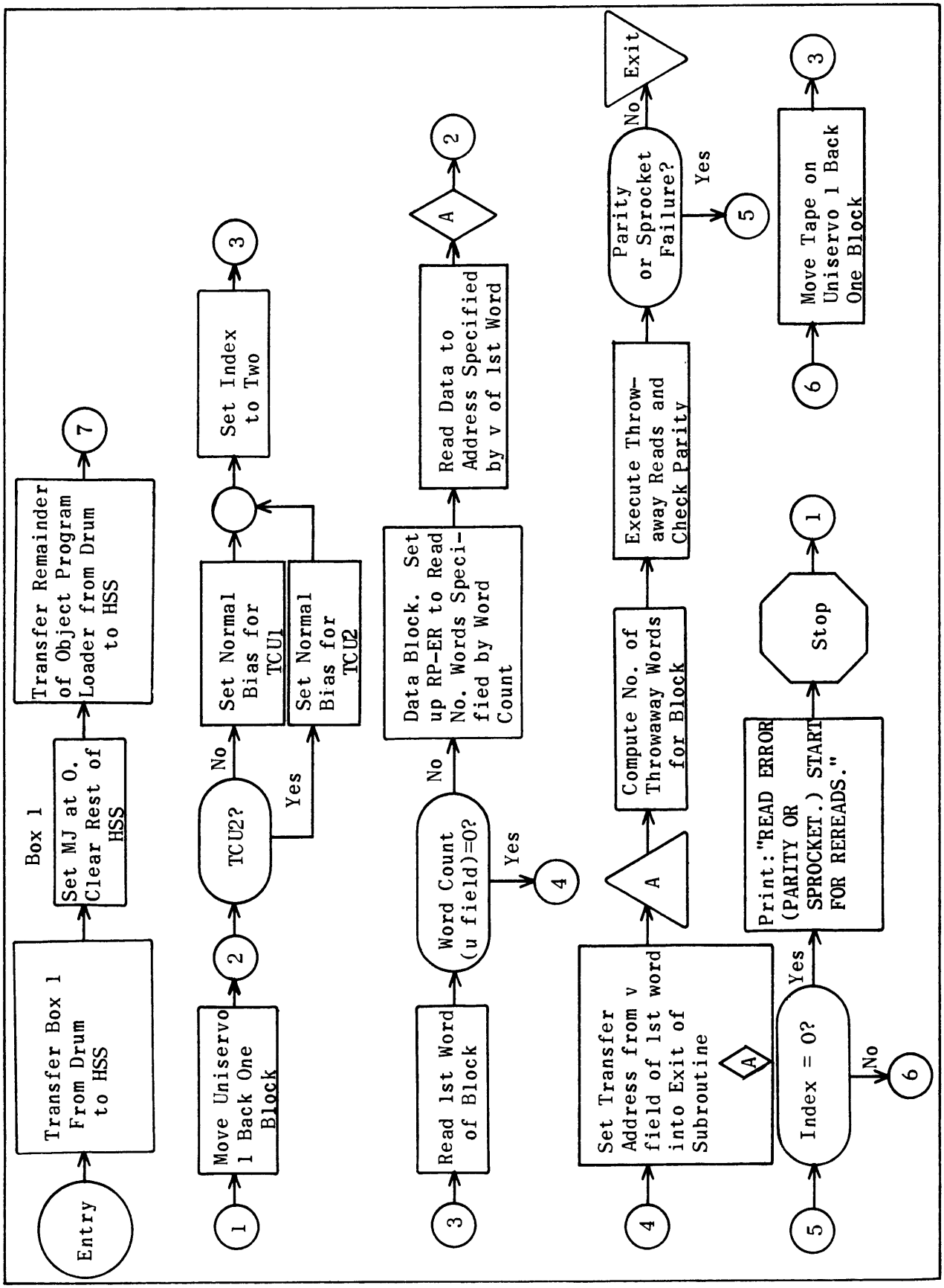
Object Program Loader (1103A)

```
          IA  DA
    0     RP  30004   0
    1     TP  DA4     0
    2     RP  30124   LD34
    3     TP  DA10    LD
0   4     MJ  0       2
1   5     0   0       0
2   6     RP  17776   DA2  )       Clear one bank of core
3   7     TP  1       2    }
          CA  DA10
```

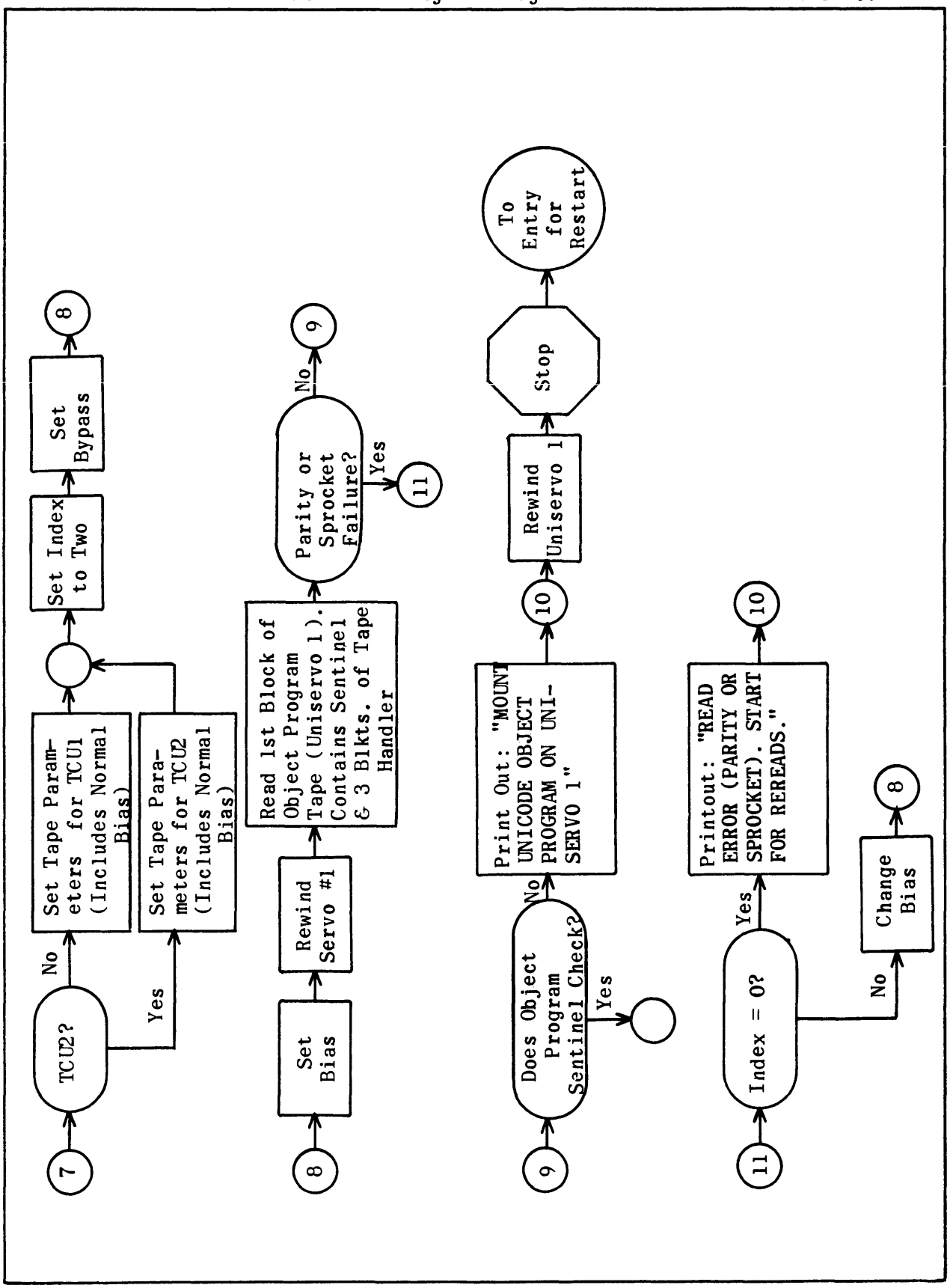
```
                IA   DA10
(1) LD    0     EF   0       LE1        Move back one block
(2)       1     TP   LE      LT2        Set to normal bias
          2     TP   LE7     LT1        Set index
(3)       3     MJ   167     LD4
          4     EF   0       LE2        Prepare to read one block
          5     ER   10000   Q          Read one word
          6     QT   LE5     LT         Save word count
          7     ZJ   LD10    LD16       = 0 ?
(4)      10     TV   Q       LD13       Non-zero so set up Read and Repeat
         11     AT   LE6     LD12
         12     0    30000   30000      Read info to Loading Address
         13     ER   10000   30000
         14     RJ   LD24    LD17       Check parity and throw away
         15     MJ   0       LD1        Back to read next block
(5)      16     TV   Q       LD24       Set transfer address
         17     TU   LD3     LD21       Determine no. of throw away words
         20     RS   LD21    LT
         21     RP   0       LD23       Throw away
         22     ER   10000   A
         23     ER   0       A          Check parity
         24     ZJ   LD25    30000
(6)      25     IJ   LT1     LD31
         26     TP   LE10    PR3        Print READ ERROR
         27     RJ   PR2     PR
         30     MS   0       LD
         31     RA   LT2     LE4        Change bias
(7)      32     EF   0       LE1        Move back one block
         33     MJ   0       LD3
(8)      34     TP   LE      LT2        Set to normal bias
         35     TP   LE7     LT1        Set index
         36     MJ   0       LD37
(9)      37     EF   0       LE3        Rewind Uniservo 1
         40     EF   0       LE2        Prepare to Read one block Uniservo 1
         41     RP   10170   LD43       Read 1st block, Sentinel + 3 blkts.
         42     ER   10000   SN           of Tape Handler.
         43     ER   0       A          Parity error?
         44     ZJ   LD45    LD52
(12)     45     IJ   LT1     LD50
         46     TP   LE10    PR3        Print READ ERROR
         47     MJ   0       LD55
         50     RA   LT2     LE4        Change bias
         51     MJ   0       LD36       Try again
(10)     52     TP   (SN24)  A
         53     EJ   (LE32)  LD60
         54     TP   LE11    PR3        Print MOUNT UNICODE OBJECT PROGRAM
         55     RJ   PR2     PR
(11)     56     EF   0       LE3        Rewind
         57     MS   0       DA
         60     RA   LD52    LE36       Increment test instructions
         61     RA   LD53    LE36
         62     IJ   LE37    LD52
         63     MJ   0       LD1
                CA   DA74
```

		IA	DA74			
LE	0	02	1	50000	Normal bias	
	1	02	14	10001	Move Uniservo 1 back one block	
	2	02	602	10000	Read Uniservo 1 one block	
	3	02	200	10000	Rewind Uniservo 1	
	4	0	0	10000	Change bias	
	5	0	177	0	Word count mask	
	6	RP	10000	LD14	Dummy	
	7	0	0	2	Index for parity rereads	
	10	0	LE12	10	Parameter for READ Error	
	11	0	LE22	10	Parameter for wrong tape	
	12	45	45471	22030	CR CR ↑ R E A	
	13	22	04201	21203	D Δ E R R 0	
	14	12	04461	53012	R Δ (P A R	
	15	14	01254	25742	I T Y) ↓ .	
	16	04	04472	40130	Δ Δ ↑ S T A	
	17	12	01042	60312	R T Δ F 0 R	Flex
	20	04	12201	22030	Δ R E R E A	
	21	22	24574	24545	D S ↓ . CR CR	
	22	45	45470	70334	CR CR ↑ M 0 U	
	23	06	01043	40614	N T Δ U N I	
	24	16	03222	00403	C 0 D E Δ 0	
	25	23	32201	60104	B J E C T Δ	Flex
	26	15	12031	31230	P R 0 G R A	
	27	07	04030	60424	M Δ 0 N Δ S	
	30	20	12170	30457	E R V 0 Δ ↓	
	31	52	42040	44545	1 . Δ Δ CR CR	
	32	67	50342	65127	U N I C 0 D	Excess Three
	33	30	01512	54430	E Δ 0 B J E	
	34	26	66015	25451	C T Δ P R 0	
	35	32	54244	72201	G R A M . Δ	
	36	0	1	0	u-advance	
	37	0	0	3	Index for Sentinel check	
		CA	DA134			

1105 Service Routine - Object Program Loader Flow Chart



1105 Service Routine - Object Program Loader Flow Chart (Cont.)



Regions for 1105 Object Program Loader

```
RE  DA77300                 Storage address
RE  PR77250                 Flex print routine

RE  LD1500   )
RE  LE1575   }              Operating area
RE  LT1645   )
RE  SN114                   Address for loading Sentinel block
```

Object Program Loader (1105)

```
      IA  DA
   0  RP  30004     0
   1  TP  DA4       0
   2  RP  30145     LD40
   3  TP  DA10      LD
0  4  MJ  0         2
1  5  0   0         0
2  6  RP  17776     DA2 )
3  7  TP  1         2   }      Clear one bank of core
      CA  DA10
```

```
                IA    DA10
(1) LD   0      EF    0       LT3      Move back one block
(2)      1      MJ    20000   LD4      TCU1 or TCU2?
         2      TP    LE      LT2      Set normal bias for TCU1
         3      MJ    0       LD5
         4      TP    LE5     LT2      Set normal bias for TCU2
         5      EF    167     LT2      Set Bias
         6      TP    LE15    LT1      Set Index
         7      EF    0       LT5      Set Bypass
(3)     10      EF    0       LT4      Prepare to read one block
        11      ER    10000   Q        Read one word
        12      QT    LE13    LT       Set word count into temp.
        13      ZJ    LD14    LD22     = 0?
        14      TV    Q       LD17     Non-zero, so set up Read
        15      AT    LE14    LD16       and repeat
        16      0     30000   30000    Read info to loading address
        17      ER    10000   30000
        20      RJ    LD30    LD23     Check parity and throw away
        21      MJ    0       LD1      Back to read next block
(4)     22      TV    Q       LD30     Set transfer address
        23      TU    LD5     LD25     Determine no. of throw away words
        24      RS    LD25    LT
        25      RP    0       LD27     Throw away
        26      ER    10000   A
        27      ER    0       A        Check Parity
        30      ZJ    LD31    30000
(5)     31      IJ    LT1     LD35     Parity failure
        32      TP    LE16    PR3      Print  READ ERROR
        33      RJ    PR2     PR
        34      MS    0       LD
        35      RA    LT2     LE12     Change bias
        36      EF    0       LT3      Move back one block
        37      MJ    0       LD7
(7)     40      MJ    20000   LD43     TCU1 or TCU2?
        41      RP    30005   LD45     Set TCU1 parameters
        42      TP    LE      LT2
        43      RP    30005   LD45     Set TCU2 parameters
        44      TP    LE5     LT2
        45      TP    LE15    LT1      Set Index
        46      EF    0       LT5      Set Bypass
(8)     47      EF    0       LT2      Set Bias
        50      EF    0       LT6      Rewind Uniservo 1
        51      EF    0       LT4      Prepare to read first block
        52      RP    10170   LD54     Read 1st block, Sentinel + 3 blkts. of
        53      ER    10000   SN         Tape Handler
        54      ER    0       A
        55      ZJ    LD56    LD63     Parity or sprocket error?
        56      IJ    LT1     LD61
        57      TP    LE16    PR3      READ ERROR
        60      MJ    0       LD66
        61      RA    LT2     LE12     Change bias
```

```
      62  MJ  0       LD47          Try again
⑨     63  TP  (SN24)  A
      64  EJ  (LE42)  LD71
      65  TP  LE17    PR3  }        Print MOUNT UNICODE OBJECT PROGRAM
      66  RJ  PR2     PR   }
      67  EF  0       LT6           Rewind
      70  MS  0       DA
      71  RA  LD63    LE46 }        Increment test instructions
      72  RA  LD64    LE46 }
      73  IJ  LE47    LD63
      74  MJ  0       LD1           To continue loading
          CA  DA105
```

```
        IA   DA105
LE   0  01   1      50000   Normal bias                                  )
     1  01   14     10001   Move Uniservo 1 back one block   } TCU1
     2  01   602    10000   Read Uniservo 1                              )
     3  0    10000  4000    Buffer 1 bypass                              )
     4  01   200    10000   Rewind Uniservo 1                            )
     5  02   1      50000   Normal bias                                  )
     6  02   14     10001   Move Uniservo 1 back one block               )
     7  02   602    10000   Read Uniservo 1                  } TCU2      )
    10  0    20000  4000    Buffer 2 bypass                              )
    11  02   200    10000   Rewind Uniservo 1                            )
    12  0    0      10000   Change bias
    13  0    177    0       Word count mask
    14  RP   10000  LD20    Dummy
    15  0    0      2       Index
    16  0    LE20   12      Parameter for Read Error
    17  0    LE32   10      Parameter for Wrong Tape on Uniservo 1
    20  45   45471  22030   CR CR ↑ R E A
    21  22   04201  21203   D Δ E R R 0
    22  12   04461  53012   R Δ ( P A R
    23  14   01250  40312   I T Y Δ 0 R
    24  04   24151  20316   Δ S P R 0 C                    Flex
    25  36   20014  25742   K E T ) ↓ .
    26  04   04472  40130   Δ Δ ↑ S T A
    27  12   01042  60312   R T Δ F 0 R
    30  04   12201  22030   Δ R E R E A
    31  22   24574  24545   D S ↓ . CR CR
    32  45   45470  70334   CR CR ↑ M 0 U
    33  06   01043  40614   N T Δ U N I                    Flex
    34  16   03222  00403   C 0 D E Δ 0
    35  23   32201  60104   B J E C T Δ
    36  15   12031  31230   P R 0 G R A
    37  07   04030  60424   M Δ 0 N Δ S
    40  20   12170  30457   E R V 0 Δ ↓
    41  52   42040  44545   1 . Δ Δ CR CR
    42  67   50342  65127   U N I C 0 D                    XS-3
    43  30   01512  54430   E Δ 0 B J E
    44  26   66015  25451   C T Δ P R 0
    45  32   54244  72201   G R A M . Δ
    46  0    1      0       u-advance
    47  0    0      3       Index for Sentinel check
        CA   DA155
```

2. LIBRARY ROUTINES

2. Library Routines

a. Unicode Librarian Symbolic Listing

```
                                        X    1.  ,        , SUB     ,WFUA05   ,714      ,UNICODE LIBRARIAN      $
                                        X    2.  ,        , INOUT   ,3        ,0        ,                       $
00004  00004  45  00000  00012          X    3.  ,        , MJ      ,         ,ENTRY    ,                       $
00005  00005  37  40005  40007          X    4.  , ALARM  , ALARM   ,         ,         ,                       $
00006  00006  45  00000  30000          X    5.  , EXIT   , MJ      ,         ,FILL     ,                       $
00007  00007  00  00000  00000          X    6.  , PAR1   ,         ,         ,0        ,PARAMETERS             $
00010  00010  00  00000  00000          X    7.  , PAR2   ,         ,         ,0        ,                       $
00011  00011  00  00000  00000          X    8.  , PAR3   ,         ,         ,0        ,                       $
                                        X    9.  , RTN    , EQUALS  ,60000)B  ,60000)B  ,                       $
00012  00012  55  00420  31006          X   10.  , ENTRY  , LQ      ,C1       ,Q+6      ,IS INPUT SERVO ZERO    $
00013  00013  51  00007  32000          X   11.  ,        , QT      ,PAR1     ,A        ,                       $
00014  00014  47  00042  00015          X   12.  ,        , ZJ      ,S1       ,L+1      ,                       $
00015  00015  75  11130  00017          X   13.  ,        , RPV     ,600      ,L+2      ,YES-LOAD BUFFER        $
00016  00016  11  00421  01316          X   14.  ,        , TP      ,C2       ,BF       ,                       $
00017  00017  11  00422  01342          X   15.  ,        , TP      ,C3       ,BF+20    ,                       $
00020  00020  11  00423  01343          X   16.  ,        , TP      ,C4       ,BF+21    ,                       $
00021  00021  11  00424  01506          X   17.  ,        , TP      ,C5       ,BF+120   ,                       $
00022  00022  11  00425  01507          X   18.  ,        , TP      ,C6       ,BF+121   ,                       $
00023  00023  23  01510  32000          X   19.  ,        , RS      ,BF+122   ,A        ,                       $
00024  00024  11  00426  01676          X   20.  ,        , TP      ,C7       ,BF+240   ,                       $
00025  00025  11  00427  01677          X   21.  ,        , TP      ,C8       ,BF+241   ,                       $
00026  00026  23  01700  32000          X   22.  ,        , RS      ,BF+242   ,A        ,                       $
00027  00027  11  00422  02066          X   23.  ,        , TP      ,C3       ,BF+360   ,                       $
00030  00030  11  00430  02067          X   24.  ,        , TP      ,C9       ,BF+361   ,                       $
00031  00031  11  00431  02256          X   25.  ,        , TP      ,C10      ,BF+480   ,                       $
00032  00032  11  00432  02257          X   26.  ,        , TP      ,C11      ,BF+481   ,                       $
00033  00033  11  00433  31000          X   27.  ,        , TP      ,C12      ,Q        ,SET SERVO NUMBERS      $
00034  00034  75  10010  00036          X   28.  ,        , RPV     ,8        ,L+2      ,                       $
00035  00035  53  00434  00663          X   29.  ,        , QS      ,C13      ,MTWRIT   ,                       $
00036  00036  31  00477  00000          X   30.  ,        , SP      ,5V       ,0        ,WRITE DUMMY LIBRARY    $
00037  00037  37  00543  00533          X   31.  ,        , RJ      ,MTWREX   ,MTWRE 1  ,                       $
00040  00040  37  00635  00622          X   32.  ,        , RJ      ,MTCHEX   ,MTCHEK   ,CHECK DUMMY LIBRARY    $
00041  00041  45  00000  00046          X   33.  ,        , MJ      ,         ,S2       ,                       $
00042  00042  54  32000  00006          X   34.  , S1     , LA      , A       ,6        ,SET SERVO NUMBERS      $
```

```
00043  00043  11  00433  31000  X  35.  ,         ,  TP  ,C12      ,Q          ,                          $
00044  00044  75  10004  00046  X  36.  ,         ,  RPV ,4        ,L+2        ,                          $
00045  00045  53  32000  00667  X  37.  ,         ,  QS  ,A        ,MTRDFW     ,                          $
00046  00046  54  00007  32014  X  38.  , S2      ,  LA  ,PAR1     ,A+12       ,                          $
00047  00047  75  10004  00051  X  39.  ,         ,  RPV ,4        ,L+2        ,                          $
00050  00050  53  32000  00663  X  40.  ,         ,  QS  ,A        ,MTWRIT     ,                          $
00051  00051  31  00474  00000  X  41.  ,         ,  SP  ,2V       ,0          ,READ TAPE AND CATALOGUE   $
                                                                                  (LABELS)
00052  00052  37  00565  00555  X  42.  ,         ,  RJ  ,MTRDEX   ,MTREAD     ,                          $
00053  00053  31  01342  00000  X  43.  ,         ,  SP  ,BF+20    ,           ,ALARM IF NO TAPE LABEL    $
00054  00054  43  00422  00067  X  44.  ,         ,  EJ  ,C3       ,S3         ,                          $
00055  00055  11  00436  31000  X  45.  ,         ,  TP  ,C15      ,Q          ,                          $
00056  00056  31  00477  00000  X  46.  ,         ,  SP  ,5V       ,0          ,                          $
00057  00057  55  31000  00006  X  47.  ,         ,  LQ  ,Q        ,6          ,                          $
00060  00060  61  00000  31000  X  48.  ,         ,  PR  ,         ,Q          ,                          $
00061  00061  41  32000  00057  X  49.  ,         ,  IJ  ,A        ,L-2        ,                          $
00062  00062  17  00000  00660  X  50.  , ALARM1  ,  EF  ,         ,MTNOBI     ,                          $
00063  00063  17  00000  00670  X  51.  ,         ,  EF  ,         ,MTINRW     ,                          $
00064  00064  17  00000  00665  X  52.  ,         ,  EF  ,         ,MTOURW     ,                          $
00065  00065  11  00437  31000  X  53.  ,         ,  TP  ,C16      ,Q          ,                          $
00066  00066  45  00000  00005  X  54.  ,         ,  MJ  ,         ,ALARM      ,                          $
00067  00067  11  01510  00520  X  55.  , S3      ,  TP  ,BF+122   ,T1         ,STORE WORD COUNT          $
00070  00070  55  00007  31014  X  56.  ,         ,  LQ  ,PAR1     ,Q+12       ,REVISE WORD COUNT         $
00071  00071  51  00440  00521  X  57.  ,         ,  QT  ,C17      ,T2         ,                          $
00072  00072  47  00073  00074  X  58.  ,         ,  ZJ  ,L+1      ,L+2        ,                          $
00073  00073  23  01510  00474  X  59.  ,         ,  RS  ,BF+122   ,2V         ,                          $
00074  00074  55  00010  31014  X  60.  ,         ,  LQ  ,PAR2     ,Q+12       ,                          $
00075  00075  51  00440  00522  X  61.  ,         ,  QT  ,C17      ,T3         ,                          $
00076  00076  47  00077  00100  X  62.  ,         ,  ZJ  ,L+1      ,L+2        ,                          $
00077  00077  21  01510  00474  X  63.  ,         ,  RA  ,BF+122   ,2V         ,                          $
00100  00100  21  00521  00504  X  64.  ,         ,  RA  ,T2       ,120V       ,                          $
00101  00101  21  00522  00504  X  65.  ,         ,  RA  ,T3       ,120V       ,                          $
00102  00102  11  01510  00523  X  66.  ,         ,  TP  ,BF+122   ,T4         ,STORE REVISED WORD        $
                                                                                  COUNT
00103  00103  31  00474  00000  X  67.  ,         ,  SP  ,2V       ,0          ,WRITE TAPE AND            $
                                                                                  CATALOGUE LABELS
00104  00104  37  00543  00533  X  68.  ,         ,  RJ  ,MTWREX   ,MTWRE1     ,                          $
```

```
00105  00105  31  00520  00000  X  69.  ,     , SP  ,T1     ,      ,READ CATALOGUE          $
00106  00106  73  00504  00520  X  70.  ,     , DV  ,120V   ,T1    ,                       $
00107  00107  47  00110  00111  X  71.  ,     , ZJ  ,L+1    ,L+2   ,                       $
00110  00110  21  00520  00473  X  72.  ,     , RA  ,T1     ,1V    ,                       $
00111  00111  31  00520  00000  X  73.  , S4  , SP  ,T1     ,      ,                       $
00112  00112  75  12000  00114  X  74.  ,     , RPV ,1024   ,L+2   ,                       $
00113  00113  11  00421  01316  X  75.  ,     , TP  ,C2     ,BF    ,                       $
00114  00114  37  00565  00555  X  76.  ,     , RJ  ,MTRDEX ,MTREAD,                       $
00115  00115  37  00115  00116  X  77.  , S5  , RJ  ,L      ,L+1   ,                       $
00116  00116  55  00007  31030  X  78.  ,     , LQ  ,PAR1   ,Q+24  ,GENERATE CATALOGUE     $
                                                                      ITEM
00117  00117  51  00502  32000  X  79.  ,     , QT  ,7V     ,A     ,                       $
00120  00120  47  00121  00132  X  80.  ,     , ZJ  ,L+1    ,S6    ,                       $
00121  00121  35  00443  00525  X  81.  ,     , AT  ,C21    ,T6    ,                       $
00122  00122  11  00011  31000  X  82.  ,     , TP  ,PAR3   ,Q     ,                       $
00123  00123  11  00477  00524  X  83.  ,     , TP  ,5V     ,T5    ,                       $
00124  00124  55  31000  00006  X  84.  ,     , LQ  ,Q      ,6     ,                       $
00125  00125  51  00420  32000  X  85.  ,     , QT  ,C1     ,A     ,                       $
00126  00126  47  00130  00127  X  86.  ,     , ZJ  ,L+2    ,L+1   ,                       $
00127  00127  21  31000  00420  X  87.  ,     , RA  ,Q      ,C1    ,                       $
00130  00130  41  00524  00124  X  88.  ,     , IJ  ,T5     ,L-4   ,                       $
00131  00131  11  31000  00524  X  89.  ,     , TP  ,Q      ,T5    ,                       $
00132  00132  11  00442  00526  X  90.  , S6  , TP  ,C20    ,T7    ,GENERATE NEW CATALOGUE  $
00133  00133  11  00445  00134  X  91.  ,     , TP  ,C23    ,L+1   ,                       $
00134  00134  11  30000  30000  X  92.  ,     , TP  ,FILL   ,FILL  ,                       $
00135  00135  23  00134  00446  X  93.  ,     , RS  ,L-1    ,C24   ,                       $
00136  00136  41  00526  00134  X  94.  ,     , IJ  ,T7     ,L-2   ,                       $
00137  00137  37  00137  00140  X  95.  , S7  , RJ  ,L      ,L+1   ,                       $
00140  00140  11  00447  00527  X  96.  ,     , TP  ,C25    ,T8    ,                       $
00141  00141  11  00450  00530  X  97.  ,     , TP  ,C26    ,T9    ,                       $
00142  00142  11  00440  00531  X  98.  ,     , TP  ,C17    ,T10   ,                       $
00143  00143  11  00441  00526  X  99.  ,     , TP  ,C18    ,T7    ,                       $
00144  00144  11  00504  00514  X 100.  ,     , TP  ,120V   ,ADDCWD,                       $
00145  00145  11  00451  00150  X 101.  , S8  , TP  ,C27    ,S9    ,                       $
00146  00146  15  00452  00151  X 102.  ,     , TU  ,C28    ,L+3   ,                       $
00147  00147  16  00452  00154  X 103.  ,     , TV  ,C28    ,L+5   ,                       $
00150  00150  11  30000  30000  X 104.  , S9  , TP  ,FILL   ,FILL  ,                       $
00151  00151  55  30000  32000  X 105.  ,     , LQ  ,FILL   ,A     ,                       $
```

```
00152  00152  43  00421  00177  X  106.  ,      , EJ  ,C2      ,S12     ,                    $
00153  00153  51  00530  32000  X  107.  ,      , QT  ,T9      ,A       ,                    $
00154  00154  35  00526  30000  X  108.  ,      , AT  ,T7      ,FILL    ,                    $
00155  00155  51  00531  00514  X  109.  ,      , QT  ,T10     ,ADDCWD  ,                    $
00156  00156  43  00522  00166  X  110.  ,      , EJ  ,T3      ,S11     ,                    $
00157  00157  43  00521  00164  X  111.  ,      , EJ  ,T2      ,S10+4   ,                    $
00160  00160  21  00526  00527  X  112.  , S10  , RA  ,T7      ,T8      ,                    $
00161  00161  21  00150  00474  X  113.  ,      , RA  ,S9      ,2V      ,                    $
00162  00162  21  00154  00474  X  114.  ,      , RA  ,S9+4    ,2V      ,                    $
00163  00163  37  00163  00164  X  115.  ,      , RJ  ,L       ,L+1     ,                    $
00164  00164  75  20002  00150  X  116.  ,      , RPU ,2       ,S9      ,                    $
00165  00165  21  00150  00475  X  117.  ,      , RA  ,S9      ,2U      ,                    $
00166  00166  37  00166  00167  X  118.  , S11  , RJ  ,L       ,L+1     ,                    $
00167  00167  16  00150  00171  X  119.  ,      , TV  ,S9      ,L+2     ,                    $
00170  00170  16  00154  00173  X  120.  ,      , TV  ,S9+4    ,L+3     ,                    $
00171  00171  11  00524  30000  X  121.  ,      , TP  ,T5      ,FILL    ,                    $
00172  00172  31  00526  00000  X  122.  ,      , SP  ,T7      ,        ,                    $
00173  00173  35  00525  30000  X  123.  ,      , AT  ,T6      ,FILL    ,                    $
00174  00174  16  00164  00163  X  124.  ,      , TV  ,S10+4   ,S10+3   ,                    $
00175  00175  16  00176  00166  X  125.  ,      , TV  ,L+1     ,S11     ,                    $
00176  00176  45  00000  00160  X  126.  ,      , MJ  ,        ,S10     ,                    $
00177  00177  21  00514  00527  X  127.  , S12  , RA  ,ADDCWD  ,T8      ,                    $
00200  00200  43  00522  00167  X  128.  ,      , EJ  ,T3      ,S11+1   ,                    $
00201  00201  16  00150  00203  X  129.  ,      , TV  ,S9      ,L+2     ,                    $
00202  00202  75  10004  00204  X  130.  ,      , RPV ,4       ,L+2     ,                    $
00203  00203  11  00421  30000  X  131.  ,      , TP  ,C2      ,FILL    ,                    $
00204  00204  37  00166  00166  X  132.  ,      , RJ  ,S11     ,S11     ,                    $
00205  00205  37  00205  00206  X  133.  , S13  , RJ  ,L       ,L+1     ,                    $
00206  00206  31  00523  00000  X  134.  ,      , SP  ,T4      ,        ,WRITE NEW CATALOG   $
00207  00207  73  00504  00532  X  135.  ,      , DV  ,120V    ,T11     ,                    $
00210  00210  47  00211  00212  X  136.  ,      , ZJ  ,L+1     ,L+2     ,                    $
00211  00211  21  00532  00473  X  137.  ,      , RA  ,T11     ,1V      ,                    $
00212  00212  31  00532  00000  X  138.  ,      , SP  ,T11     ,        ,                    $
00213  00213  37  00543  00535  X  139.  ,      , RJ  ,MTWREX  ,MTWRE2  ,                    $
00214  00214  31  00473  00000  X  140.  ,      , SP  ,1V      ,0       ,READ OP FILE LABEL  $
00215  00215  37  00565  00555  X  141.  ,      , RJ  ,MTRDEX  ,MTREAD  ,                    $
00216  00216  11  00523  01320  X  142.  ,      , TP  ,T4      ,BF+2    ,SET REVISED WORD COUNT $
00217  00217  31  00473  00000  X  143.  ,      , SP  ,1V      ,0       ,WRITE OP FILE LABEL $
```

```
00220  00220  37  00543  00535  X  144.  ,      ,  RJ  ,MTWREX  ,MTWRE2  ,                    $
00221  00221  37  00115  00111  X  145.  ,      ,  RJ  ,S5      ,S4      ,READ OP FILE         $
00222  00222  55  00007  31025  X  146.  ,      ,  LQ  ,PAR1    ,Q+21    ,GENERATE OP FILE ITEM $
00223  00223  51  00453  00524  X  147.  ,      ,  QT  ,C30     ,T5      ,                     $
00224  00224  55  00010  31022  X  148.  ,      ,  LQ  ,PAR2    ,Q+18    ,                     $
00225  00225  52  00464  00524  X  149.  ,      ,  QA  ,K7      ,T5      ,                     $
00226  00226  11  00435  31000  X  150.  ,      ,  TP  ,C14     ,Q       ,                     $
00227  00227  42  00506  00231  X  151.  ,      ,  TJ  ,513V    ,L+2     ,                     $
00230  00230  45  00000  00056  X  152.  ,      ,  MJ  ,        ,ALARM1-4,                     $
00231  00231  55  00525  00017  X  153.  ,      ,  LQ  ,T6      ,15      ,                     $
00232  00232  45  00000  00233  X  154.  ,      ,  MJ  ,        ,S13+22  ,                     $
00233  00233  55  00527  00017  X  155.  ,      ,  LQ  ,T8      ,15      ,GENERATE NEW OP FILE $
00234  00234  55  00530  00017  X  156.  ,      ,  LQ  ,T9      ,15      ,                     $
00235  00235  55  00531  00017  X  157.  ,      ,  LQ  ,T10     ,15      ,                     $
00236  00236  55  00521  00017  X  158.  ,      ,  LQ  ,T2      ,15      ,                     $
00237  00237  55  00522  00017  X  159.  ,      ,  LQ  ,T3      ,15      ,                     $
00240  00240  11  00444  00526  X  160.  , S14  ,  TP  ,C22     ,T7      ,                     $
00241  00241  11  00454  00245  X  161.  ,      ,  TP  ,C32     ,L+4     ,                     $
00242  00242  15  00455  00244  X  162.  ,      ,  TU  ,C33     ,L+2     ,                     $
00243  00243  16  00452  00246  X  163.  ,      ,  TV  ,C28     ,L+3     ,                     $
00244  00244  11  30000  31000  X  164.  ,      ,  TP  ,FILL    ,Q       ,                     $
00245  00245  11  30000  30000  X  165.  ,      ,  TP  ,FILL    ,FILL    ,                     $
00246  00246  11  31000  30000  X  166.  ,      ,  TP  ,Q       ,FILL    ,                     $
00247  00247  21  00244  00475  X  167.  ,      ,  RA  ,L-3     ,2U      ,                     $
00250  00250  21  00245  00472  X  168.  ,      ,  RA  ,L-3     ,K13     ,                     $
00251  00251  21  00246  00474  X  169.  ,      ,  RA  ,L-3     ,2V      ,                     $
00252  00252  41  00526  00244  X  170.  ,      ,  IJ  ,T7      ,L-6     ,                     $
00253  00253  37  00253  00254  X  171.  , S15  ,  RJ  ,L       ,L+1     ,                     $
00254  00254  37  00137  00132  X  172.  ,      ,  RJ  ,S7      ,S6      ,                     $
00255  00255  11  00465  00526  X  173.  ,      ,  TP  ,K8      ,T7      ,                     $
00256  00256  16  00474  00526  X  174.  ,      ,  TV  ,2V      ,T7      ,                     $
00257  00257  11  00466  00514  X  175.  ,      ,  TP  ,K9      ,ADDCWD  ,                     $
00260  00260  37  00205  00145  X  176.  ,      ,  RJ  ,S13     ,S8      ,                     $
00261  00261  37  00253  00240  X  177.  ,      ,  RJ  ,S15     ,S14     ,                     $
00262  00262  31  00532  00000  X  178.  ,      ,  SP  ,T11     ,        ,WRITE NEW OP FILE    $
00263  00263  37  00543  00535  X  179.  ,      ,  RJ  ,MTWREX  ,MTWRE2  ,                     $
00264  00264  31  00473  00000  X  180.  ,      ,  SP  ,1V      ,0       ,READ SUBROUTINE LABEL $
00265  00265  37  00565  00555  X  181.  ,      ,  RJ  ,MTRDEX  ,MTREAD  ,                     $
```

```
00266  00266  31  00473  00000  X  182.  ,     ,  SP  ,1V       ,0        ,WRITE SUBROUTINE LABEL  $
00267  00267  37  00543  00535  X  183.  ,     ,  RJ  ,MTWREX   ,MTWRE2   ,                        $
00270  00270  11  00465  00507  X  184.  ,     ,  TP  ,K8       ,ICWDC    ,PRESET INPUT AND OUTPUT $
00271  00271  11  00465  00510  X  185.  ,     ,  TP  ,K8       ,OCWDC    ,  CWD COUNTERS
00272  00272  11  00007  31000  X  186.  ,     ,  TP  ,PAR1     ,Q        ,EXTRACT INFO FROM       $
                                                                            PARAMETERS
00273  00273  51  00457  00511  X  187.  ,     ,  QT  ,K2       ,DELCWD

00274  00274  51  00460  00512  X  188.  ,     ,  QT  ,K3       ,INPUTS   ,                        $
00275  00275  51  00462  00513  X  189.  ,     ,  QT  ,K5       ,NOWDS    ,                        $
00276  00276  11  00010  31000  X  190.  ,     ,  TP  ,PAR2     ,Q        ,                        $
00277  00277  51  00457  00514  X  191.  ,     ,  QT  ,K2       ,ADDCWD   ,                        $
00300  00300  51  00463  00515  X  192.  ,     ,  QT  ,K6       ,QNP      ,                        $
00301  00301  51  00464  00516  X  193.  ,     ,  QT  ,K7       ,P        ,                        $
00302  00302  55  00511  00033  X  194.  ,     ,  LQ  ,DELCWD   ,27       ,                        $
00303  00303  55  00514  00033  X  195.  ,     ,  LQ  ,ADDCWD   ,27       ,                        $
00304  00304  21  00511  00466  X  196.  ,     ,  RA  ,DELCWD   ,K9       ,                        $
00305  00305  21  00514  00466  X  197.  ,     ,  RA  ,ADDCWD   ,K9       ,                        $
00306  00306  37  01222  01141  X  198.  ,     ,  RJ  ,HSPEX    ,HSPE1    ,                        $
00307  00307  31  00507  00000  X  199.  , B1  ,  SP  ,ICWDC    ,0        ,                        $
00310  00310  43  00514  00312  X  200.  ,     ,  EJ  ,ADDCWD   ,L+2      ,IS ROUTINE TO BE ADDED  $
00311  00311  45  00000  00354  X  201.  ,     ,  MJ  ,         ,B5       ,  NO                    $
00312  00312  31  00513  00025  X  202.  ,     ,  SP  ,NOWDS    ,21       ,    YES-SET UP PRELUDE  $
00313  00313  22  00000  01317  X  203.  ,     ,  LTL ,         ,BF+1     ,                        $
00314  00314  31  00512  00003  X  204.  ,     ,  SP  ,INPUTS   ,3        ,                        $
00315  00315  32  00510  00000  X  205.  ,     ,  SA  ,OCWDC    ,0        ,                        $
00316  00316  32  01317  00000  X  206.  ,     ,  SA  ,BF+1     ,0        ,                        $
00317  00317  35  00461  01316  X  207.  ,     ,  AT  ,K4       ,BF       ,                        $
00320  00320  55  00512  00030  X  208.  ,     ,  LQ  ,INPUTS   ,24       ,                        $
00321  00321  11  31000  01321  X  209.  ,     ,  TP  ,Q        ,BF+3     ,                        $
00322  00322  23  01317  00516  X  210.  ,     ,  RS  ,BF+1     ,P        ,                        $
00323  00323  11  00515  01320  X  211.  ,     ,  TP  ,QNP      ,BF+2     ,                        $
00324  00324  11  00473  01322  X  212.  ,     ,  TP  ,1V       ,BF+4     ,                        $
00325  00325  11  00011  01323  X  213.  ,     ,  TP  ,PAR3     ,BF+5     ,                        $
00326  00326  11  00477  01324  X  214.  ,     ,  TP  ,5V       ,BF+6     ,                        $
00327  00327  55  01323  00006  X  215.  , B2  ,  LQ  ,BF+5     ,6        ,                        $
00330  00330  51  00420  32000  X  216.  ,     ,  QT  ,C1       ,A        ,                        $
00331  00331  47  00333  00332  X  217.  ,     ,  ZJ  ,L+2      ,L+1      ,                        $
```

```
00332  00332  21  31000  00473  X  218.  ,      , RA  ,Q       ,1V      ,                          $
00333  00333  41  01324  00327  X  219.  ,      , IJ  ,BF+6    ,B2      ,                          $
00334  00334  21  00510  00456  X  220.  ,      , RA  ,OCWDC   ,K1      ,                          $
00335  00335  11  00462  31000  X  221.  ,      , TP  ,K5      ,Q       ,                          $
00336  00336  53  00513  00341  X  222.  ,      , QS  ,NOWDS   ,B3      ,                          $
00337  00337  53  00513  00344  X  223.  ,      , QS  ,NOWDS   ,B4      ,                          $
00340  00340  21  00341  00505  X  224.  ,      , RA  ,B3      ,120U    ,                          $
00341  00341  75  10000  00343  X  225.  , B3   , RPV ,0       ,L+2     ,FILL BUFFER WITH Z CODES$
00342  00342  11  00421  01324  X  226.  ,      , TP  ,C2      ,BF+6    ,                          $
00343  00343  37  01222  01156  X  227.  ,      , RJ  ,HSPEX   ,HSPE2   ,                          $
00344  00344  75  30000  00346  X  228.  , B4   , RPB ,        ,L+2     ,                          $
00345  00345  11  60000  01324  X  229.  ,      , TP  ,RTN     ,BF+6    ,                          $
00346  00346  21  00513  00501  X  230.  ,      , RA  ,NOWDS   ,6U      ,                          $
00347  00347  73  00505  31000  X  231.  ,      , DV  ,120U    ,Q       ,                          $
00350  00350  47  00351  00352  X  232.  ,      , ZJ  ,L+1     ,L+2     ,                          $
00351  00351  21  31000  00473  X  233.  ,      , RA  ,Q       ,1V      ,                          $
00352  00352  31  31000  00000  X  234.  ,      , SP  ,Q       ,0       ,                          $
00353  00353  37  00543  00535  X  235.  ,      , RJ  ,MTWREX  ,MTWRE2  ,WRITE ROUTINE ON OUTPUT $
                                                                          TAPE
00354  00354  31  00473  00000  X  236.  , B5   , SP  ,1V      ,0       ,                          $
00355  00355  37  00565  00555  X  237.  ,      , RJ  ,MTRDEX  ,MTREAD  ,READ 1 BLOCK FROM        $
                                                                          INPUT TAPE
00356  00356  31  01316  00000  X  238.  ,      , SP  ,BF      ,0       ,                          $
00357  00357  43  00431  00407  X  239.  ,      , EJ  ,C10     ,B7      ,IS THIS END OF ENTRY     $
                                                                        , BLOCK
00360  00360  11  00467  31000  X  240.  ,      , TP  ,K10     ,Q       ,NO                        $
00361  00361  51  01316  32000  X  241.  ,      , QT  ,BF      ,A       ,                          $
00362  00362  73  00504  00517  X  242.  ,      , DV  ,120V    ,TEMP1   ,                          $
00363  00363  47  00364  00365  X  243.  ,      , ZJ  ,L+1     ,L+2     ,                          $
00364  00364  21  00517  00473  X  244.  ,      , RA  ,TEMP1   ,1V      ,                          $
00365  00365  31  00507  00000  X  245.  ,      , SP  ,ICWDC   ,0       ,                          $
00366  00366  43  00511  00401  X  246.  ,      , EJ  ,DELCWD  ,B6      ,IS ROUTINE TO BE         $
                                                                          DELETED
00367  00367  21  00507  00456  X  247.  ,      , RA  ,ICWDC   ,K1      ,NO                        $
00370  00370  11  00470  31000  X  248.  ,      , TP  ,K11     ,Q       ,                          $
00371  00371  53  00510  01316  X  249.  ,      , QS  ,OCWDC   ,BF      ,                          $
00372  00372  21  00510  00456  X  250.  ,      , RA  ,OCWDC   ,K1      ,                          $
00373  00373  37  01222  01156  X  251.  ,      , RJ  ,HSPEX   ,HSPE2   ,                          $
00374  00374  23  00517  00473  X  252.  ,      , RS  ,TEMP1   ,1V      ,                          $
```

```
00375  00375  37  00565  00556  X  253.  ,      , RJ  ,MTRDEX  ,MTREAD+1  ,READ REST OF ROUTINE      $
00376  00376  21  00517  00473  X  254.  ,      , RA  ,TEMP1   ,1V        ,                          $
00377  00377  37  00543  00535  X  255.  ,      , RJ  ,MTWREX  ,MTWRE2    ,WRITE ROUTINE TO OUTPUT
                                                                            TAPE                     $
00400  00400  45  00000  00307  X  256.  ,      , MJ  ,        ,B1        ,                          $
00401  00401  23  00517  00473  X  257.  , B6   , RS  ,TEMP1   ,1V        ,                          $
00402  00402  11  00471  31000  X  258.  ,      , TP  ,K12     ,Q         ,                          $
00403  00403  53  00517  00672  X  259.  ,      , QS  ,TEMP1   ,MVFWIN    ,                          $
00404  00404  17  00000  00672  X  260.  ,      , EF  ,        ,MVFWIN    ,MOVE PAST ROUTINE TO      $
                                                                            BE DELETED
00405  00405  21  00507  00456  X  261.  ,      , RA  ,ICWDC   ,K1        ,                          $
00406  00406  45  00000  00307  X  262.  ,      , MJ  ,        ,B1        ,                          $
00407  00407  11  00432  01317  X  263.  , B7   , TP  ,C11     ,BF+1      ,WRITE END OF ENTRY        $
                                                                            BLOCKS TO OUTPUT TAPE
00410  00410  75  10546  00412  X  264.  ,      , RPV ,358     ,L+2       ,                          $
00411  00411  11  00421  01320  X  265.  ,      , TP  ,C2      ,BF+2      ,                          $
00412  00412  31  00476  00000  X  266.  ,      , SP  ,3V      ,0         ,                          $
00413  00413  37  00543  00535  X  267.  ,      , RJ  ,MTWREX  ,MTWRE2    ,                          $
00414  00414  17  00000  00670  X  268.  ,      , EF  ,        ,MTINRW    ,                          $
00415  00415  37  01222  01250  X  269.  ,      , RJ  ,HSPEX   ,HSPE3     ,                          $
00416  00416  37  00635  00622  X  270.  ,      , RJ  ,MTCHEX  ,MTCHEK    ,CHECK OUTPUT TAPE         $
00417  00417  45  00000  00006  X  271.  ,      , MJ  ,        ,EXIT      ,                          $
00420  00420  00  00000  00077  X  272.  , C1   , B   ,        ,77        ,CONSTANTS                 $
00421  00421  74  74747  47474  X  273.  , C2   , B74 ,74747   ,47474     ,                          $
00422  00422  01  01463  42501  X  274.  , C3   , B01 ,01463   ,42501     ,                          $
00423  00423  01  66245  23001  X  275.  , C4   , B01 ,66245   ,23001     ,                          $
00424  00424  46  34250  10101  X  276.  , C5   , B46 ,34250   ,10101     ,                          $
00425  00425  26  24660  10101  X  277.  , C6   , B26 ,24660   ,10101     ,                          $
00426  00426  01  01015  15201  X  278.  , C7   , B01 ,01015   ,15201     ,                          $
00427  00427  31  34463  00104  X  279.  , C8   , B31 ,34463   ,00104     ,                          $
00430  00430  65  67255  46650  X  280.  , C9   , B65 ,67255   ,46650     ,                          $
00431  00431  30  50270  15131  X  281.  , C10  , B30 ,50270   ,15131     ,                          $
00432  00432  01  30506  65473  X  282.  , C11  , B01 ,30506   ,65473     ,                          $
00433  00433  00  00001  70000  X  283.  , C12  , B   ,1       ,70000     ,                          $
00434  00434  00  00000  40000  X  284.  , C13  , B   ,        ,40000     ,                          $
00435  00435  31  03122  22404  X  285.  , C14  , B31 ,03122   ,22404     ,                          $
00436  00436  06  03041  11423  X  286.  , C15  , B06 ,03041   ,11423     ,                          $
00437  00437  31  26343  03762  X  287.  ,C16   , B31 ,26343   ,03762     ,                          $
```

```
00440  00440  00  00000  07770  X  288.  , C17  , B    ,          ,7770      ,  $
00441  00441  00  00000  00200  X  289.  , C18  , B    ,          ,200       ,  $
00442  00442  00  00000  01775  X  290.  , C20  ,      ,          ,1021      ,  $
00443  00443  00  00000  50000  X  291.  , C21  , B    ,          ,50000     ,  $
00444  00444  00  00000  00776  X  292.  , C22  ,      ,          ,510       ,  $
00445  00445  11  03313  03315  X  293.  , C23  , TP   ,BF+1021   ,BF+1023   ,  $
00446  00446  00  00001  00001  X  294.  , C24  ,      ,1         ,1         ,  $
00447  00447  00  00000  00010  X  295.  , C25  , B    ,          ,10        ,  $
00450  00450  00  00000  50007  X  296.  , C26  , B    ,          ,50007     ,  $
00451  00451  11  01320  01316  X  297.  , C27  , TP   ,BF+2      ,BF        ,  $
00452  00452  00  01321  01317  X  298.  , C28  ,      ,BF+3      ,BF+1      ,  $
00453  00453  00  00000  07777  X  299.  , C30  , B    ,          ,7777      ,  $
00454  00454  11  01317  01316  X  300.  , C32  , TP   ,BF+1      ,BF        ,  $
00455  00455  00  01316  00000  X  301.  , C33  ,      ,BF        ,          ,  $
00456  00456  00  00010  00000  X  302.  , K1   , B00  ,00010     ,00000     ,  $
00457  00457  77  70000  00000  X  303.  , K2   , B77  ,70000     ,00000     ,  $
00460  00460  00  00000  70000  X  304.  , K3   , B00  ,00000     ,70000     ,  $
00461  00461  00  50000  00006  X  305.  , K4   , B00  ,50000     ,00006     ,  $
00462  00462  00  07777  00000  X  306.  , K5   , B    ,7777      ,00000     ,  $
00463  00463  00  07777  77777  X  307.  , K6   , B    ,7777      ,77777     ,  $
00464  00464  00  00000  00777  X  308.  , K7   , B    ,          ,777       ,  $
00465  00465  00  00200  00000  X  309.  , K8   , B    ,200       ,00000     ,  $
00466  00466  00  00170  00000  X  310.  , K9   , B    ,170       ,00000     ,  $
00467  00467  00  00000  77777  X  311.  , K10  , B    ,          ,77777     ,  $
00470  00470  00  07770  00000  X  312.  , K11  , B    ,7770      ,00000     ,  $
00471  00471  00  00000  07777  X  313.  , K12  , B    ,          ,7777      ,  $
00472  00472  00  00002  00002  X  314.  , K13  ,      ,2         ,2         ,  $
00473  00473  00  00000  00001  X  315.  , 1V   ,      ,          ,1         ,  $
00474  00474  00  00000  00002  X  316.  , 2V   ,      ,          ,2         ,  $
00475  00475  00  00002  00000  X  317.  , 2U   ,      ,2         ,          ,  $
00476  00476  00  00000  00003  X  318.  , 3V   ,      ,          ,3         ,  $
00477  00477  00  00000  00005  X  319.  , 5V   ,      ,          ,5         ,  $
00500  00500  00  00000  00006  X  320.  , 6V   ,      ,          ,6         ,  $
00501  00501  00  00006  00000  X  321.  , 6U   ,      ,6         ,          ,  $
00502  00502  00  00000  00007  X  322.  , 7V   ,      ,          ,7         ,  $
00503  00503  00  00000  00017  X  323.  , 15V  ,      ,          ,15        ,  $
00504  00504  00  00000  00170  X  324.  , 120V ,      ,          ,120       ,  $
00505  00505  00  00170  00000  X  325.  , 120U ,      ,120       ,          ,  $
```

```
00506  00506  00  00000  01001  X  326.  , 513V   ,      ,      ,513     ,                      $
00507  00507  00  00000  00000  X  327.  , ICWDC  ,      ,      ,0       ,TEMPORARIES           $
00510  00510  00  00000  00000  X  328.  , OCWDC  ,      ,      ,0       ,                      $
00511  00511  00  00000  00000  X  329.  , DELCWD ,      ,      ,0       ,                      $
00512  00512  00  00000  00000  X  330.  , INPUTS ,      ,      ,0       ,                      $
00513  00513  00  00000  00000  X  331.  , NOWDS  ,      ,      ,0       ,                      $
00514  00514  00  00000  00000  X  332.  , ADDCWD ,      ,      ,0       ,                      $
00515  00515  00  00000  00000  X  333.  , QNP    ,      ,      ,0       ,                      $
00516  00516  00  00000  00000  X  334.  , P      ,      ,      ,0       ,                      $
00517  00517  00  00000  00000  X  335.  , TEMP1  ,      ,      ,0       ,                      $
00520  00520  00  00000  00000  X  336.  , T1     ,      ,      ,        ,                      $
00521  00521  00  00000  00000  X  337.  , T2     ,      ,      ,        ,                      $
00522  00522  00  00000  00000  X  338.  , T3     ,      ,      ,        ,                      $
00523  00523  00  00000  00000  X  339.  , T4     ,      ,      ,        ,                      $
00524  00524  00  00000  00000  X  340.  , T5     ,      ,      ,        ,                      $
00525  00525  00  00000  00000  X  341.  , T6     ,      ,      ,        ,                      $
00526  00526  00  00000  00000  X  342.  , T7     ,      ,      ,        ,                      $
00527  00527  00  00000  00000  X  343.  , T8     ,      ,      ,        ,                      $
00530  00530  00  00000  00000  X  344.  , T9     ,      ,      ,        ,                      $
00531  00531  00  00000  00000  X  345.  , T10    ,      ,      ,        ,                      $
00532  00532  00  00000  00000  X  346.  , T11    ,      ,      ,        ,                      $
00533  00533  11  00674  00712  X  347.  , MTWRE1 , TP     ,MTZERO  ,MTBLKS  ,ENTRY 1 WRITE MAG TAPE $
00534  00534  11  00674  00713  X  348.  ,        , TP     ,MTZERO  ,MTSUM   ,                      $
00535  00535  16  00673  00554  X  349.  , MTWRE2 , TV     ,MTSET   ,MTEW    ,ENTRY 2               $
00536  00536  15  00673  00551  X  350.  ,        , TU     ,MTSET   ,MTWR3   ,                      $
00537  00537  17  00000  00663  X  351.  ,        , EF     ,        ,MTWRIT  ,                      $
00540  00540  11  32000  00714  X  352.  ,        , TP     ,A       ,MTIND1  ,                      $
00541  00541  41  00714  00544  X  353.  , MTWR1  , IJ     ,MTIND1  ,MTWR2   ,                      $
00542  00542  17  00000  00657  X  354.  ,        , EF     ,        ,MTSTOP  ,                      $
00543  00543  45  00000  30000  X  355.  , MTWREX , MJ     ,        ,FILL    ,EXIT FOR WRITE MAG TAPE $
00544  00544  21  00554  00701  X  356.  , MTWR2  , RA     ,MTEW    ,MT120V  ,                      $
00545  00545  21  00551  00702  X  357.  ,        , RA     ,MTWR3   ,MT120U  ,                      $
00546  00546  21  00712  00675  X  358.  ,        , RA     ,MTBLKS  ,MT1V    ,                      $
00547  00547  31  00713  00000  X  359.  ,        , SP     ,MTSUM   ,0       ,                      $
00550  00550  75  20170  00552  X  360.  ,        , RPU    ,120     ,L+2     ,                      $
00551  00551  32  30000  00000  X  361.  , MTWR3  , SA     ,FILL    ,0       ,                      $
00552  00552  11  32000  00713  X  362.  ,        , TP     ,A       ,MTSUM   ,                      $
00553  00553  75  10170  00541  X  363.  ,        , RPV    ,120     ,MTWR1   ,                      $
```

```
00554  00554  77  10000  30000  X  364.  , MTEW   , EWB  ,          ,FILL     ,                    $
00555  00555  16  00673  00570  X  365.  , MTREAD, TV   ,MTSET     ,MTER     ,ENTRY TO READ MT    $
00556  00556  11  32000  00714  X  366.  ,        , TP   ,A         ,MTIND1   ,                     $
00557  00557  47  00560  00565  X  367.  ,        , ZJ   ,L+1       ,MTRDEX   ,                     $
00560  00560  17  00000  00667  X  368.  ,        , EF   ,          ,MTRDFW   ,                     $
00561  00561  11  00677  00715  X  369.  , MTRD1  , TP   ,MT5V      ,MTIND2   ,                     $
00562  00562  41  00714  00566  X  370.  ,        , IJ   ,MTIND1    ,MTRD2    ,                     $
00563  00563  17  00000  00657  X  371.  ,        , EF   ,          ,MTSTOP   ,                     $
00564  00564  17  00000  00660  X  372.  ,        , EF   ,          ,MTNOBI   ,                     $
00565  00565  45  00000  30000  X  373.  , MTRDEX, MJ   ,          ,FILL     ,EXIT FOR READ MT    $
00566  00566  21  00570  00701  X  374.  , MTRD2  , RA   ,MTER      ,MT120V   ,                     $
00567  00567  75  10170  00571  X  375.  ,        , RPV  ,120       ,L+2      ,                     $
00570  00570  76  10000  30000  X  376.  , MTER   , ERB  ,          ,FILL     ,                     $
00571  00571  76  00000  32000  X  377.  ,        , ERA  ,          ,A        ,                     $
00572  00572  47  00573  00561  X  378.  ,        , ZJ   ,L+1       ,MTRD1    ,                     $
00573  00573  41  00715  00605  X  379.  ,        , IJ   ,MTIND2    ,MTRD4    ,                     $
00574  00574  15  00710  00603  X  380.  ,        , TU   ,MTK6      ,MTRD3+3  ,                     $
00575  00575  11  00703  31000  X  381.  ,        , TP   ,MTK1      ,Q        ,                     $
00576  00576  31  00704  00044  X  382.  ,        , SP   ,MTK2      ,36       ,                     $
00577  00577  27  32000  00705  X  383.  ,        , CC   ,A         ,MTK3     ,                     $
00600  00600  54  32000  00006  X  384.  , MTRD3  , LA   ,A         ,6        ,                     $
00601  00601  61  00000  32000  X  385.  ,        , PR   ,          ,A        ,                     $
00602  00602  44  00603  00600  X  386.  ,        , QJ   ,L+1       ,MTRD3    ,                     $
00603  00603  31  30000  00044  X  387.  ,        , SP   ,FILL      ,36       ,                     $
00604  00604  44  00062  00600  X  388.  ,        , QJ   ,ALARM1    ,MTRD3    ,                     $
00605  00605  17  00000  00671  X  389.  , MTRD4  , EF   ,          ,MTMVBA   ,                     $
00606  00606  45  00000  00607  X  390.  ,        , MJ   ,          ,L+1      ,                     $
00607  00607  17  00000  00661  X  391.  ,        , EF   ,          ,MTHIBI   ,                     $
00610  00610  21  00606  00676  X  392.  ,        , RA   ,MTRD4+1   ,MT3V     ,                     $
00611  00611  45  00000  00617  X  393.  ,        , MJ   ,          ,MTRD5    ,                     $
00612  00612  17  00000  00662  X  394.  ,        , EF   ,          ,MTLOBI   ,                     $
00613  00613  21  00606  00676  X  395.  ,        , RA   ,MTRD4+1   ,MT3V     ,                     $
00614  00614  45  00000  00617  X  396.  ,        , MJ   ,          ,MTRD5    ,                     $
00615  00615  17  00000  00660  X  397.  ,        , EF   ,          ,MTNOBI   ,                     $
00616  00616  23  00606  00700  X  398.  ,        , RS   ,MTRD4+1   ,MT6V     ,                     $
00617  00617  37  00617  00620  X  399.  , MTRD5  , RJ   ,L         ,L+1      ,                     $
00620  00620  17  00000  00667  X  400.  ,        , EF   ,          ,MTRDFW   ,                     $
00621  00621  45  00000  00567  X  401.  ,        , MJ   ,          ,MTRD2+1  ,                     $
```

```
00622  00622  75  10170  00624  X  402.  , MTCHEK, RPV      ,120      ,L+2      ,ENTRY TO CHECK MAG TAPE $
00623  00623  11  00674  01316  X  403.  ,       , TP       ,MTZERO   ,BF       ,                        $
00624  00624  17  00000  00664  X  404.  ,       , EF       ,         ,MTRDBW   ,                        $
00625  00625  11  00677  00715  X  405.  , MTCH2 , TP       ,MT5V     ,MTIND2   ,                        $
00626  00626  41  00712  00636  X  406.  ,       , IJ       ,MTBLKS   ,MTCHEX+1 ,                        $
00627  00627  17  00000  00657  X  407.  ,       , EF       ,         ,MTSTOP   ,                        $
00630  00630  17  00000  00665  X  408.  ,       , EF       ,         ,MTOURW   ,                        $
00631  00631  37  00643  00636  X  409.  ,       , RJ       ,MTCH3-1  ,MTCHEX+1 ,                        $
00632  00632  31  00713  00000  X  410.  ,       , SP       ,MTSUM    ,0        ,                        $
00633  00633  47  00651  00634  X  411.  ,       , ZJ       ,MTCH5-2  ,L+1      ,                        $
00634  00634  17  00000  00660  X  412.  ,       , EF       ,         ,MTNOBI   ,                        $
00635  00635  45  00000  30000  X  413.  , MTCHEX, MJ       ,         ,FILL     ,EXIT FOR CHECK MAG TAPE $
00636  00636  31  00457  00011  X  414.  ,       , SP       ,K2       ,9        ,                        $
00637  00637  32  00713  00000  X  415.  ,       , SA       ,MTSUM    ,0        ,                        $
00640  00640  75  20170  00642  X  416.  ,       , RPU      ,120      ,L+2      ,                        $
00641  00641  34  01316  00000  X  417.  ,       , SS       ,BF       ,0        ,                        $
00642  00642  11  32000  00713  X  418.  ,       , TP       ,A        ,MTSUM    ,                        $
00643  00643  37  00643  00644  X  419.  ,       , RJ       ,L        ,L+1      ,                        $
00644  00644  75  10170  00646  X  420.  , MTCH3 , RPV      ,120      ,L+2      ,                        $
00645  00645  76  10000  01316  X  421.  ,       , ERB      ,         ,BF       ,                        $
00646  00646  76  00000  32000  X  422.  ,       , ERA      ,         ,A        ,                        $
00647  00647  47  00650  00625  X  423.  ,       , ZJ       ,L+1      ,MTCH2    ,                        $
00650  00650  41  00715  00653  X  424.  ,       , IJ       ,MTIND2   ,MTCH5    ,                        $
00651  00651  15  00711  00603  X  425.  ,       , TU       ,MTK7     ,MTRD3+3  ,                        $
00652  00652  45  00000  00575  X  426.  ,       , MJ       ,         ,MTRD3-3  ,                        $
00653  00653  17  00000  00666  X  427.  , MTCH5 , EF       ,         ,MTMVFW   ,                        $
00654  00654  37  00617  00606  X  428.  ,       , RJ       ,MTRD5    ,MTRD4+1  ,                        $
00655  00655  17  00000  00664  X  429.  ,       , EF       ,         ,MTRDBW   ,                        $
00656  00656  45  00000  00644  X  430.  ,       , MJ       ,         ,MTCH3    ,                        $
00657  00657  02  00600  00000  X  431.  , MTSTOP, B02      ,00600    ,00000    ,                        $
00660  00660  02  00001  50000  X  432.  , MTNOBI, B02      ,00001    ,50000    ,                        $
00661  00661  02  00001  70000  X  433.  , MTHIBI, B02      ,00001    ,70000    ,                        $
00662  00662  02  00001  60000  X  434.  , MTLOBI, B02      ,00001    ,60000    ,                        $
00663  00663  02  00006  00000  X  435.  , MTWRIT, B02      ,00006    ,00000    ,                        $
00664  00664  02  00012  00000  X  436.  , MTRDBW, B02      ,00012    ,00000    ,                        $
00665  00665  02  00200  00000  X  437.  , MTOURW, B02      ,00200    ,00000    ,                        $
00666  00666  02  00004  00001  X  438.  , MTMVFW, B02      ,00004    ,00001    ,                        $
00667  00667  02  00002  00000  X  439.  , MTRDFW, B02      ,00002    ,00000                             $
```

```
00670  00670  02  00200  00000  X  440.  , MTINRW, B02      ,00200     ,00000     ,                        $
00671  00671  02  00014  00001  X  441.  , MTMVBA, B02      ,00014     ,00001     ,                        $
00672  00672  02  00004  00000  X  442.  , MVFWIN, B02      ,00004     ,00000     ,                        $
00673  00673  00  01126  01126  X  443.  , MTSET  ,          ,BF-120    ,BF-120    ,                        $
00674  00674  00  00000  00000  X  444.  , MTZERO,          ,          ,0         ,                        $
00675  00675  00  00000  00001  X  445.  , MT1V   ,          ,          ,1         ,                        $
00676  00676  00  00000  00003  X  446.  , MT3V   , B       ,          ,3         ,                        $
00677  00677  00  00000  00005  X  447.  , MT5V   ,          ,          ,5         ,                        $
00700  00700  00  00000  00006  X  448.  , MT6V   ,          ,          ,6         ,                        $
00701  00701  00  00000  00170  X  449.  , MT120V,          ,          ,120       ,                        $
00702  00702  00  00170  00000  X  450.  , MT120U,          ,120       ,          ,                        $
00703  00703  00  01006  00000  X  451.  , MTK1   , B00      ,01006     ,00000     ,                        $
00704  00704  45  47040  70104  X  452.  , MTK2   , B45      ,47040     ,70104     ,                        $
00705  00705  26  30141  12404  X  453.  , MTK3   , B26      ,30141     ,12404     ,                        $
00706  00706  12  20302  20404  X  454.  , MTK4   , B12      ,20302     ,20404     ,                        $
00707  00707  16  05201  63604  X  455.  , MTK5   , B16      ,05201     ,63604     ,                        $
00710  00710  00  00706  00000  X  456.  , MTK6   ,          ,MTK4      ,0         ,                        $
00711  00711  00  00707  00000  X  457.  , MTK7   ,          ,MTK5      ,0         ,                        $
00712  00712  00  00000  00000  X  458.  , MTBLKS,          ,          ,0         ,                        $
00713  00713  00  00000  00000  X  459.  , MTSUM  ,          ,          ,0         ,                        $
00714  00714  00  00000  00000  X  460.  , MTIND1,          ,          ,0         ,                        $
00715  00715  00  00000  00000  X  461.  , MTIND2,          ,          ,0         ,                        $
                                X  462.  , HSPBKT, RESERV   ,20        ,20        ,                        $
                                X  463.  , HSPBLK, RESERV   ,120       ,120       ,                        $
                                X  464.  , HSPTS , RESERV   ,6         ,6         ,                        $
01140  01140  00  00000  00000  X  465.  , HSPCTR,          ,          ,0         ,                        $
01141  01141  75  10214  01143  X  466.  , HSPE1  , RPV      ,140       ,L+2       ,ENTRY 1 HSP LISTING     $
01142  01142  11  01262  00716  X  467.  ,        , TP       ,HSPK1     ,HSPBKT    ,                        $
01143  01143  75  30005  01145  X  468.  ,        , RPB      ,5         ,L+2       ,                        $
01144  01144  11  01300  00742  X  469.  ,        , TP       ,HSPHD1    ,HSPBLK    ,                        $
01145  01145  75  30003  01147  X  470.  ,        , RPB      ,3         ,L+2       ,                        $
01146  01146  11  01305  01012  X  471.  ,        , TP       ,HSPHD2    ,HSPBLK+40 ,                        $
01147  01147  11  01310  01016  X  472.  ,        , TP       ,HSPHD2+3  ,HSPBLK+44 ,                        $
01150  01150  11  01311  01020  X  473.  ,        , TP       ,HSPHD2+4  ,HSPBLK+46 ,                        $
01151  01151  11  01312  01022  X  474   ,        , TP       ,HSPHD2+5  ,HSPBLK+48 ,                        $
01152  01152  11  01313  01024  X  475.  ,        , TP       ,HSPHD2+6  ,HSPBLK+50 ,                        $
01153  01153  11  01314  01026  X  476.  ,        , TP       ,HSPHD2+7  ,HSPBLK+52 ,                        $
01154  01154  11  01275  01140  X  477.  ,        , TP       ,HSP4V     ,HSPCTR    ,                        $
```

```
01155  01155  45  00000  01222  X  478.  ,        , MJ  ,        ,HSPEX     ,                        $
01156  01156  11  01316  31000  X  479.  , HSPE2  , TP  ,BF      ,Q         ,ENTRY 2 HSP LISTING     $
01157  01157  51  01264  01136  X  480.  ,        , QT  ,HSPK3   ,HSPTS+4   ,                        $
01160  01160  55  31000  00022  X  481.  ,        , LQ  ,Q       ,18        ,                        $
01161  01161  51  01263  32000  X  482.  ,        , QT  ,HSPK2   ,A         ,                        $
01162  01162  36  00503  32000  X  483.  ,        , ST  ,15V     ,A         ,                        $
01163  01163  37  01247  01231  X  484.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01164  01164  11  32000  00716  X  485.  ,        , TP  ,A       ,HSPBKT    ,                        $
01165  01165  11  01323  00720  X  486.  ,        , TP  ,BF+5    ,HSPBKT+2  ,                        $
01166  01166  23  01136  01276  X  487.  ,        , RS  ,HSPTS+4 ,HSP6V     ,                        $
01167  01167  37  01247  01231  X  488.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01170  01170  11  32000  00722  X  489.  ,        , TP  ,A       ,HSPBKT+4  ,                        $
01171  01171  31  01321  00000  X  490.  ,        , SP  ,BF+3    ,0         ,                        $
01172  01172  37  01247  01231  X  491.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01173  01173  11  32000  00724  X  492.  ,        , TP  ,A       ,HSPBKT+6  ,                        $
01174  01174  11  01320  31000  X  493.  ,        , TP  ,BF+2    ,Q         ,                        $
01175  01175  51  01263  01136  X  494.  ,        , QT  ,HSPK2   ,HSPTS+4   ,                        $
01176  01176  55  31000  00022  X  495.  ,        , LQ  ,Q       ,18        ,                        $
01177  01177  51  01263  01137  X  496.  ,        , QT  ,HSPK2   ,HSPTS+5   ,                        $
01200  01200  55  31000  00011  X  497.  ,        , LQ  ,Q       ,9         ,                        $
01201  01201  51  01263  32000  X  498.  ,        , QT  ,HSPK2   ,A         ,                        $
01202  01202  37  01247  01231  X  499.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01203  01203  11  32000  00730  X  500.  ,        , TP  ,A       ,HSPBKT+10 ,                        $
01204  01204  31  01136  00000  X  501.  ,        , SP  ,HSPTS+4 ,0         ,                        $
01205  01205  37  01247  01231  X  502.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01206  01206  11  32000  00732  X  503.  ,        , TP  ,A       ,HSPBKT+12 ,                        $
01207  01207  31  01137  00000  X  504.  ,        , SP  ,HSPTS+5 ,0         ,                        $
01210  01210  37  01247  01231  X  505.  ,        , RJ  ,HSP5    ,HSP3      ,                        $
01211  01211  11  32000  00726  X  506.  ,        , TP  ,A       ,HSPBKT+8  ,                        $
01212  01212  21  01140  01273  X  507.  , HSP1   , RA  ,HSPCTR  ,HSP1V     ,                        $
01213  01213  71  01140  01277  X  508.  ,        , MP  ,HSPCTR  ,HSP20V    ,                        $
01214  01214  35  01265  01216  X  509.  ,        , AT  ,HSPK4   ,L+2       ,                        $
01215  01215  75  30024  01217  X  510.  ,        , RPB ,20      ,L+2       ,                        $
01216  01216  00  00000  00000  X  511.  ,        , B   ,        ,0         ,                        $
01217  01217  31  01140  00000  X  512.  ,        , SP  ,HSPCTR  ,0         ,                        $
01220  01220  43  01276  01223  X  513.  ,        , EJ  ,HSP6V   ,HSP2A     ,                        $
01221  01221  37  01221  01222  X  514.  , HSP2   , RJ  ,L       ,L+1       ,                        $
01222  01222  45  00000  30000  X  515.  , HSPEX  , MJ  ,        ,FILL      ,EXIT FOR HSP LISTING    $
```

```
01223  01223  17  00000  01272  X  516.  , HSP2A , EF    ,        ,HSPWR    ,                        $
01224  01224  23  01140  32000  X  517.  ,       , RS    , HSPCTR ,A        ,                        $
01225  01225  75  10170  01227  X  518.  ,       , RPV   ,120     ,L+2      ,                        $
01226  01226  77  10000  00742  X  519.  ,       , EWB   ,        ,HSPBLK   ,                        $
01227  01227  75  10170  01221  X  520.  ,       , RPV   ,120     ,HSP2     ,                        $
01230  01230  11  01262  00742  X  521.  ,       , TP    ,HSPK1   ,HSPBLK   ,                        $
01231  01231  75  30003  01233  X  522.  , HSP3  , RPB   ,3       ,L+2      ,                        $
01232  01232  73  01266  01132  X  523.  ,       , DV    ,HSPK5   ,HSPTS    ,                        $
01233  01233  35  01274  01135  X  524.  ,       , AT    ,HSP2V   ,HSPTS+3  ,                        $
01234  01234  23  31000  32000  X  525.  ,       , RS    ,Q       ,A        ,                        $
01235  01235  42  01132  01241  X  526.  ,       , TJ    ,HSPTS   ,HSP4     ,                        $
01236  01236  42  01133  01242  X  527.  ,       , TJ    ,HSPTS+1 ,HSP4+1   ,                        $
01237  01237  42  01134  01243  X  528.  ,       , TJ    ,HSPTS+2 ,HSP4+2   ,                        $
01240  01240  45  00000  01244  X  529.  ,       , MJ    ,        ,HSP4+3   ,                        $
01241  01241  21  01132  01274  X  530.  , HSP4  , RA    ,HSPTS   ,HSP2V    ,                        $
01242  01242  21  01133  01274  X  531.  ,       , RA    ,HSPTS+1 ,HSP2V    ,                        $
01243  01243  21  01134  01274  X  532.  ,       , RA    ,HSPTS+2 ,HSP2V    ,                        $
01244  01244  31  01262  00052  X  533.  ,       , SP    ,HSPK1   ,42       ,                        $
01245  01245  75  20004  01247  X  534.  ,       , RPU   ,4       ,L+2      ,                        $
01246  01246  32  01132  00006  X  535.  ,       , SA    ,HSPTS   ,6        ,                        $
01247  01247  45  00000  30000  X  536.  , HSP5  , MJ    ,        ,FILL     ,                        $
01250  01250  75  10024  01252  X  537.  , HSPE3 , RPV   ,20      ,L+2      ,ENTRY 3 HSP LISTING     $
01251  01251  11  01262  00716  X  538.  ,       , TP    ,HSPK1   ,HSPBKT   ,                        $
01252  01252  37  01221  01212  X  539.  ,       , RJ    ,HSP2    ,HSP1     ,                        $
01253  01253  31  00712  00000  X  540.  ,       , SP    ,MTBLKS  ,0        ,                        $
01254  01254  37  01247  01231  X  541.  ,       , RJ    ,HSP5    ,HSP3     ,                        $
01255  01255  11  32000  00716  X  542.  ,       , TP    ,A       ,HSPBKT   ,                        $
01256  01256  11  01315  00717  X  543.  ,       , TP    ,HSPHD3  ,HSPBKT+1 ,                        $
01257  01257  37  01221  01212  X  544.  ,       , RJ    ,HSP2    ,HSP1     ,                        $
01260  01260  11  01271  01106  X  545.  ,       , TP    ,HSPK6   ,HSPBLK+100,                       $
01261  01261  45  00000  01223  X  546.  ,       , MJ    ,        ,HSP2A    ,                        $
01262  01262  01  01010  10101  X  547.  , HSPK1 , BO1   ,01010   ,10101    ,                        $
01263  01263  00  00000  00777  X  548.  , HSPK2 , B     ,        ,777      ,                        $
01264  01264  00  00000  77777  X  549.  , HSPK3 , B     ,        ,77777    ,                        $
01265  01265  11  00716  00716  X  550.  , HSPK4 , TP    ,HSPBKT  ,HSPBLK-20,                        $
01266  01266  00  00000  01750  X  551.  , HSPK5 , X     ,        ,1000     ,                        $
01267  01267  00  00000  00144  X  552.  ,       , X     ,        ,100      ,                        $
01270  01270  00  00000  00012  X  553.  ,       , X     ,        ,10       ,                        $
```

```
01271  01271  37  60010  10101  X  554.  , HSPK6  , B37      ,60010    ,10101    ,  $
01272  01272  02  00646  30000  X  555.  , HSPWR  , B02      ,00646    ,30000    ,  $
01273  01273  00  00000  00001  X  556.  , HSP1V  ,          ,         ,1        ,  $
01274  01274  00  00000  00002  X  557.  , HSP2V  ,          ,         ,2        ,  $
01275  01275  00  00000  00004  X  558.  , HSP4V  ,          ,         ,4        ,  $
01276  01276  00  00000  00006  X  559.  , HSP6V  ,          ,         ,6        ,  $
01277  01277  00  00000  00024  X  560.  , HSP20V ,          ,         ,20       ,  $
01300  01300  67  50342  65127  X  561.  , HSPHD1 , B67      ,50342    ,65127    ,  $
01301  01301  30  01656  72554  X  562.  ,         , B30      ,01656    ,72554    ,  $
01302  01302  51  67663  45030  X  563.  ,         , B51      ,67663    ,45030    ,  $
01303  01303  01  46342  55424  X  564.  ,         , B01      ,46342    ,55424    ,  $
01304  01304  54  73010  10137  X  565.  ,         , B54      ,73010    ,10137    ,  $
01305  01305  52  51653  46634  X  566.  , HSPHD2 , B52      ,51653    ,46634    ,  $
01306  01306  51  50010  10101  X  567.  ,         , B51      ,50010    ,10101    ,  $
01307  01307  01  50244  73001  X  568.  ,         , B01      ,50244    ,73001    ,  $
01310  01310  01  71515  42765  X  569.  ,         , B01      ,71515    ,42765    ,  $
01311  01311  34  50526  76665  X  570.  ,         , B34      ,50526    ,76665    ,  $
01312  01312  01  01010  15301  X  571.  ,         , B01      ,01010    ,15301    ,  $
01313  01313  01  01010  15001  X  572.  ,         , B01      ,01010    ,15001    ,  $
01314  01314  01  01010  15201  X  573.  ,         , B01      ,01010    ,15201    ,  $
01315  01315  25  46512  64565  X  574.  , HSPHD3 , B25      ,46512    ,64565    ,  $
                               X  575.  , BF     , EQUALS   ,HSPHD3+1 ,         ,  $
                               X  576.  ,         , ENDSUB   ,         ,         ,  $
```

B. Permanent Library Routines

The Permanent Library consists of routines required by UNICODE to provide its Input/Output facilities, and the general exponentiation system. Three sections of the Master Tape are used to contain the routines and related information in positions most advantageous to the compiling procedure.

The first such section is the Permanent Library Catalog containing the names of all routines and their associated UNICODE call words. This catalog is read from the Master Tape by the Dimension #2 Translator and is stored in the Combination List. It is then available throughout the Translation phase for recognition and proper handling of references to these routines.

The second section of the Permanent Library appearing on the Master Tape is Op File I consisting of an item for each routine. This file gives all the information that is necessary in allocating storage to those routines used by the various segments of the Object Program.

The third section is the collection of Permanent Library routines coded relative to address 01000. These routines are read from the Master Tape by the processor and modified according to the Object Program addresses given by the allocator.

Permanent Library Catalog

```
     RE   LC7230
     IA   LC
 0   0    0        24              No words following
 1   31   46307    25266           FLEXPT
 2   0    0        50002
 3   32   30505    25171           GENPOW
 4   0    0        50012
 5   70   24543    07252           VAREXP
 6   0    0        50022
 7   46   50777    77777           LN
10   0    0        50031
11   30   72527    77777           EXP
12   0    0        50041
13   65   53546    67777           SQRT
14   0    0        50051
15   31   46662    67066           FLTCVT
16   0    0        50062
17   46   34656    65450           LISTRN
20   0    0        50077
21   54   30242    75450           READRN
22   0    0        50100
23   34   50662    67066           INTCVT
24   0    0        50112
     CA   LC25
```

Op File I for Permanent Library Routines

	RE	PF7230		
	IA	PF		
0	0	50002	2	FLEXPT
1	0	0	41	
2	0	50012	5	GENPOW
3	0	0	107	
4	0	50002	0	
5	0	50022	0	
6	0	50051	0	
7	0	50022	5	VAREXP
10	0	0	113	
11	0	50002	0	
12	0	50031	0	
13	0	50041	0	
14	0	50031	3	LN
15	0	0	67	
16	0	50002	0	
17	0	50041	3	EXP
20	0	0	77	
21	0	50002	0	
22	0	50051	3	SQRT
23	0	0	53	
24	0	50002	0	
25	0	50062	2	FLTCVT
26	0	0	203	
27	0	50077	2	LISTRN
30	0	0	234	
31	0	50100	2	READRN
32	0	0	606	
33	0	50112	2	INTCVT
34	0	0	73	
35	74	74747	47474	End Sentinel
	CA	PF36		

Region Definitions for Permanent Library Loading

```
RE   TP10          50002

RE   PW200         50012

RE   VP370         50022

RE   LN560         50031

RE   EX750         50041

RE   SQ1140        50051

RE   VC1330        50062

RE   WI1710        50077

RE   NI2270        50100

RE   CI3230        50112

RE   PT1000        50002

RE   GP1000        50012

RE   VE1000        50022

RE   LG1000        50031

RE   XP1000        50041

RE   SR1000        50051

RE   CV1000        50062

RE   IW1000        50077
RE   XE1147
RE   RV1157
RE   TZ1200
RE   GT210
RE   BI71004
RE   BF71005
RE   TN71003

RE   IN1000        50100
RE   DR1012
RE   BM1101
RE   ST1131
RE   PS1146
RE   SC1233
```

```
RE    MF1243
RE    TB1252
RE    EP1264
RE    GG1303
RE    CF1473
RE    NT1551
RE    XX1552
RE    CC1557
RE    FX71000
RE    BU610
RE    GT210
RE    PR77250

RE    IC1000          50112
```

Flex-Print Routine (50002)

Purpose

To print out a stored sequence of Flex codes, with automatic carriage returns where required.

Calling Sequence

```
α       TP   L(Parameter 1)    PT3
+1      TP   L(Parameter 2)    PT4
+2      RJ          PT2        PT
+3        Control returned here
```

Parameter 1 is 00 - L - n, where L = initial loc. of stored Flex codes, and n = no. of words. The characters should be packed to the left, and filled in with zeros.

Parameter 2 is 00 - x - x, where x = address of an index previously set to the desired number of characters per line. Each time a character is printed by this routine, this index is reduced, and when zero a c.r. is given and the index reset to 79_{10}.

Storage

37_8 orders and constants
2 erasable locations

Flex-Print Routine (50002)

```
          IA    TP
          0     50002   45          Call word; No. of lines prelude &
                                      routine
          0     0       31          No. of lines for address modification
          0     0       6           No. of unmodifiable constants
          0     0       2           No. of inputs
          0     0       0           No. of outputs
          31    46307   25266       Name (FLEXPT)
PT    0   MJ    0       PT5
      1   RJ    30000   30000
      2   MJ    0       [30000]     EXIT
      3   [0    30000   30000]      Parameter; u = init. loc.; v = No. of words
      4   [0    30000   30000]      Line index address.  u & v = address
      5   TP    PT31    PT37        Zeroize index
      6   TV    PT3     PT37        Set it up
      7   TU    PT3     PT14        Location of lst word
     10   TU    PT4     PT22 }      Set line index addresses
     11   TV    PT4     PT24 }
     12   IJ    PT37    PT14        Countdown
     13   MJ    0       PT2
     14   TP    [30000] Q           Word for printing to Q
     15   LQ    Q       6
     16   PR    0       Q45         Print one character
     17   QT    PT32    PT40        Save it
     20   EJ    PT33    PT25 }      Test for non-printing character
     21   EJ    PT34    PT25 }
     22   IJ    [30000] PT25
     23   PR    0       PT16        Car. ret. (if necessary)
     24   TP    PT35    [30000]     Reset index
     25   RS    Q       PT40        Delete character already printed
     26   ZJ    PT15    PT27
     27   RA    PT14    PT36        Modify
     30   MJ    0       PT12        and back
     31   0     0       0           Constant
     32   0     0       77
     33   0     0       47
     34   0     0       57
     35   0     0       117         79 decimal
     36   0     1       0           Constant
     37   CA    TP45                Erasable (number of words index)
     40                             Erasable (Temp. storage for printed
                                      digit code)
```

General Power Routine (50012)

This routine is called from the Permanent Library as a result of the appearance in the source program of an expression of the form X POW Y where Y is unknown during compilation.

The routine tests the values of X and Y during the Object Program to determine what other routine, if any, of the Permanent Library is needed to evaluate the expression. For special cases of X and Y values the expression is evaluated within this routine. These include integral values for $|y| < 64$; $x = 0$, $y > 0$; $x \neq 0$, $y = 0$. Other cases are handled by reference to the Variable Exponent or to the Square Root routines.

GENPOW (50012) X^Y

```
                IA   PW
                0    50012    126
                0    0        104        No. of words for address modification
                0    0        14         No. of unmodifiable constants
                0    0        2
                0    0        1
                32   30505    25171      Name (GENPOW)
GP        0     MJ   0        GP6        Entry
          1     RJ   30000    30000
          2     MJ   0        30000      Exit
          3     0    0        0          Out
          4     0    0        0          In  X (base)
          5     0    0        0          In  Y (exponent)
          6     SP   GP4      0          }
          7     ZJ   GP25     GP10       }  X = 0?
         10     EJ   GP5      GP17       Y = 0?
         11     TJ   GP5      GP22       Y > 0?
         12     TP   GP70     50002      }
                10   0        3          }  RUN ERROR 2 (X = 0, Y< 0)
         13     TP   GP71     50002      }
                10   0        4          }
         14     TP   GP73     GP106      Set index
         15     RJ   50002    50002
                10   2        0
         16     MS   0        GP2
         17     TP   GP67     50002      }
                10   0        3          }  RUN ERROR 1 (X = 0, Y = 0)
         20     TP   GP71     50002      }
                10   0        4          }
         21     MJ   0        GP14
         22     TP   A        Q          }
         23     TP   A        GP3        }  X = 0, Y > 0  ⇒  X^Y = 0
         24     MJ   0        GP2
         25     TM   GP5      A          }
         26     ZJ   GP32     GP27       }  X ≠ 0, Y = 0  ⇒  X^Y = 1
         27     TP   GP76     Q          }
         30     TP   Q        GP3        }
         31     MJ   0        GP2
         32     EJ   GP75     GP40       |Y| = 1/2?
         33     TJ   GP77     GP43       |Y| < 64 ?
         34     TP   GP4      50022      }
                10   0        4          }
         35     TP   GP5      50022      }  Use VAREXP (50022)
                10   0        5          }
         36     RJ   50022    50022      }
                10   2        25         }
```

```
37   MJ  0       GP2
40   TP  GP4     50051
     10  0       4        }   Use SQRT (50051)
41   RJ  50051   50051
     10  2       0
42   MJ  0       GP62
43   LT  10011   Q            Mantissa of |Y| →Q35...9
44   LT  0       A            Characteristic + 128 → A
45   SS  GP74    0            Less bias
46   SJ  GP34    GP47         If characteristic < 0 use VAREXP
47   TV  A       GP50    }    Shift mantissa by characteristic
50   SP  Q       30000   }
51   LT  0       Q            Integral part →Q
52   SP  A       0       }    If |Y| non-integral use VAREXP
53   ZJ  GP34    GP54    }
54   SP  Q       0       }    |Y| integral so form |Y| -1
55   SS  GP72    17      }
56   TU  A       GP60
57   TP  GP4     Q            X → Q
60   RP  30000   GP62    }    Form X^|Y| in Q
61   FM  Q       GP4     }
62   TP  GP5     A
63   SJ  GP64    GP65         Y < 0?
64   FD  GP76    Q            Form reciprocal of (Q) if Y < 0
65   TP  Q       GP3          Store result in output line
66   MJ  0       GP2
67   0   GP100   3            Parameter for RUN ERROR 1
70   0   GP103   3            Parameter for RUN ERROR 2
71   0   GP106   GP106        Index address
72   0   0       1            1
73   0   0       117          79
74   0   0       200          128
75   20  4000    0            1/2 F.P.
76   20  14000   0            1 F.P.
77   20  74000   0            64 F.P.
100  45  45471   23406        CR CR ↑ R U N
101  04  20121   20312        Δ E R R 0 R
102  04  57524   20445        Δ ↓ 1 . Δ CR
103  45  45471   23406        CR CR ↑ R U N
104  04  20121   20312        Δ E R R 0 R
105  04  57744   20445        Δ ↓ 2 . Δ CR
106  CA  PW126                Temporary
```

Variable Exponent Routine (50022)

This routine is referenced directly from the Object Program equations as a result of the appearance in the Source Program of expressions X POW Y or X^a, where Y and a are known during compilation, but are neither integers with magnitude less than 64, nor are they equal in magnitude to 1/2.

As stated previously, this routine may also be referenced by the General Power routine as a result of the above determinations during the Object Program.

The routine references the Natural Logarithm routine and the Exponential routine to evaluate the expression.

VAREXP (50022) X^Y

```
          IA    VP
          0     50022     133          Cw; No. of lines  Prelude & Routine
          0     0         107          No. of words for address modification
          0     0         16           No. of unmodifiable constants
          0     0         2
          0     0         1
          70    24543     07252        Name (VAREXP)
VE    0   MJ    0         VE6          Entry
      1   RJ    30000     30000
      2   MJ    0         30000        Exit
      3   0     0         0            Out
      4   0     0         0            In  X (base)
      5   0     0         0            In  Y (exponent)
      6   SP    VE4       0
      7   ZJ    VE25      VE10         X = 0?
     10   EJ    VE5       VE17         Y = 0?
     11   TJ    VE5       VE22         Y > 0?
     12   TP    VE71      50002
          10    0         3
     13   TP    VE73      50002        RUN ERROR 2  (X = 0, Y < 0)
          10    0         4
     14   TP    VE76      VE112        Set index
     15   RJ    50002     50002
          10    2         0
     16   MS    0         VE2
     17   TP    VE70      50002
          10    0         3            RUN ERROR 1  (X = 0, Y = 0)
     20   TP    VE73      50002
          10    0         4
     21   MJ    0         VE14
     22   TP    A         Q
     23   TP    A         VE3          X = 0, Y > 0  ⇒  X^Y = 0
     24   MJ    0         VE2
```

```
25   TM   VE5      A        }
26   ZJ   VE32     VE27     }   Is |Y| = 0?
27   TP   VE100    Q        }
30   TP   Q        VE3      }   X ≠ 0, |Y| = 0 ⇒ X^Y = 1
31   MJ   0        VE2
32   LT   10011    Q            Mantissa of |Y| → Q35...9
33   LT   0        A            128 + characteristic of |Y| → A
34   SS   VE77     0            Less bias
35   SJ   VE45     VE36         If characteristic of |Y| < 0, Y is
                                  non-integral
36   TJ   VE75     VE40         Characteristic ≥ 28? If so |Y| is even
                                  integer
37   MJ   0        VE52
40   TV   A        VE41         0 ≤ characteristic of |Y| ≤ 27
41   SP   Q        30000
42   LT   0        Q            Integral part to Q
43   SP   A        0        }
44   ZJ   VE45     VE53     }   Is |Y| an integer?
45   TP   VE4      A        }
46   SJ   VE47     VE52     }   Y non-integral, is X > 0?
47   TP   VE72     50002    )
     10   0        3        }
50   TP   VE73     50002    }   RUN ERROR 3  (X< 0, Y non-integral)
     10   0        4        )
51   MJ   0        VE14
52   LT   0        Q            Set (Q) = 0 if Y is even integer or
                                  non-integral
53   QT   VE74     VE3          If Y is odd integer set output line to
                                  1 indicator
54   TM   VE4      50031    )
     10   0        4        }   Compute  LN |X| in Q
55   RJ   50031    50031    )
     10   2        0
56   FM   Q        VE5          Form Y · LN |X|
57   TP   Q        50041    )
     10   0        4        }   Compute  e^(Y·LN|X|) = |X|^Y → Q
60   RJ   50041    50041    )
     10   2        0
61   TP   VE4      A        }
62   SJ   VE63     VE66     }   Is X< 0?
63   SP   VE3      0        }
64   ZJ   VE65     VE66     }   Is Y an odd integer?
65   TN   Q        Q            X< 0 and Y is odd integer, so form
66   TP   Q        VE3            - |X|^Y → Q
67   MJ   0        VE2
70   0    VE101    3            Parameter for RUN ERROR 1
71   0    VE104    3            Parameter for RUN ERROR 2
72   0    VE107    3            Parameter for RUN ERROR 3
73   0    VE112    VE112        Index address
74   0    0        1            1
75   0    0        34           28
76   0    0        117          79
```

```
 77   0   0        200          128
100   20  14000    0            1 F.P.
101   45  45471    23406        CR  CR  ↑  R  U  N
102   04  20121    20312        Δ   E   R  R  0  R
103   04  57524    20445        Δ   ↓   1  .  Δ  CR
104   45  45471    23406        CR  CR  ↑  R  U  N
105   04  20121    20312        Δ   E   R  R  0  R
106   04  57744    20445        Δ   ↓   2  .  Δ  CR
107   45  45471    23406        CR  CR  ↑  R  U  N
110   04  20121    20312        Δ   E   R  R  0  R
111   04  57704    20445        Δ   ↓   3  .  Δ  CR
112   CA  VP133                 Temporary
```

Natural Logarithm Routine (50031)

This routine is adapted from the USE subroutine, WFMR03, Natural Logarithm - Single Precision Floating Point.

Given x, this routine computes $y(x) = {}^{*}\mathrm{Log}_e\ x$

with accuracy: $|y(x) - \mathrm{Log}_e\ x| \leq 2^{-27}$

where x ranges: $2^{-129} \leq x < 2^{127}$

For numerical method see Rand Sheet 42.

---

*Approximation.

Natural Logarithm Routine (50031)

```
        IA   LN
        0    50031    77          Cw; No. of lines on tape
        0    0        53          No. of lines for address modification
        0    0        16          No. of unmodifiable constants
        0    0        1           No. of inputs
        0    0        1           No. of outputs
        46   50010    10101       Name (LN)
LG0     MJ   0        LG5         Entry
  1     RJ   30000    30000
  2     MJ   0        30000       Exit
  3     0    0        0           Out
  4     0    0        0           In      x
  5     LQ   LG4      A
  6     TJ   LG57     LG40        x > 0?  If not, alarm
  7     UP   Q        A           Unpack x
 10     SS   LG55     11    }     Form and store C-1
 11     LT   0        LG3   }
 12     SP   Q        7     )
 13     AT   LG54     Q     )     Form and store y = (2u - Root 2)/
 14     SS   LG56     42    )      (2u + Root 2) scaled 34
 15     DV   Q        LG66  )
 16     MP   Q        Q     }     Form and store y^2 scaled 34
 17     LT   2        LG4   }
 20     TP   LG47     LG24  )     Form (LN2u-2)/y - (LN 2)/2 --> Q
 21     TP   LG50     Q     )       scaled 35
 22     MP   Q        LG4   )
 23     LT   2        A     )
 24     0    30000    30000 )
 25     RA   LG24     LG57  )
 26     TJ   LG13     LG22
 27     LA   LG66     A1    }     Form 2Y + (LN2)/2 scaled  69 --> A
 30     SA   LG60     43    }
 31     MA   Q        LE66  }     Form LN2u scaled 33 --> A
 32     LT   0        A     }
 33     MA   LG3      LG60  }     Form and store (C-1)LN2 + LN2u, scaled 28
 34     LT   37       LG3   }
 35     NP   LG3      LG61  }     Form and store LN x in floating point
 36     TP   LG3      Q     }
 37     MJ   0        LG2         Exit
 40     TP   LG62     LG66        Set index
 41     TP   LG45     50002 )
        10   0        3     )
 42     TP   LG46     50002 )     RUN ERROR 4  (LN x, x ≤ 0)
        10   0        4     )
 43     RJ   50002    50002 )
        10   2        0     )
```

```
44   MS   0       LG2
45   0    LG63    3          Parameter for RUN ERROR 4
46   0    LG66    LG66       Index address
47   AT   LG51    Q          Dummy command
50   11   50353   45377  )
51   14   62377   45540  }   Rand coefficients scaled 35
52   25   25255   47723  }
53   77   77777   77445  )
54   13   24047   46320      Root 2 scaled 33
55   20   10000   0          Bias for input
56   26   50117   14640      2 Root 2, scaled 33
57   0    1       0
60   05   42710   27760      LN2, scaled 33
61   17   70000   0          Bias for output
62   0    0       117        Print index
63   45   45471   23406      CR  CR ↑  R  U  N
64   04   20121   20312      Δ   E  R  R  0  R
65   04   57644   20445      Δ   ↓  4  .  Δ  CR
66   CA   LN77                Temporary
```

Exponential Routine (50041)

This routine is adapted from the USE subroutine, WFMR06, Exponential - Single Precision Floating Point.

Given x, this routine computes y(x) = EXP* x

with accuracy: $\left|\frac{y(x) - EXP\ x}{EXP\ x}\right| \leq 2^{-27}$

where x ranges: $-129\ \log_e 2 \leq x < 127\ \log_e 2$

For numerical method, see Ramo-Wooldridge EXP-2 in ERA Central Exchange Letter No. 8.

*Approximation

Exponential Routine (50041)

```
        IA    EX
        0     50041   107          Cw; no. of lines on tape
        0     0       56           No. of lines for address modification
        0     0       23           No. of unmodifiable constants
        0     0       1            No. of inputs
        0     0       1            No. of outputs
        30    72520   10101        Name (EXP)
XP0     MJ    0       XP5          Entry
  1     RJ    30000   30000
  2     MJ    0       30000        Exit
  3     0     0       0            Out
  4     0     0       0            In X
  5     TP    XP4     XP76  }
  6     UP    XP76    A     }      Unpack X
  7     LT    11      A     }
 10     AT    XP63    Q     }      Form C+8 in A and Q
 11     TJ    XP64    XP17         If C > 34, X is out of range
 12     TP    XP72    XP76         Set index
 13     TP    XP51    50002 |
        10    0       3     |
 14     TP    XP52    50002 |      RUN ERROR 5 (X is out of range)
        10    0       4     }
 15     RJ    50002   50002 |
        10    2       0     |
 16     MS    0       XP2   |
 17     SJ    XP20    XP22  }      If C+8 < 0, form C+44 in A
 20     SA    XP65    0     }
 21     SJ    XP25    XP22         If C+44 < 0, EXP X = 1
 22     TV    A       XP23
 23     LA    XP76    30000 |
 24     QJ    XP25    XP26  }      Set X in A scaled 35
 25     LT    0       A     |
 26     SA    XP66    0            X = K LN2 + R where R ≤ |LN2/2|
 27     DV    XP67    XP76
 30     ST    XP66    XP3
 31     TU    XP50    XP35  |      Compute EXP R scaled 34
 32     TP    XP53    Q     |
 33     MP    Q       XP3   |
 34     LT    1       A     }
 35     AT    30000   Q     |
 36     RA    XP35    XP70  |
 37     TJ    XP10    XP33  |
 40     NP    Q       XP71         Pack and adjust characteristic of EXP X
 41     LA    XP76    A33
 42     AT    Q       XP3
 43     SJ    XP44    XP45  }      If characteristic underflow occurs,
 44     RS    XP3     A     }        EXP X is zero
 45     TP    XP3     Q
 46     EJ    A       XP2          EXP X out of range?
 47     MJ    0       XP12         Yes, so ALARM
 50     0     XP54    0            Set-up
```

51	0	XP73	3	Parameter for RUN ERROR Print
52	0	XP76	XP76	Index address
53	0	150	35404	
54	0	1333	23520	
55	0	10421	01327	
56	0	52525	06225	
57	02	52525	25343	
60	10	0	271	
61	17	77777	77777	
62	20	0	0	
63	77	77777	77607	-120
64	0	0	53	43
65	0	0	44	36
66	13	5620	57737	(LN 2)/2 scaled 35
67	26	13441	37677	LN 2 scaled 35
70	0	1	0	
71	17	10000	0	
72	0	0	117	
73	45	45471	23406	CR CR ↑ R U N
74	04	20121	20312	Δ E R R O R
75	04	57624	20445	Δ ↓ 5 . Δ CR
76	CA	EX107		Temporary

Square Root Routine (50051)

This routine is adapted from the USE subroutine, WFMR04, Square Root - Single Precision Floating Point.

Given x, this routine computes $y(x) = \sqrt{x}$ *

with accuracy: $\left| \frac{y(x) - \sqrt{x}}{\sqrt{x}} \right| \leq 2^{-27}$

where x ranges: $0 \leq x < 2^{127}$

The numerical method is by Newton-Raphson approximation.

*Approximation

Square Root Routine (50051)

```
      IA    SQ
      0     50051   63          Cw; no. of words on tape
      0     0       50          No. of lines for address modification
      0     0       5           No. of unmodifiable constants
      0     0       1           No. of inputs
      0     0       1           No. of outputs
      65    53546   60101       SQRT
SR0   MJ    0       SR5         Entry
  1   RJ    30000   30000
  2   MJ    0       30000       Exit
  3   0     0       0           Out
  4   0     0       0           In  X
  5   TP    SR4     A71
  6   SJ    SR7     SR14
  7   TP    SR45    SR52        Set index
 10   TP    SR43    50002
      10    0       3
 11   TP    SR44    50002       RUN ERROR 6 (SQRT X or X< 0)
      10    0       4
 12   RJ    50002   50002
      10    2       0
 13   MS    0       SR2
 14   LT    10      SR4         If characteristic is odd (even), form
 15   TP    A       Q             and store y = 2u (u), scaled 35
 16   QJ    SR17    SR20
 17   SA    A       0
 20   TP    A       SR3         If X = 0, SQRT  X = 0
 21   ZJ    SR22    SR2
 22   SP    A       0
 23   SA    SR46    37          First approximation is (y+(2 Root 2-1)/
 24   LT    0       SR52          2)/2 scaled 31
 25   SP    SR3     33
 26   DV    SR52    A           Newton-Raphson Iteration
 27   AT    SR52    SR52
 30   SP    SR3     35
 31   DV    SR52    A
 32   AT    SR52    SR52
 33   SP    SR3     37
 34   DV    SR52    A
 35   AT    SR52    SR3
 36   SP    SR4     0           Form characteristic of Root X
 37   SA    SR5     33
 40   NP    SR3     A
 41   TP    SR3     Q           Pack and exit
 42   MJ    0       SR2
 43   0     SR47    3           Parameter for RUN ERROR 6
 44   0     SR52    SR52        Index address
 45   0     0       117
 46   35    20236   31500       (2 Root 2-1)/2, scaled 35
```

```
47   45   45471   23406          CR CR ↑ R U N
50   04   20121   20312          Δ E R R O R
51   04   57664   20445          Δ ↓ 6 . Δ CR
52   CA   SQ63                   Temporary
```

Floating-Point Conversion & Print-Out (50062)

Purpose

To convert to decimal, and print out on-line, a binary floating-point number, with a carriage return depending on the contents of a specified index.

Calling Sequence

```
α     TP  L(Number)     CV3
+1    TP  L(Parameter)  CV4
+2    RJ  CV2           CV
+3        Control returned here.
```

The parameter is the same as "Parameter 2" described in the Flex-print routine.

Storage

176_8 orders and constants.
5 erasable locations.

Floating Point to Decimal Print Routine (50062)

```
        IA      VC
        0       50062     204         Call word; no. of lines prelude + routine
        0       0         142         No. of lines for address modification
        0       0         34          No. of unmodifiable constants
        0       0         2           No. of inputs
        0       0         0           No. of outputs
        31      46662     67066       Name (FLTCVT)
CV0     MJ      0         CV5         Entrance
  1     RJ      30000     30000
  2     MJ      0         [30000]     Exit
  3     [0      30000     30000]      Quantity for conversion
  4     [0      30000     30000]      Line index address
  5     PR      0         CV10        Shift down
  6     TU      CV4       CV134}      Set up line index address
  7     TV      CV4       CV136}
 10     TP      CV3       A57         Quantity ⟶ A
 11     ZJ      CV15      CV12        Zero?
 12     PR      0         CV163       Yes  Print Zero
 13     RJ      CV137     CV134       Go count
 14     MJ      0         CV2         And out
 15     TP      CV142     CV176       Zeroize dec. exp. store
 16     TM      A         Q56         |q| ⟶ Q
 17     SJ      CV20      CV22        Negative?
 20     PR      0         CV16        Yes, print - sign
 21     RJ      CV137     CV134       Count
 22     SP      Q         0           |q|⟶A
 23     TJ      CV146     CV32        Test v.  MIN.  Jump if too small
 24     TJ      CV151     CV36        Test v.  MAX.+ 1.  Jump if O.K.
 25     FD      A         CV150       Too big, divide by 10
 26     RA      CV176     CV162       Increase dec. exp. by 1
 27     SP      Q         0
 30     TJ      CV150     CV36        Test v. 10.  If < , then O.K.
 31     MJ      0         CV25
 32     FM      A         CV150       Too small, so multiply by 10
 33     RS      CV176     CV162       Decrease exponent by 1
 34     SP      Q         0
 35     TJ      CV147     CV32        If < 1, back again
 36     TP      A         CV177       O.K., in scale now.  Send to erasable
 37     UP      CV177     A42         Characteristic⟶A
 40     SS      CV145     55          C - 119⟶A_V
 41     TV      A         CV42
 42     SP      CV177     [0]         Position point between A_L, A_R
 43     TP      A         CV177       Save fractional part
 44     LT      0         A4          Integral part⟶A_R
 45     TP      CV143     CV200       8⟶index, as 9 digits wanted (in ALL)
 46     ZJ      CV52      CV47        Integral part zero?
 47     PR      0         CV163       Yes, so print
 50     RJ      CV137     CV134       Count
 51     MJ      0         CV57        Jump to process fractional part
```

```
        52    TU  CV140   CV116       Non-zero integral part.  Start dividing
                                        by 10^8
        53    TP  CV143   CV201       9 powers of 10 to be used (∴ 8→index)
        54    RJ  CV127   CV113       Go convert
        55    TP  CV200   A20   }     Now if  digit index -ve, no fractional
        56    SJ  CV2     CV57  }       part wanted and if so, out
        57    PR  0       CV37        Otherwise, print decimal point
        60    RJ  CV137   CV134       Go  Count
        61    SP  CV177   2           4f
        62    ZJ  CV63    CV72        Zero?
        63    SA  CV177   1           No, then digit formed in A_L
        64    TP  A       CV177       Preserve remainder
        65    LT  0       A45         Digit to A_R
        66    AT  CV12    CV67        Form print order
        67    [0  30000   30000]
        70    RJ  CV137   CV134       Count
        71    IJ  CV200   CV61        Back for 9 digits in all
        72    TP  CV176   A           How about decimal exponent?
        73    ZJ  CV74    CV2         If zero, out.
        74    PR  0       CV44        Print space
        75    RJ  CV137   CV134       Count
        76    PR  0       CV55        Print E
        77    RJ  CV137   CV134       Count
       100    PR  0       CV44        Print space
       101    RJ  CV137   CV134       Count
       102    TP  CV176   A           Exp→A again
       103    SJ  CV104   CV107       Sign?
       104    PR  0       CV16        Negative, so print
       105    RJ  CV137   CV134       Count
       106    TM  CV176   A           |exp|→A
       107    TU  CV141   CV116       Start dividing with 10^1
       110    TP  CV162   CV201       2 powers of 10 (no matter about no. of
                                        digits printed)
       111    RJ  CV127   CV113       Convert exponent
       112    MJ  0       CV2         And out.
Integer
conver-
sion   113    TP  A       CV202       Int→working store
       114    TP  CV132   CV122       Set switch to suppress leading zeros
       115    SP  CV202   0           Prepare to divide
       116    DV  [30000] Q           Divide by 10^n
       117    TP  A       CV202       Save remainder
       120    RA  CV116   CV175       Modify divisor address
       121    SP  Q       0           Examine quotient
       122    [0  30000   30000]      ZJ              ¦ AT            CV123
       123    [0  30000   30000]      PR   0
       124    RJ  CV137   CV134       Count across line
       125    RS  CV200   CV162       Count 1 output digit
       126    IJ  CV201   CV115       Count 1 power 10
       127    MJ  0       [30000]     Exit
```

```
130    TP   CV133    CV122 }
131    MJ   0        CV122 }   Set switch to commence printing
132    ZJ   CV130    CV126 }
133    AT   CV12     CV123 }   Instructional constants
134    IJ   [30000]  CV137     Count down on line index
135    PR   0        CV65      Carriage return
136    TP   CV144    [30000]   Reset index
137    MJ   0        [30000]   And out
140    0    CV152    0         Address of 10^8
141    0    CV161    0         Address of 10^1
142    0    0        0         Zero
143    0    0        10        8
144    0    0        117       79
145    16   70000    0         119 in ch. field
146    DE   1.0 ∧ -1∧F          Min.
147    DE   1.0∧0∧F             1
150    DE   1.0∧1∧F             10
151    DE   1.0∧9∧F             Max. +1
152    0    5753     60400     10^8
153    0    461      13200     10^7
154    0    36       41100     10^6
155    0    3        03240     10^5
156    0    0        23420     10^4
157    0    0        1750      10^3
160    0    0        144       10^2
161    0    0        12        10^1
162    0    0        1         10^0
163    0    0        37        0 }
164    0    0        52        1 }
165    0    0        74        2 }
166    0    0        70        3 }
167    0    0        64        4 }   Flex codes
170    0    0        62        5 }
171    0    0        66        6 }
172    0    0        72        7 }
173    0    0        60        8 }
174    0    0        33        9 }
175    0    1        0
176    CA   VC204               Dec. exponent   }
177                              Fractional part }   Working space
```

"Inner" List Routine (50077)

Purpose

Either: - To convert a binary floating-point number to decimal XS3 form, and insert in buffer

Or: - To convert a binary fixed-point number to decimal XS3 form, and insert in buffer

Or: - To test buffer for contents, and if anything there, to dump buffer onto tape, clearing out buffer afterwards.

Calling Sequence

1) To empty buffer onto tape (only after some other usage)

```
α      RJ   IW3   IW2
+1     ⟵ Control returned here ⟶
```

2) To write a floating-point number (n)

```
α      TP   L(n)   A
+1     RJ   IW3    IW
+2   ⟵ Control returned here ⟶
```

3) To write a fixed-point number (i)

```
α      TP   L(i)   A
+1     RJ   IW3    IW1
+2   ⟵ Control returned here ⟶
```

Initialization

Before the first usage of this routine, we must have:

```
TV    BI    IW121
```

where BI_V holds the address in the buffer of the first word to be written into. Each word in binary, when converted, occupies four words in the List buffer. Each List sentence will use this routine from 1 to 5 times, each time it is referenced. After the initializing order above, words will be sequentially written in the buffer (thus occupying, at most, one blockette). After all

such references in one List sentence coding, we must increment BI (the buffer index) so that the next reference starts in the next blockette (i.e., - next line) at the correct point.

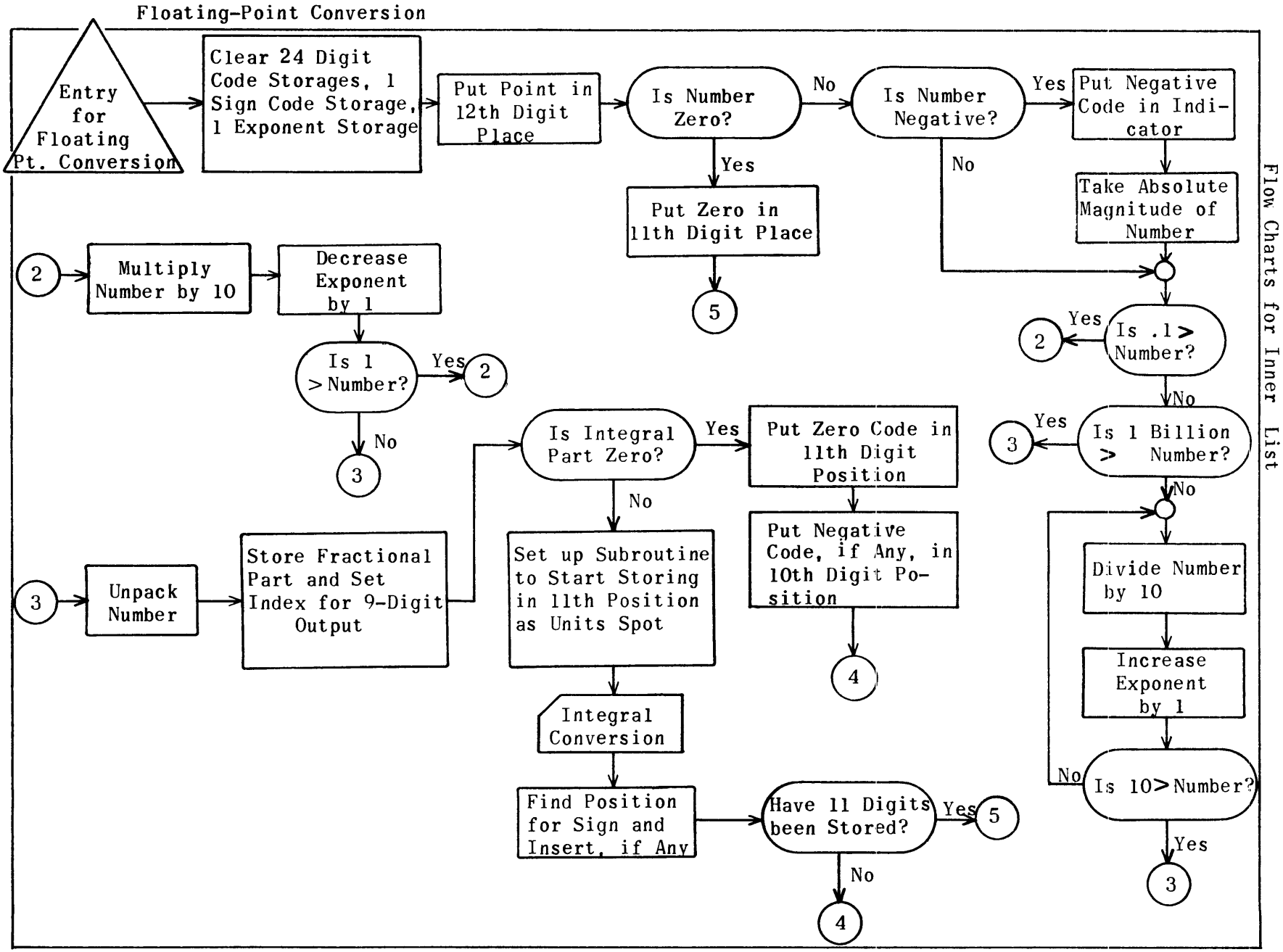
For 1 variable, increment BI by 4
For 2-5 variables, increment BI by 20_{10}

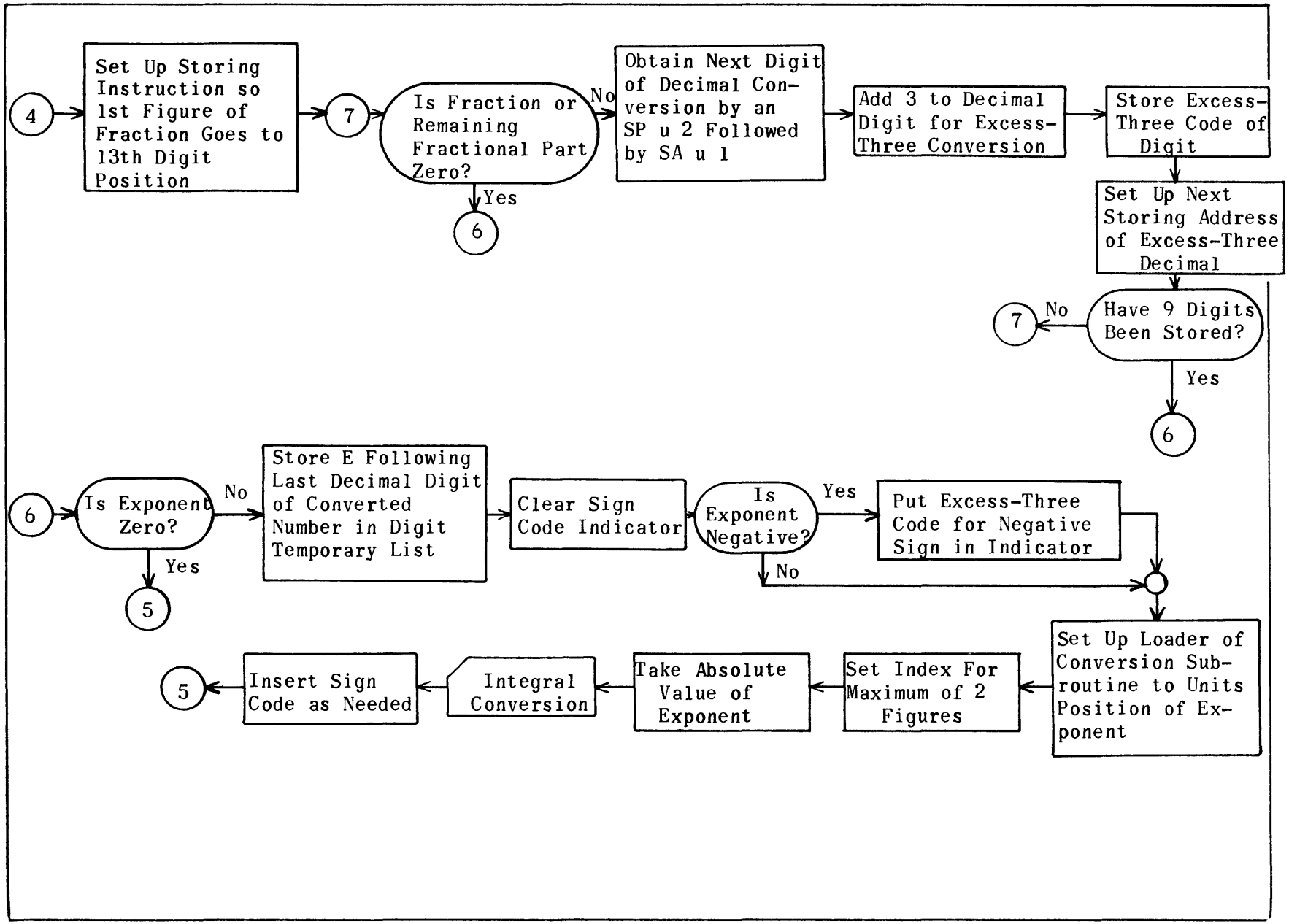
When this has been done, test the index to see if the buffer needs to be emptied. (Test with BF165, where BF is initial address of buffer). After any emptying, reset BI, in the case of 2 to 4 variables, in order to space the XS3 correctly in the middle of the paper.

For 2 variables, increment BI by 6
For 3 variables, increment BI by 4
For 4 variables, increment BI by 2

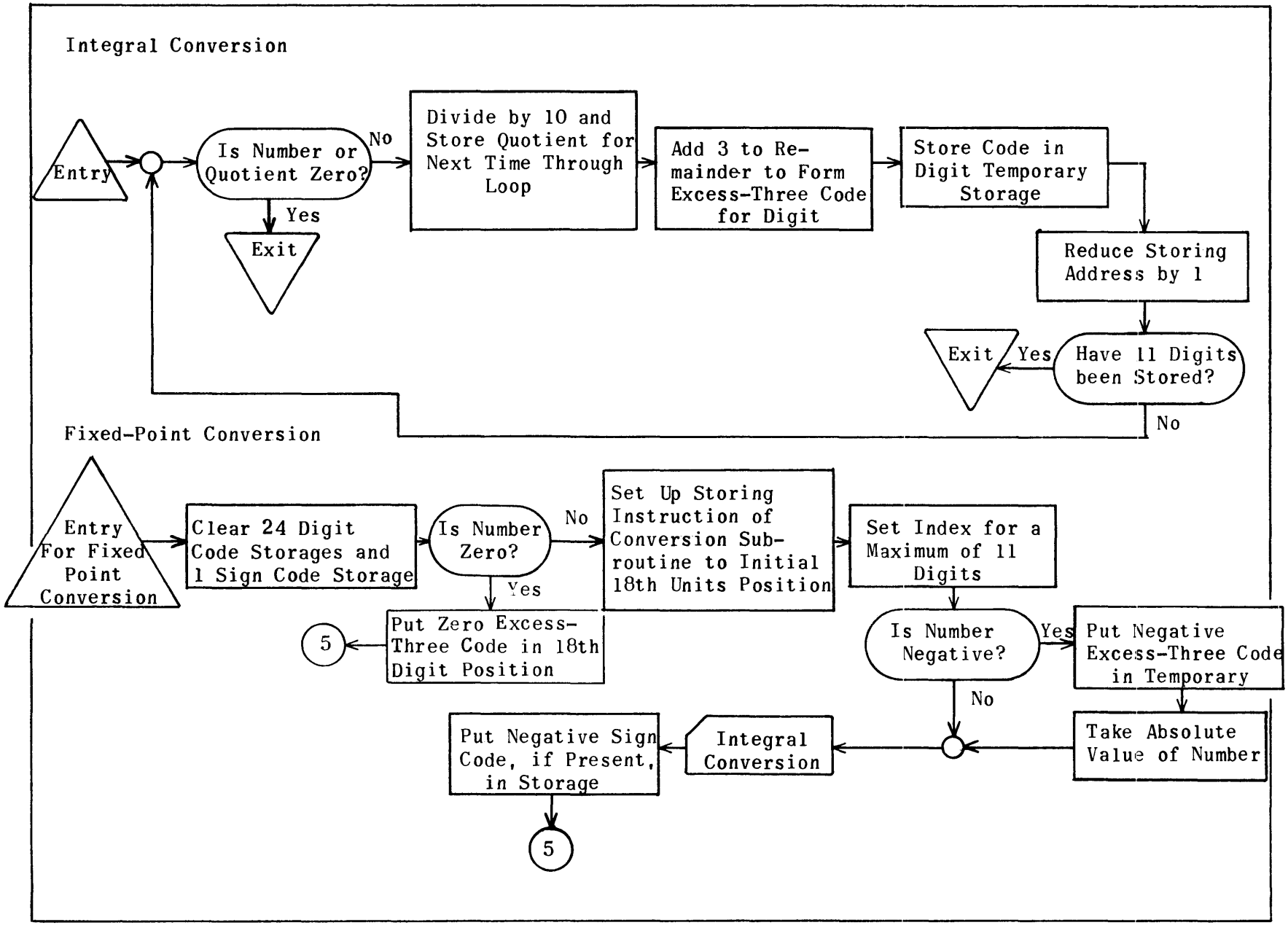
For 1 and 5 variables, writing starts at the left side of the paper.

In addition, fixed location TN should hold the current tape number.

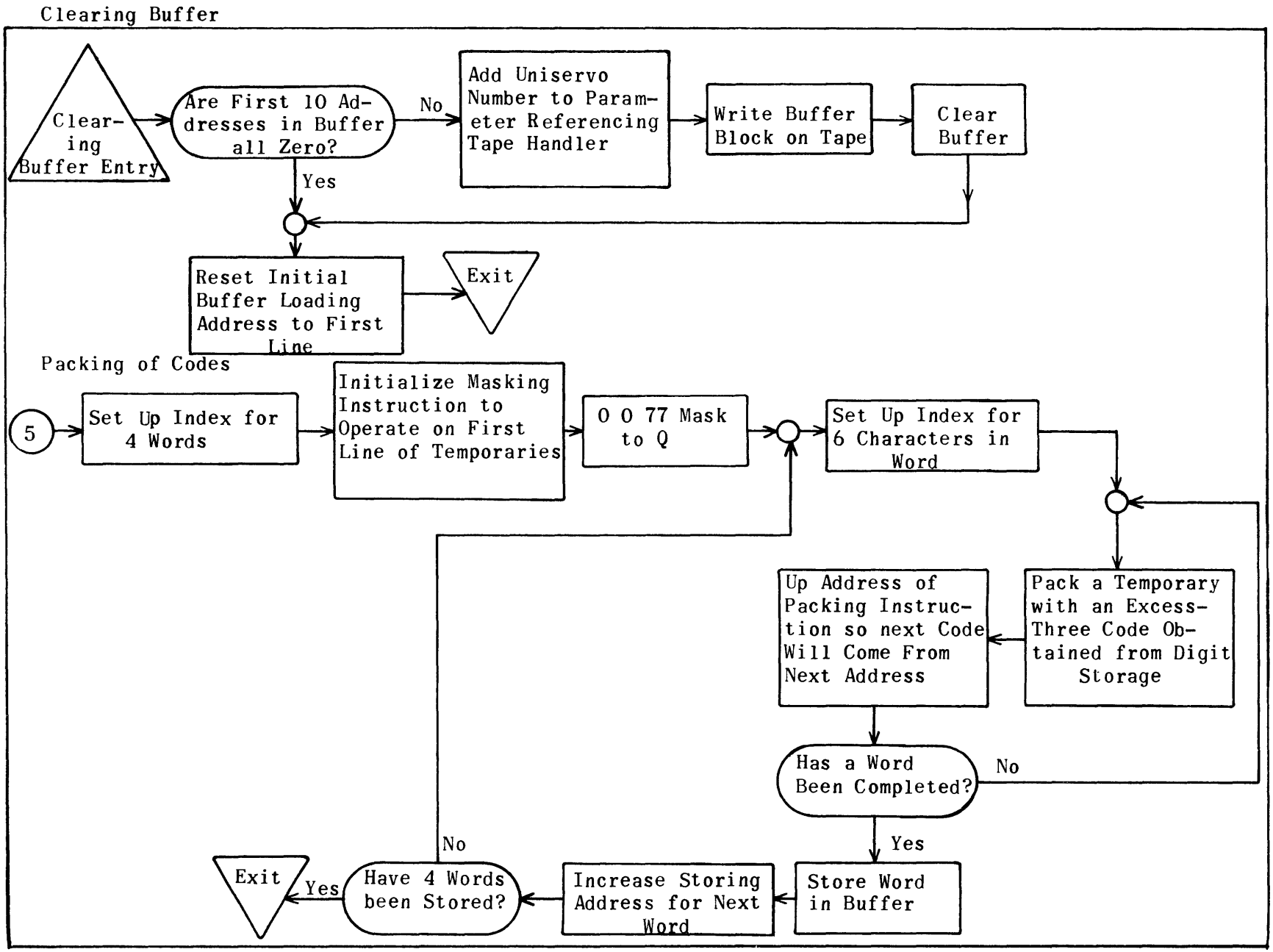

Floating-Point Conversion
Entry for Floating Pt. Conversion
Clear 24 Digit Code Storages, 1 Sign Code Storage, 1 Exponent Storage
Put Point in 12th Digit Place
Is Number Zero?
No
Is Number Negative?
Yes
Put Negative Code in Indicator
Take Absolute Magnitude of Number
Yes
Put Zero in 11th Digit Place
5
No
2
Multiply Number by 10
Decrease Exponent by 1
Is 1 > Number?
Yes
2
No
3
Is .1 > Number?
Yes
2
No
Is 1 Billion > Number?
Yes
3
No
Divide Number by 10
Increase Exponent by 1
Is 10 > Number?
No
Yes
3
3
Unpack Number
Store Fractional Part and Set Index for 9-Digit Output
Is Integral Part Zero?
Yes
Put Zero Code in 11th Digit Position
Put Negative Code, if Any, in 10th Digit Position
4
No
Set up Subroutine to Start Storing in 11th Position as Units Spot
Integral Conversion
Find Position for Sign and Insert, if Any
Have 11 Digits been Stored?
Yes
5
No
4



Flow Charts for Inner List (Cont.)



Flow Charts for Inner List (Cont.)


Clearing Buffer
Clear-ing Buffer Entry
Are First 10 Addresses in Buffer all Zero?
No
Add Uniservo Number to Parameter Referencing Tape Handler
Write Buffer Block on Tape
Clear Buffer
Yes
Reset Initial Buffer Loading Address to First Line
Exit
Packing of Codes
5
Set Up Index for 4 Words
Initialize Masking Instruction to Operate on First Line of Temporaries
0 0 77 Mask to Q
Set Up Index for 6 Characters in Word
Up Address of Packing Instruction so next Code Will Come From Next Address
Pack a Temporary with an Excess-Three Code Obtained from Digit Storage
Has a Word Been Completed?
No
Yes
Store Word in Buffer
Increase Storing Address for Next Word
Have 4 Words been Stored?
No
Yes
Exit

Inner List Routine (50077)

		IA	WI		
Prelude		0	50077	206	u is call word v is number of words in routine plus prelude
		0	0	157	Number of lines for address modification
		0	0	21	Number of unmodifiable constants
		77	77777	77775	Number inputs (-2 for Processor)
		0	0	0	Number outputs
		46	34656	65450	Name (LISTRN)
IW	0	MJ	0	IW4	Entry for floating-point conversion
	1	MJ	0	IW101	Entry for fixed-point conversion
	2	MJ	0	IW125	Entry for buffer emptying
	3	MJ	0	30000	Exit
	4	RP	10032	IW6	Clear 24 digit code stores, one sign code store, one exponent store
	5	TP	RV	TZ	
	6	TP	RV7	TZ13	Insert point code
	7	ZJ	IW12	IW10	Quantity zero?
	10	TP	RV3	TZ12	Yes, insert zero code
	11	MJ	0	IW111	Go pack
	12	SJ	IW13	IW15	Non-zero, test sign
	13	TP	RV2	TZ30	Negative, so note
	14	TM	A	A	
	15	TJ	RV17	IW24	Test v. Min., jump if too small
	16	TJ	RV20	IW30	Test v. Max. +1, jump if O.K.
	17	FD	A	RV16	Too big, divide by 10
	20	RA	TZ31	RV1	Increase exponent by 1
	21	SP	Q	0	
	22	TJ	RV16	IW30	If now < 10.0, O.K.
	23	MJ	0	IW17	Otherwise, divide again
	24	FM	A	RV16	Too small, multiply by 10
	25	RS	TZ31	RV1	Decrease exponent by 1
	26	SP	Q	0	
	27	TJ	RV15	IW24	If > 1.0, O.K., otherwise back
	30	TP	A	TZ33	Now in range
	31	UP	TZ33	A	Biased char. $\longrightarrow A_{CH}$
	32	SS	RV13	55	Position binary point between A_L and A_R
	33	TV	A	IW34	
	34	SP	TZ33	30000	
	35	TP	A	TZ33	Save fractional part
	36	LT	0	A	Integral part to A_R
	37	TP	RV5	TZ32	Set index to 8
	40	ZJ	IW44	IW41	Or is integral part zero?
	41	TP	RV3	TZ12	Yes, insert zero code
	42	TP	TZ30	TZ11	Place sign in position
	43	MJ	0	IW51	Go process fractional part
	44	TV	IW41	IW143	Units position is 11th digit code store
	45	RJ	IW146	IW137	Go convert integral part
	46	TV	IW143	IW47	Find sign position
	47	TP	TZ30	30000	And insert it
	50	SJ	IW111	IW51	If A < 0, no fractional digits (or exponent)
	51	TV	XE7	IW56	1st fractional position is 13th digit code store

```
                52   SP    TZ33    2
                53   ZJ    IW54    IW61
                54   SA    TZ33    1
                55   LT    44      TZ33
                56   AT    RV3     30000      Form and store code  } Fractional loop
                57   RA    IW56    RV1        Form and store code  }
                60   IJ    TZ32    IW52       Form and store code  }
                61   TP    TZ31    A          Now process decimal exponent
                62   ZJ    IW63    IW111
                63   TV    IW56    IW64       Find address for E
                64   TP    RV10    30000      Store E code
                65   TP    RV      TZ30       Clear sign code temp. store
                66   SJ    IW67    IW70  }    Note negative sign if wanted
                67   TP    RV2     TZ30  }    Note negative sign if wanted
                70   SP    IW64    0     }    Units position of decimal exponent
                71   SA    RV3     0     }      is 3 beyond "E" store
                72   TV    A       IW143 }    Units position of decimal exponent
                73   TP    RV1     TZ32       1 to index
                74   TM    TZ31    Q
                75   RJ    IW146   IW140      Go count
                76   TV    IW143   IW77  }    Insert sign code as required
                77   TP    TZ30    30000 }    Insert sign code as required
               100   MJ    0       IW111      Jump to pack
               101   RP    10031   IW103 }    Fixed-point entry. Clear 24(10) digit
               102   TP    RV      TZ    }      code stores and 1 sign code store
Fixed-         103   ZJ    IW106   IW104      Quantity zero
Point          104   TP    RV3     TZ21       Yes, so note
Con-           105   MJ    0       IW111      Jump to pack
version        106   RJ    XE6     XE         Patch correction
               107   TV    IW143   IW110 }    Negative sign in excess three
               110   TP    TZ30    30000 }      added, if number is negative
               111   TP    RV3     TZ32       3 to 1st index
               112   TU    XE7     IW116      Initialize
               113   TP    RV11    Q          Mask ⟶ Q
               114   TP    RV4     TZ30       5 to 2nd index
               115   LA    TZ33    6
Packing        116   QS    30000   TZ33
  of           117   RA    IW116   RV12
 Codes         120   IJ    TZ30    IW115
               121   TP    TZ33    30000      Store one word in buffer
               122   RA    IW121   RV1        Increment buffer address
               123   IJ    TZ32    IW114
               124   MJ    0       IW3        Jump to exit
               125   SP    RV      0
               126   RP    20012   IW135 }    If 1st 10 stores not > 0, buffer empty,
               127   TJ    BF      IW130 }      so out (but index may be at BF+n)
               130   SP    TN      17         Tape number to u of A
Clear          131   AT    RV14    GT3   }    Send parameter to GTH and write
Buffer         132   RJ    GT2     GT    }      block on tape
               133   RP    10171   IW135 }    Clear buffer
               134   TP    RV      BI    }    Clear buffer
               135   TV    RV14    BI         Reset buffer index
               136   MJ    0       IW3        Out
```

Integer Conversion	137	TP	A	Q	Integer → Q
	140	SP	Q	0	
	141	ZJ	IW142	IW146	
	142	DV	RV6	Q	Divide by 10
	143	AT	RV3	30000	Form and store code
	144	RS	IW143	RV1	
	145	IJ	TZ32	IW140	
	146	MJ	0	30000	Exit
Correction Patch for Fixed-Pt. Numbers	XE0	TV	IW104	IW143	No, units position is 18th temp. store
	1	TP	RV6	TZ32	Set index to 10_{10}
	2	SJ	XE3	XE5	Is number negative?
	3	TP	RV2	TZ30	Yes, put XS3 code in temporary
	4	TM	A	A	Take absolute value of number
	5	RJ	IW146	IW137	Go convert
	6	MJ	0	30000	
	7	0	TZ	TZ14	u is beginning address of temporaries v is location of period for floating pt.
RV Constants	0	0	0	0	
	1	0	0	1	
	2	0	0	2	Negative code
	3	0	0	3	
	4	0	0	5	
	5	0	0	10	
	6	0	0	12	
	7	0	0	22	Code for period
	10	0	0	30	Code for E
	11	0	0	77	
	12	0	1	0	
	13	16	70000	0	119 in ch. field
	14	74	20100	1005	GTH code
	15	DE	1.0	0 F	1.0
	16	DE	1.0	1 F	10.
	17	DE	1.0	-1 F	0.1 (Min.)
	20	DE	1.0	9 F	Max. + 1
		CA	WI206		

Explanation of Temporaries Used

TZ-TZ27 are used for storing the XS3 codes of the 4 words resulting from any one number that is listed by the routine. Signs, decimal digits, and the exponent letter E are stored here. If the number is floating-point, the units position is TZ12 and a period is put in TZ13. The fractional part of a floating-point number starts at TZ14. If the number is fixed-point, the units position is TZ21.

TZ30 — Minus XS3 code is stored here, as needed.
TZ30 — Holds index for packing 6 characters in a word.

TZ31 — Exponent of floating-point numbers that are to be expressed in scientific notation is built up here.

TZ32 — Index for 9 digits of fractional part of floating-point number.
TZ32 — Index for possible two digits of an exponent.
TZ32 — Index for 4-word output of any one input number.
TZ32 — Index for maximum 11 decimal digits of an output number.

TZ33 — Floating-point number in a certain range put here temporarily.
TZ33 — Fractional part of floating-point number put here as it is converted to decimal form.
TZ33 — Output words of any number are built up here prior to insertion in the Output Buffer of the List routine.

"Inner" READ Routine (50100)

Purpose

Either: To position data tape to beginning of required data

Or: To extract one word of decimally coded data from tape, and convert to binary floating point.

Calling Sequence

1) Tape positioning

α	TP	L(Variable Name)	IN1
+1	RJ	IN	IN2
+2	Abnormal "end of data" return		
+3	Normal return, with tape positioned		

α + 2 will hold a MJ instruction. If READ is of type including "IF END OF DATA JUMP TO SENTENCE---", then the appropriate sentence CW is filled into v. Otherwise, we have MJ 0 α , i.e., to return to beginning of data, and commence extraction over again.

2) Data Extraction

α	RJ	IN	IN3	
+1	TP	Q	[?]	Normal return, word in Q
+2	[RA			
+3	[IJ			
+4	Abnormal return, insufficient data			

If the routine is referenced enough times to cause the tape to move onto the next data set, an alarm stop is met. If START is hit, remainder of array is filled in with zeros, and exit is made at α + 4.

Initialization

None as such, but FX and FX1 must hold the initial location of the data index, and j n, n describing it, respectively.

Storage

606_8 orders, constants, and temps
+ 170_8 word buffer (which is termination buffer)
+ 2 fixed locations.

Note: This routine includes the XS3 to Floating-Point conversion routine from GGO-CF16. (See Section III, 3-Translation Subroutines.)

A Note on the Mechanism of Tape Positioning

The "data index" is read in and translated into the form shown in the accompanying diagram. When a variable is requested by generated READ coding, the index is searched to find this variable, and hence the "tape synonym." This is noted, and then scanning continued, to find the first "TAPE" - and hence the containing tape number, and hence the "indicator."

Now, at the start of the problem, the tapes are rewound, and "ΔDATAΔ" placed in the indicator. If this is seen, we scan the tape forward, searching for the required tape synonym. If not, we search the index again, to determine whether the "indicator" word can be found before the required synonym. If so, we scan the tape forward. If not, we scan backward. If the tape is at the extreme end, this also is signified by a special "indicator", and the tape scanned backward.

When positioning is completed, the synonym is placed in the indicator for this tape.

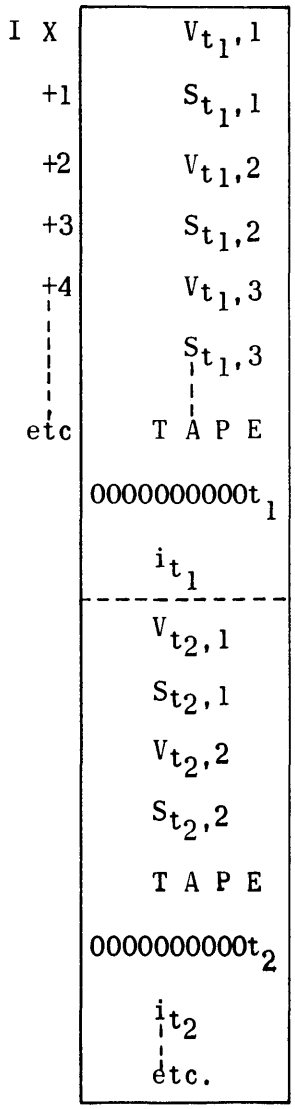
INDEX FORMAT AND DESCRIPTION

The necessary information to describe the data index is contained in two (running) fixed locations:

FX	0	L	L
+1	0	J,n	n

Where L is the initial location of the index, J = 2, and n = number of locations occupied.

Format



Where: V represents the name of the variable as referenced in the program

S represents the synonym for this data, as actually written on tape

t represents the octal tape number containing all preceding information, back as far as the previous "TAPE" word

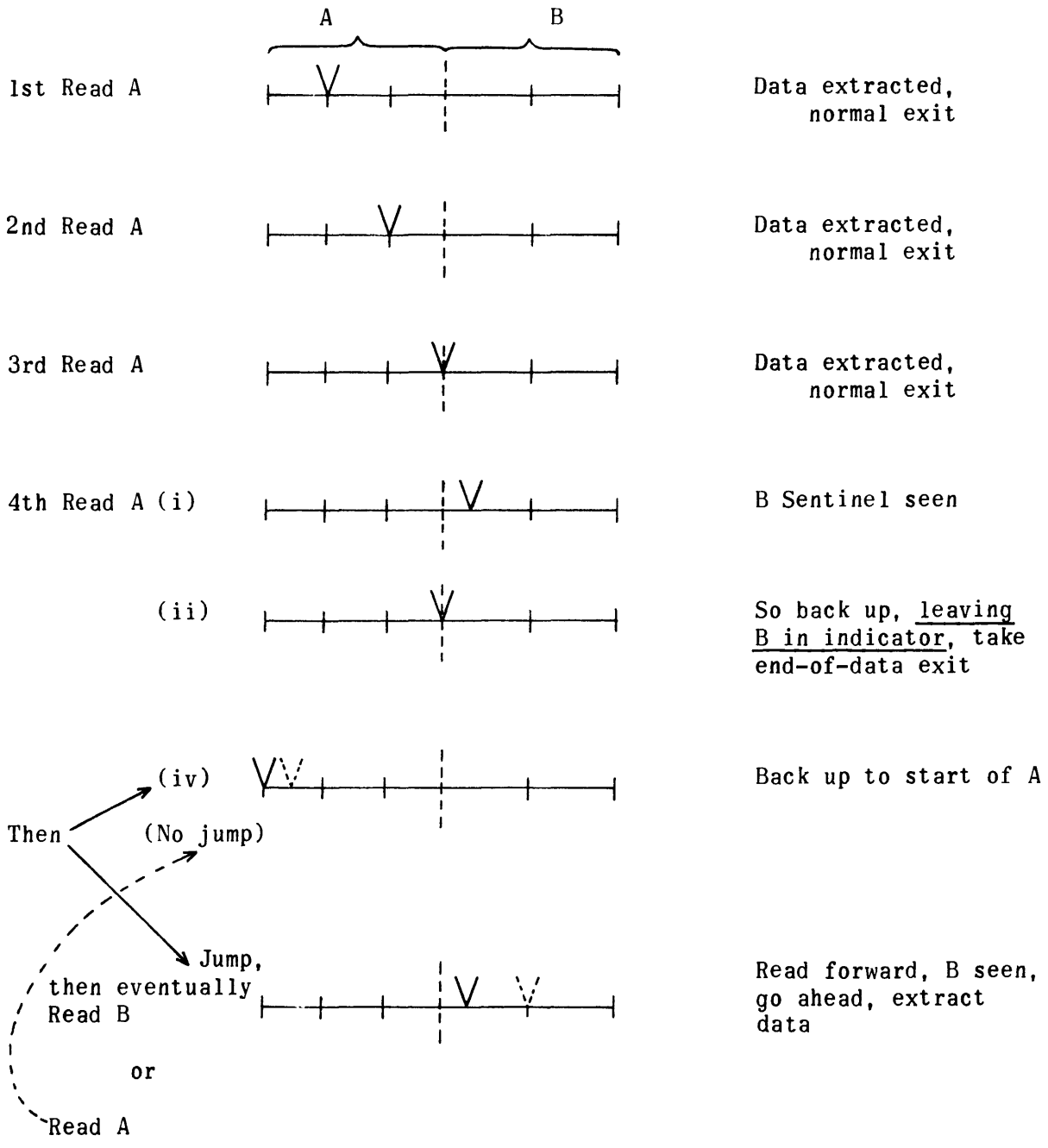
i_t represents the tape synonym of the data last read in from tape t.

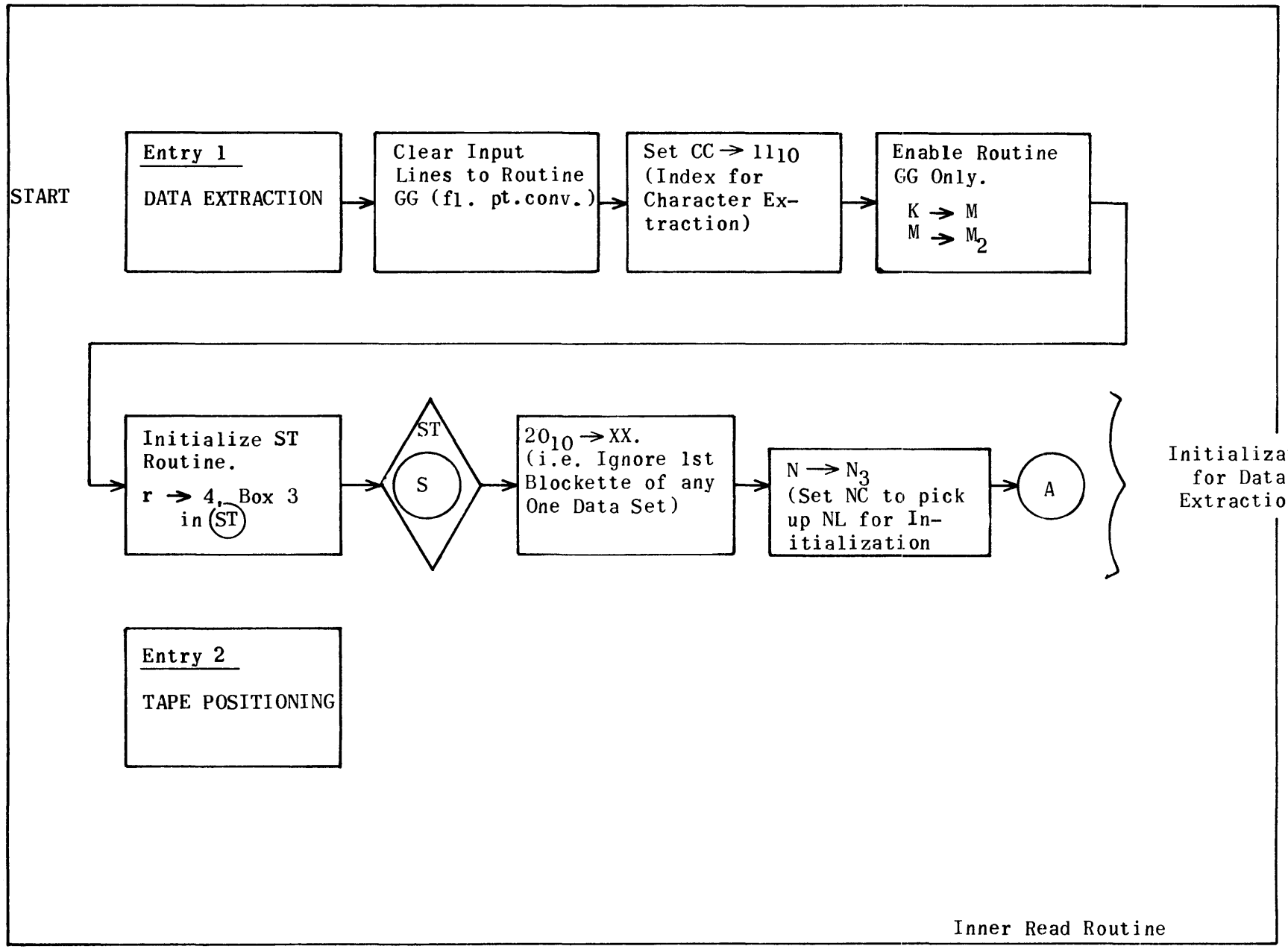
(N.B. This word may also hold "Δ DATA Δ", indicating that this tape is rewound, OR, "Δ ENDΔO", showing that the tape is at the end of all information written thereon.)

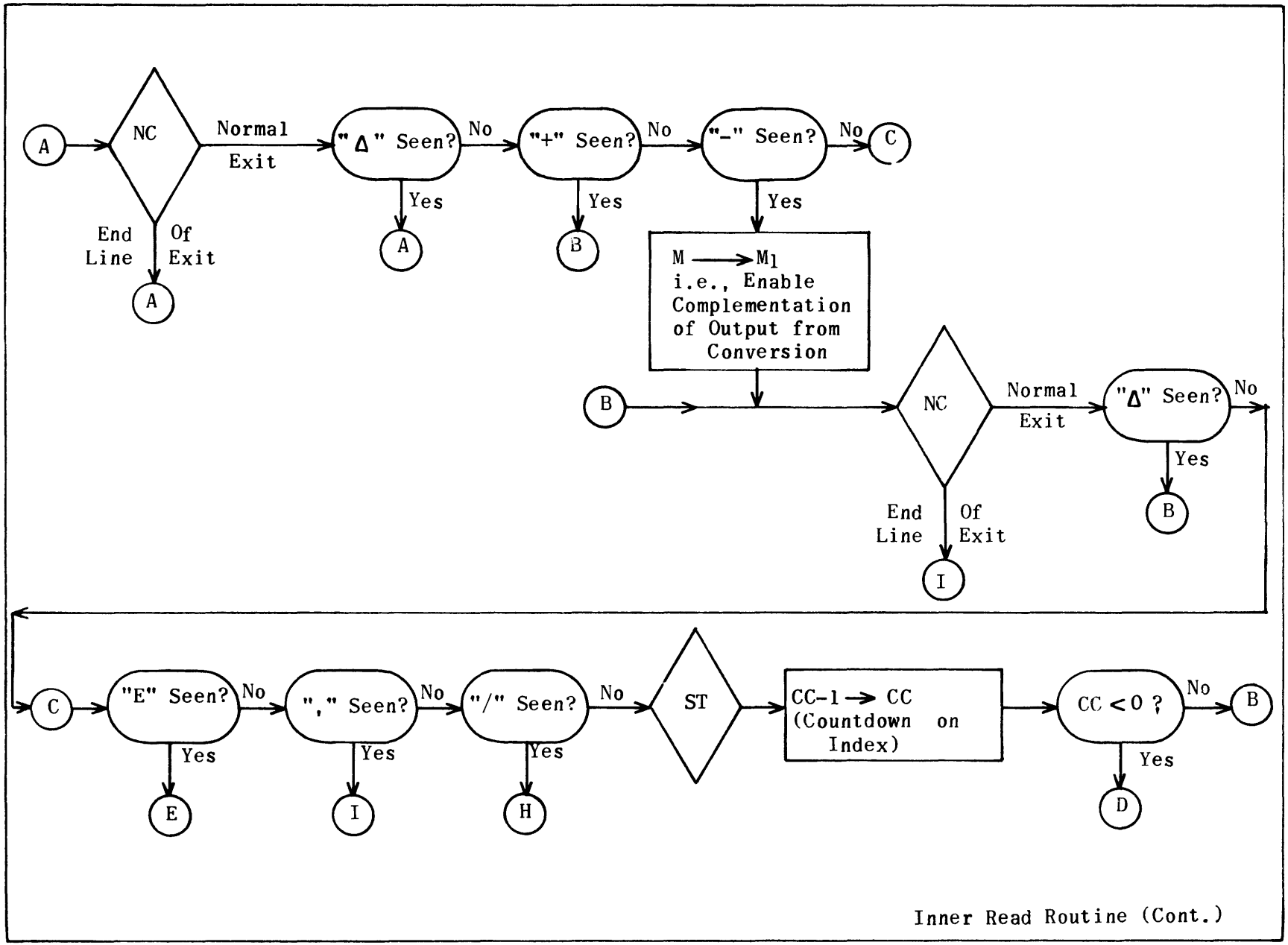
V's will be XS-3 coded, pushed to the left of the word, and filled with 77's.
S's will be XS-3 coded, pushed to the right of the word, and filled with space symbols.

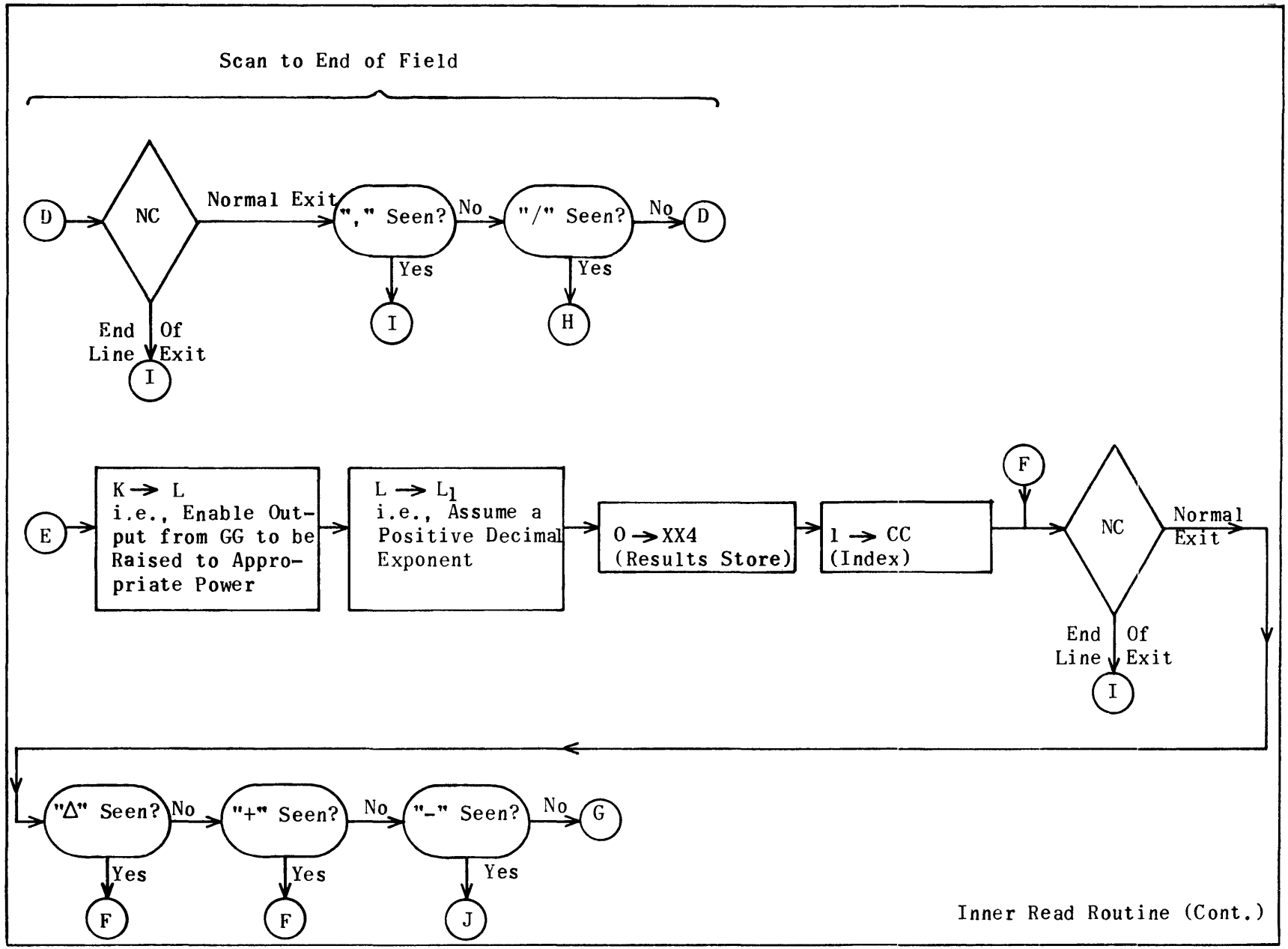
"End of Data" Procedure

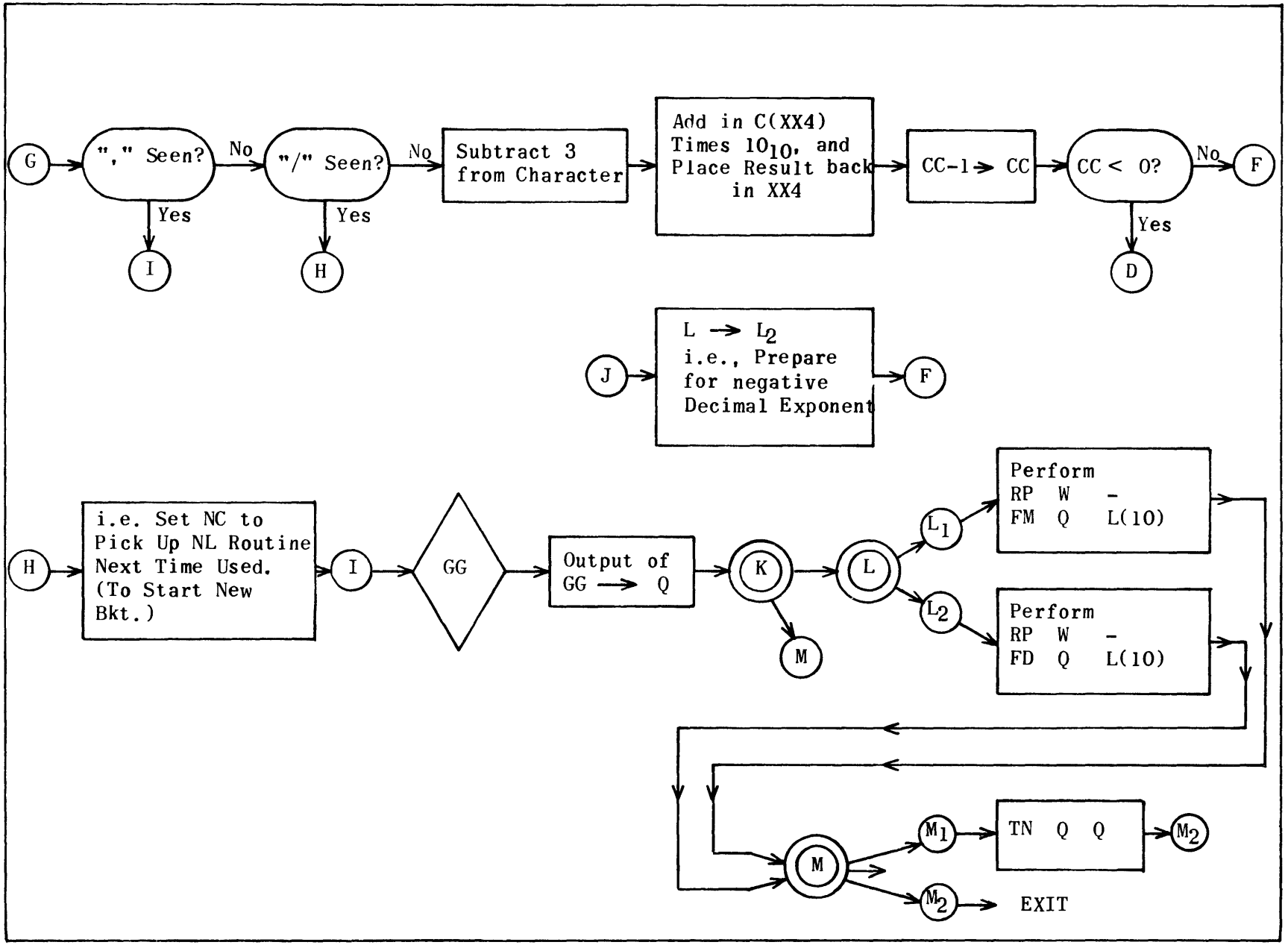
This example shows three sets of A, each of two blocks, followed by two sets of B, each of three blocks.



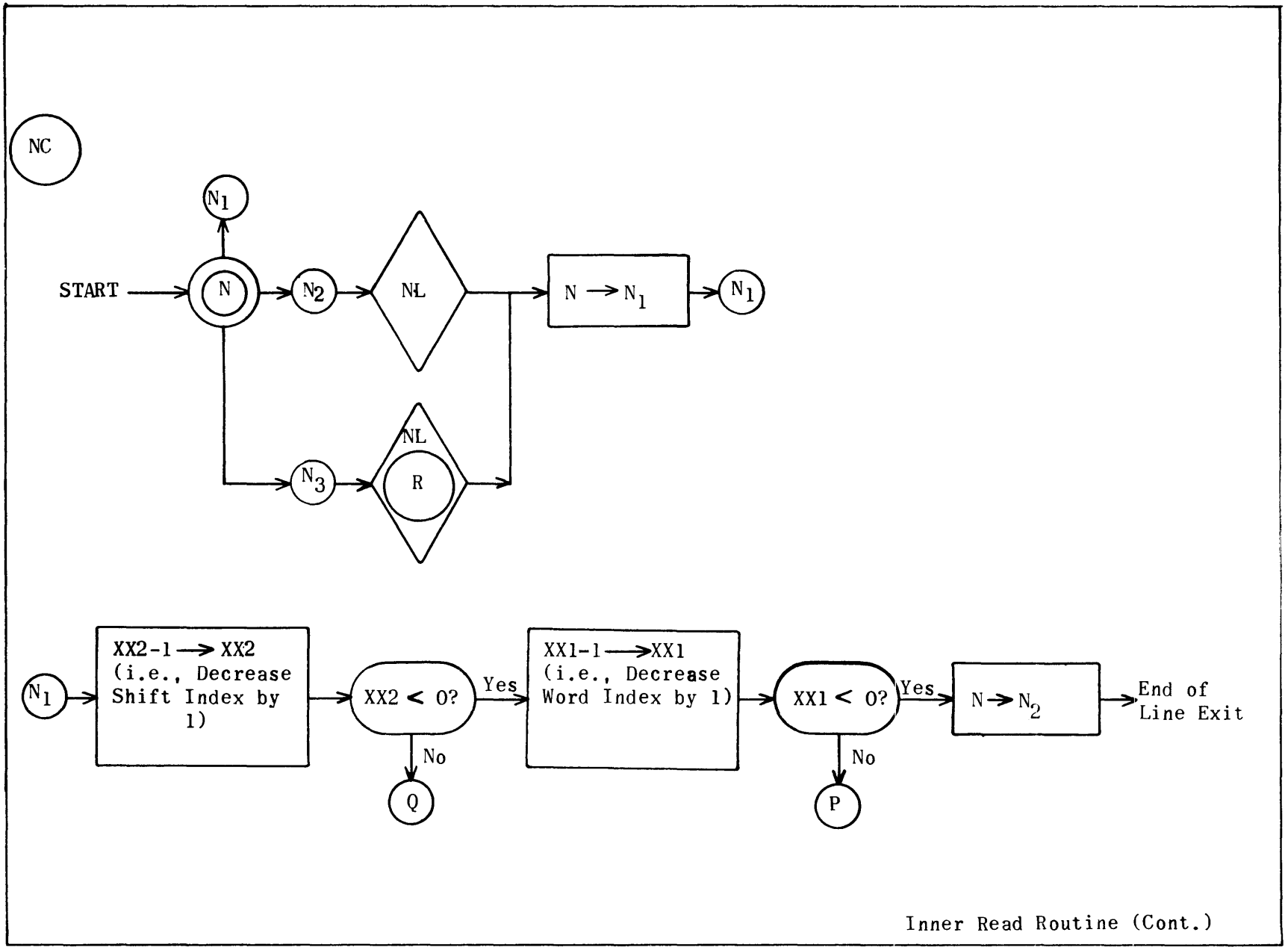

START
Entry 1
DATA EXTRACTION
Clear Input Lines to Routine GG (fl. pt.conv.)
Set CC → 11₁₀ (Index for Character Extraction)
Enable Routine GG Only.
K → M
M → M₂
Initialize ST Routine.
r → 4, Box 3 in (ST)
ST
S
20₁₀ → XX. (i.e. Ignore 1st Blockette of any One Data Set)
N → N₃ (Set NC to pick up NL for Initialization
A
Initialization for Data Extraction
Entry 2
TAPE POSITIONING
Inner Read Routine


A
NC
Normal Exit
End Line Of Exit
A
"Δ" Seen?
No
Yes
A
"+" Seen?
No
Yes
B
"-" Seen?
No
C
Yes
M → M1
i.e., Enable Complementation of Output from Conversion
B
NC
Normal Exit
End Line Of Exit
I
"Δ" Seen?
No
Yes
B
C
"E" Seen?
No
Yes
E
"," Seen?
No
Yes
I
"/" Seen?
No
Yes
H
ST
CC-1 → CC
(Countdown on Index)
CC < 0 ?
No
Yes
D
B
Inner Read Routine (Cont.)

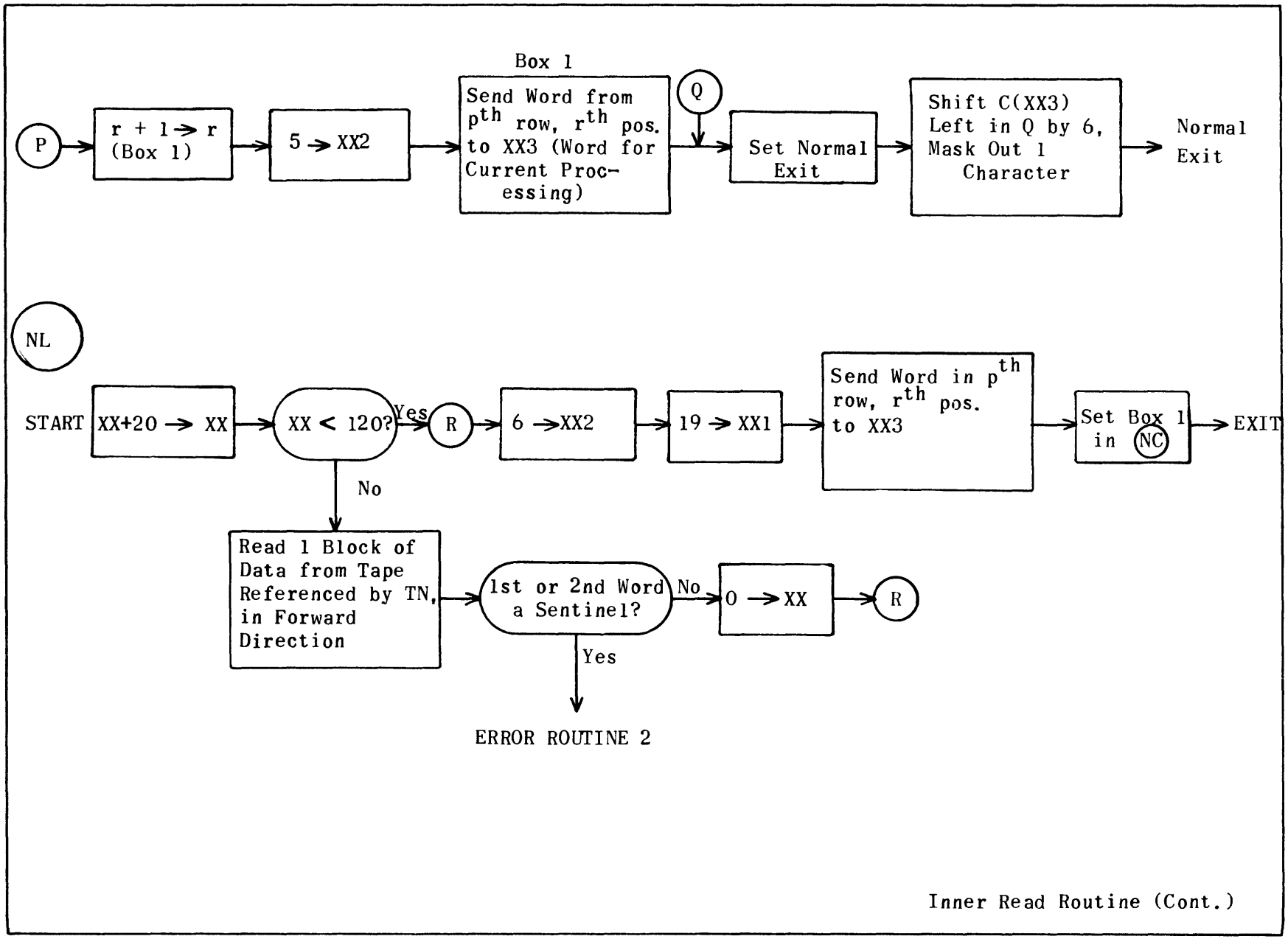

Scan to End of Field
D
NC
Normal Exit
"," Seen?
No
"/" Seen?
No
D
Yes
Yes
End Of Line Exit
I
I
H
E
K → L
i.e., Enable Output from GG to be Raised to Appropriate Power
L → L1
i.e., Assume a Positive Decimal Exponent
0 → XX4
(Results Store)
1 → CC
(Index)
F
NC
Normal Exit
End Of Line Exit
I
"Δ" Seen?
No
"+" Seen?
No
"-" Seen?
No
G
Yes
Yes
Yes
F
F
J
Inner Read Routine (Cont.)



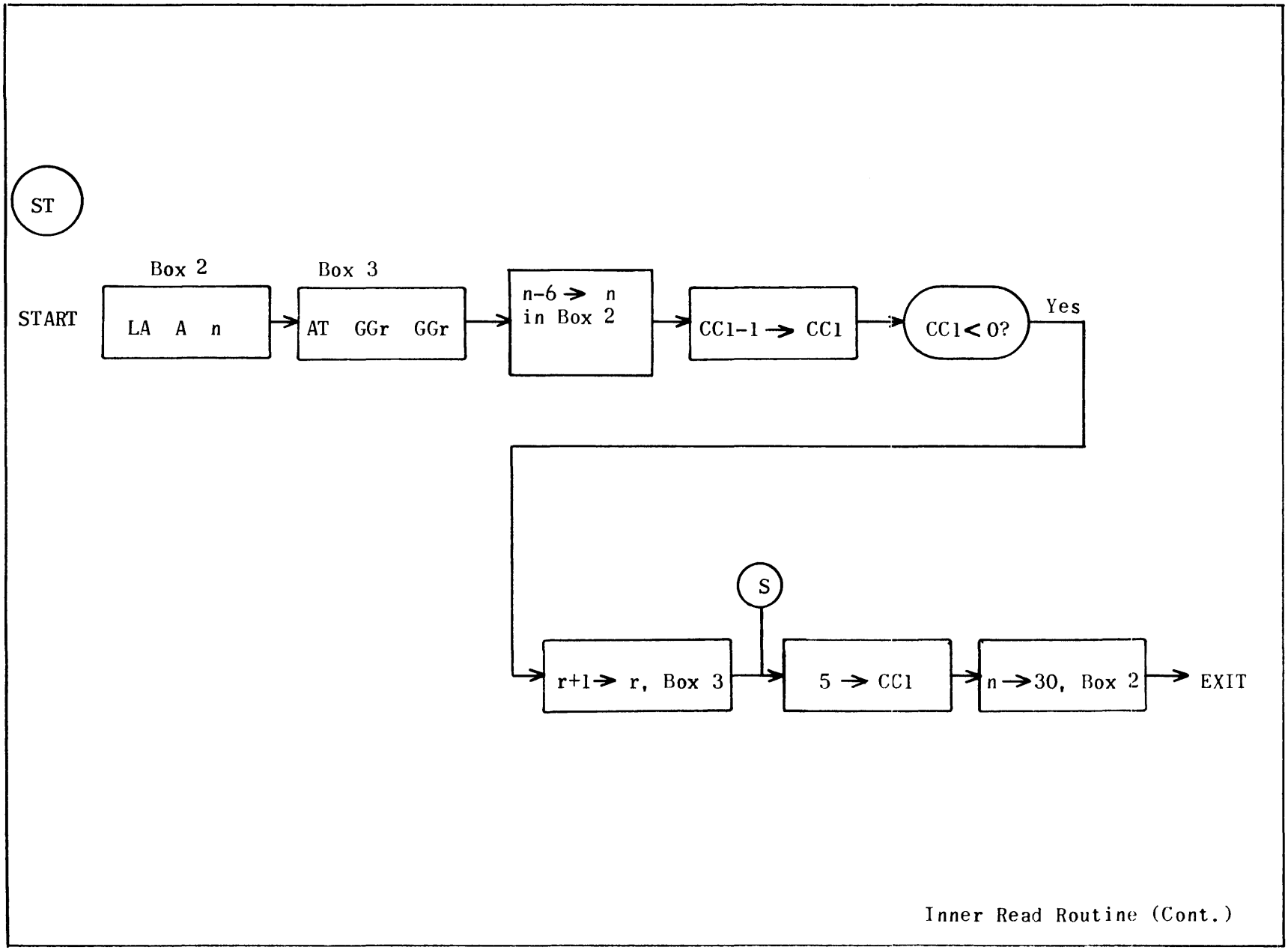
Inner Read Routine (Cont.)



Inner Read Routine (Cont.)



Inner Read Routine (Cont.)



Inner Read Routine (Cont.)

"Inner" Read Routine (50100)

```
      IA    NI
      0     50100   557       Call word; no. of lines prelude & routine
      0     0       473       No. of lines for address modification
      0     0       56        No. of unmodifiable constants
      77    77777   77775     No. of inputs (-2 for Processor)
      0     0       0         No. of outputs
      54    30242   75450     Name (READRN)
INO   MJ    0       30000     Exit
  1   0     30000   30000     Input line
  2   MJ    0       PS        Entry for tape positioning
  3   TP    CF13    GG4   }   Entry for data extraction
  4   TP    CF13    GG5   }   Clear GG input lines
  5   TP    CF3     CC        1 to ST word index (2 input lines only)
  6   TV    DR56    DR53  }   Set switches for GG conversion only
  7   TV    DR62    DR60  }
 10   TP    ST14    ST1   }   Initialize ST
 11   RJ    ST12    ST10  }
DR0   RJ    BM27    BM        Obtain one character
  1   EJ    CF3     BM        Discard leading ∧
  2   EJ    CF22    BM        Discard E
  3   EJ    CF20    DR6       If, } end line
  4   EJ    CF23    DR6       If; }
  5   MJ    0       DR11      O.K. - . or digit
  6   TP    CF13    XX    }   Start new line
  7   TP    CF13    XX1   }
 10   MJ    0       BM        Continue search for start
 11   TJ    CF1     DR21      Start. - sign?
 12   RJ    ST12    ST        No, digit, store
 13   RJ    BM27    BM        Get next character
 14   EJ    CF3     DR51      If ∧, out
 15   EJ    CF22    DR25      If E, go process exp.
 16   EJ    CF20    DR47      If, } end line, out
 17   EJ    CF23    DR47      If; }
 20   MJ    0       DR12      Otherwise, go store
 21   TV    DR64    DR60      - seen, enable complementation
 22   RJ    BM27    BM    }   Discard any ∧ between - and digits
 23   EJ    CF3     BM    }
 24   MJ    0       DR15
 25   TV    DR63    DR53      E Seen, enable exponentiation
 26   TP    DR65    DR57      Assume +ve exponent
 27   TP    CF13    XX4       Zeroize exponent store
 30   RJ    BM27    BM    }   Discard ∧ between E and first exp.
 31   EJ    CF3     BM    }     character and - and 1st exp.digit
 32   TJ    CF1     DR45      - sign?
 33   ST    CF1     Q     )
 34   SP    XX4     2     )   No, form exponent
 35   SA    XX4     1     )
 36   AT    Q       XX4   )
 37   RJ    BM27    BM        Get next character
```

```
   40   TJ   CF1      DR51        If Λ or -  , out
   41   EJ   CF20     DR47        If, ) clear, out
   42   EJ   CF23     DR47        If; )
   43   EJ   CF21     DR51        If ., out
   44   MJ   0        DR33        Otherwise, back
   45   TP   DR66     DR57        - exp. seen.  Set FD
   46   MJ   0        DR30
   47   TP   CF13     XX          Zeroize shift index
   50   TP   CF13     XX1         Zeroize word index
   51   RJ   GG2      GG          Go convert
   52   TP   GG3      Q           Output ⟶ Q
   53   MJ   0        [30000]
   54   LA   XX4      17          [Exp] ⟶ Au
   55   TU   A        DR56
   56   RP   [0]      DR60 )
   57   [0   30000    30000] )    Apply exponent
   60   MJ   0        [30000]
   61   TN   Q        Q
   62   MJ   0        IN          And out
   63   0    0        DR54
   64   0    0        DR61
   65   FM   Q        CF25
   66   FD   Q        CF25
BM 0    IJ   XX       BM21        Jump on shift index
    1   IJ   XX1      BM16        Jump on word index
    2   IJ   XX2      BM13        Jump on line index
    3   SP   TN       17
    4   AT   CF37     GT3 )       Read one block (forward)
    5   RJ   GT2      GT  )
    6   TP   CF26     A           Z —— Z ⟶ A
    7   EJ   BF       EP5 )       If Sentinel, go to probable error routine.
   10   EJ   BF1      EP5 )         (It will be necesaary to back up tape)
   11   TP   CF2      XX2         5 to line index
   12   TP   BM21     XX3         Initialize to BF
   13   TU   XX3      BM16
   14   RA   XX3      CF17        Inc. by 20 (u)
   15   TP   CF23     XX1         19₁₀ ⟶ word index
   16   TP   [30000]  BF
   17   RA   BM16     CF4         Inc. by 1 (u)
   20   TP   CF2      XX          5 ⟶ shift index
   21   LQ   BF       6   )       Extract 1 frame
   22   QT   CF       A   )
   23   EJ   CF13     BM          If 00 (i) discard
   24   TJ   CF24     BM27        If < 15₈, O.K.
   25   RP   20004    BM  )       ≥ 15 - check for permitted , . ; E
   26   EJ   CF20     BM27)         otherwise reject
   27   MJ   0        [30000]     Exit
ST  0   [0   30000    (30000)]    LA  A  n  (or MJ  0  ST12)
    1   [0   (30000)  (30000)]    AT  GGr  GGr
    2   RS   ST       CF27        Decrease shift by 6
    3   IJ   CC1      ST12        Count down on character index
    4   IJ   CC       ST7         Insure not more than two words of input
```

```
          5   TP   ST6     ST      Yes, set entrance skip
          6   MJ   0       ST12    And out
          7   RA   ST1     CF30    Augment u and v, to input 2nd line
         10   TP   CF2     CC1     Reset index to 5
         11   TP   ST13    ST      Reset 1st instruction
         12   MJ   0       [30000] Exit
         13   LA   A       36      Constant
         14   AT   GG4     GG4     Constant
       PS 0   TU   FX1     SC      Set up RP
          1   TU   FX      SC1     Set up EJ
          2   TP   IN1     A       Variable required
          3   RJ   SC7     SC
          4   TP   A       CC      Store synonym of variable requested
          5   TP   SC6     CC1     Save order containing syn. address (TP L
                                     (syn ) A)
          6   LQ   Q       17
          7   TU   Q       SC      Set up RP for continued scan
         10   TU   SC6     SC1     Set up EJ for continued scan
         11   TP   CF33    A       "∧ ∧ TAPE" ⟶ A
         12   RJ   SC7     SC
         13   TP   A       TN      Note tape number
         14   RA   SC6     CF4     Form address of indicator
         15   TU   A       PS25    And set up order
         16   SS   CC1     0
         17   SA   CF41    0       Add 17774 in "u", forming jn for repeat
         20   TU   A       PS31
         21   RA   CC1     PS35    Increment by 2 in "u" field
         22   TU   A       PS32
         23   LQ   PS25    Q25
         24   TV   Q       TB6     Set up subroutine with address of indicator
         25   TP   [30000] A       Indicator syn.⟶A
         26   EJ   CF34    PS33    =  ∧ DATA ∧ ? (i.e., at start of tape?)
         27   EJ   CC      PS55    = Required Synonym?
         30   EJ   CF35    PS36    = ∧ END ∧ 0? (i.e., at end of tape?)
         31   RP   [0]     PS33    None of these.  Scan from L(Syn)⟶L(TAPE)
         32   EJ   [30000] PS36    If indicator scan, move backward; if not,
                                     forward
FORWARD  33   RJ   MF6     MF      Go read forward
         34   EJ   CF35    EP      Return here if Sentinel seen, but incorrect
         35   MJ   2       PS33      one.  So test for end of tape, & if not
                                     carry on
         36   SP   TN      17  }
         37   AT   CF40    GT3 }   Read 1 block backwards
         40   RJ   GT2     GT      Read 1 block backwards
         41   RJ   TB11    TB
         42   MJ   0       PS36    Return here if no Sentinel seen
         43   MJ   0       PS46    Return here if correct syn seen, so go
                                     test Set No.
         44   EJ   CF34    EP      Incorrect syn, check not start of tape
         45   MJ   0       PS36
         46   SP   BF3     0   }
         47   EJ   CF32    PS51}   Test for 1st set
```

```
        50   MJ   0        PS36      Not 1st set, so go back and read more
        51   SP   TN       14        Now move 1 block forward
        52   SA   CF36     0         Add in EF code
        53   EF   0        A         And execute
        54   MJ   0        PS57      Go to normal exit (modified)
INDIC=  55   RJ   MF6      MF        Read forward until next Sentinel
 SYN    56   MJ   0        EP13      Incorrect syn;.'. end of data,.'. abnormal
                                       exit
        57   RA   IN       CF3  }    Normal (modified) exit
        60   TP   CF13     XX   }    Clear shift index
        61   TP   CF13     XX1       Clear word index
        62   TP   CF2      XX2       5 ⟶ line index
        63   TP   CF52     XX3       Erasable address store to BF24
        64   MJ   0        IN        And out
     SC 0    RP   [0]      EP   }
           1  EJ   [30000]  SC2  }    Scan index as specified
           2  SN   Q        17   )
           3  SA   FX1      0    |
           4  SA   FX       0    }
           5  TU   A        SC6  |    And extract word following that sought
           6  TP   [30000]  A    )
           7  MJ   0        [30000]   Exit
     MF 0    SP   TN       17   )
           1  AT   CF37     GT3  }    Read in one block
           2  RJ   GT2      GT   )
           3  RJ   TB11     TB        Go perform tests
           4  MJ   0        MF        No Z ——Z Sentinel, read more
           5  MJ   0        PS57      Required synonym, go modify exit (as usual)
           6  MJ   0        [30000]   Incorrect synonym - exit
     TB 0    TP   CF26     A         Z——Z ⟶ A
           1  RP   20002    TB11 }    If either of 1st two words all Z's,
           2  EJ   BF       TB3  }      accept test
           3  RA   TB11     CF3
           4  AT   CF3      TB10
           5  TP   BF2      A         Examine syn. of this data, and place in
           6  TP   A        [30000]     indicator
           7  EJ   CC       TB11      = Required synonym?
          10  [0   30000    30000]    Exit 3
          11  MJ   0        [30000]   Exits 1 and 2
     EP 0    TP   EP4      Q    )
           1  RJ   CF55     CF53 |    Data cannot be found on tape
           2  TP   IN1      Q    }    (DATA INDEX ERROR)
           3  MS   0        PS   |    (N.B. - tape may be mounted at this stage)
           4  0    CF42     4    )
           5  RA   IN       CF1       Inadequate data.  Set abnormal exit
           6  TP   EP12     Q    }
           7  RJ   CF55     CF53 }    Print out
          10  TP   IN1      Q         Note data
          11  MS   0        EP13
          12  0    CF46     4
          13  SP   TN       14   )
          14  SA   CF31     0    }    Back up tape 1 block
          15  EF   0        A    )
          16  MJ   0        IN        And exit
```

```
          .          .
          .          .
          .          .
```

XS3 to Floating Point Conversion Routine GGO-CF16 used here (See Section III, 3-Translation Subroutines)

```
CF 17  0    24      0
   20  0    0       21        ,
   21  0    0       22        .    XS3 codes
   22  0    0       30        E
   23  0    0       23        ;        (Also 19_10)
   24  0    0       15
   25  DE   1.0Δ1ΔF           10 (floating pt)
   26  74   74747   47474     Z——Z
   27  0    0       6
   30  0    1       1
   31  02   14      1         Move backward one block
   32  01   65662   45466     Δ START
   33  01   66245   23001     Δ TAPE Δ
   34  01   27246   62401     Δ DATA Δ
   35  01   30502   70151     Δ END Δ 0
   36  02   4       1         Move forward one block
   37  50   00100   BF        Read forward
   40  60   00100   BF167     Read backward
   41  0    17774   0
   42  45   47223   00130     cr ↑ DATA
   43  04   14062   22027     Δ INDEX          PS section error
   44  04   20121   20312     Δ ERROR
   45  44   24202   00435      . SEE Δ Q
   46  45   47140   63022    cr ↑ INAD
   47  20   35343   00120     EQUATE
   50  04   22300   13044     Δ DATA.
   51  04   24202   00435     Δ SEE Δ Q
   52  0    BF24    0
   53  TP   Q       PR3
   54  RJ   PR2     PR
   55  MJ   0       [30000]
       CA   NI557
```

Integer Conversion & Print-Out Routine (50112)

Purpose

To convert to decimal, and printout on-line a positive binary fixed-point integer, with a carriage return depending upon the contents of a specified index.

Calling Sequence

```
α    TP   L(Integer)    IC3
+1   TP   L(Parameter)  IC4
+2   RJ   IC2           IC
+3     Control returned here.
```

The parameter is the same as "Parameter 2", described in the Flex-Print routine.

Storage

71_8 orders and constants
2 erasable locations

A and Q destroyed

Integer Conversion and Print-Out Routine (50112)

```
        IA    CI
        0     50112    77          Call word; no. of lines prelude + routine
        0     0        42          No. of lines for address modification
        0     0        27          No. of unmodifiable constants
        0     0        2           No. of inputs
        0     0        0           No. of outputs
        34    50662    67066       Name (INTCVT)
IC 0    MJ    0        IC5         Entrance
   1    RJ    30000    30000
   2    MJ    0        30000       Exit
   3    0     30000    30000       Binary Integer Input
   4    0     30000    30000       Line Index Address
   5    PR    0        IC22        Switch to lower case
   6    TU    IC4      IC27 }      Set line index address
   7    TV    IC4      IC31 }
  10    TP    IC3      A56
  11    SJ    IC13     IC12        Negative?
  12    ZJ    IC14     IC34        Zero?
  13    PR    0        IC10        Print minus sign
  14    TM    A        IC72        Non-zero, send positive integer to working
                                     space
  15    TP    IC53     IC71        Set divide index to 10 decimal
  16    TP    IC41     IC25        Set switch to suppress zeros
  17    TU    IC16     IC22        Initialize divide
  20    RA    IC22     IC70        Prepare for next power of 10
  21    TP    IC72     A45
  22    DV    30000    Q57
  23    TP    A        IC72        Store remainder
  24    SP    Q        0           Quotient ⟶A
  25    0     30000    30000       ZJ  IC35  IC32; AT IC37  IC26
  26    0     30000    30000       PR  0  IC55 +n
  27    IJ    30000    IC32        Test for end of line
  30    PR    0        IC21        Carriage return
  31    TP    IC67     30000       Reset index
  32    IJ    IC71     IC20
  33    MJ    0        IC2         To exit
  34    TP    A        IC71        Set divide index to zero
  35    TP    IC40     IC25 }      Set switch to commence printing
  36    MJ    0        IC25 }
  37    PR    0        IC55
  40    AT    IC37     IC26
  41    ZJ    IC35     IC32
  42    11    24027    62000       10^10
  43    0     73465    45000       10^9
  44    0     5753     60400       10^8
  45    0     461      13200       10^7
  46    0     36       41100       10^6
  47    0     3        03240       10^5
```

50	0	0	23420	10^4	
51	0	0	1750	10^3	
52	0	0	144	10^2	
53	0	0	12	10^1	
54	0	0	1	10^0	
55	0	0	37	0	Flex Codes
56	0	0	52	1	
57	0	0	74	2	
60	0	0	70	3	
61	0	0	64	4	
62	0	0	62	5	
63	0	0	66	6	
64	0	0	72	7	
65	0	0	60	8	
66	0	0	33	9	
67	0	0	117		
70	0	1	0		
71	CA	CI77			Temps. and index
72					

3. UNICODE SYSTEM TAPE PACKAGE

3. UNICODE SYSTEM TAPE PACKAGE

This package makes possible a reproduction of a UNICODE Master Tape and a quick comparison by sections with the original to check the accuracy of the reproduction. Also an octal copy of the UNICODE Master Tape can be taken for listing by the High-Speed Printer. Corrections and/or alterations to the UNICODE System are facilitated by its use.

There are two versions of this set of subroutines; one for the 1103A and one for the 1105. High-speed memory usage is confined to the first core bank of 4096 addresses.

The main routines to perform the functions mentioned above are described in this write-up as well as a number of subsidiary routines which are used by the main routines and are also usable by an operator from the console.

A list of these routines follows:

1) Reproduce UNICODE Master Tape
2) Compare Two UNICODE Master Tapes
3) High-Speed Printer Octal Listing of UNICODE Master Tape
4) Read Magnetic Tape
5) Write Magnetic Tape
6) Bioctal Paper Tape Loader
7) Flex Paper Tape Loader
8) High-Speed Printer Octal Dump on Magnetic Tape
9) Read to Q
10) Write From Q
11) Changed-Word Post Mortem

The package is stored as follows:

40200 - 40330 40400 - 40435	Reproduce UNICODE Master Tape
40500 - 40520	High-Speed Printer Octal Listing of UNICODE Master Tape
40600 - 40746	Compare Two UNICODE Master Tapes
41000 - 41430	Read, Write Magnetic Tape
44600 - 45741	Routines 6-11

The Read and Write Magnetic Tape routines bootstrap themselves into core addresses 6230-6661. They use a read-write buffer located from 5270-6227. The core is not restored after their use.

Compilation with UNICODE or running an Object Program destroys this package.

On the Master Tape, UNICODE has been divided logically into 23 sections. Thus a set of 23 parameters is used to read or write a UNICODE Master Tape. Following is a table giving the sections of UNICODE, their length in blocks (octal), and their loading addresses in the core or drum.

Section Number	Section Title	Number Blocks (Octal)	Addresses
1	UNICODE Sentinel Blocks	2	7230-7607
2	Merge	16	35-3134
3	Set-up Translation	2	7230-7607
4	Translation Sub Prints & Translators	133	50333-75600
5	Region FC (Flex Codes)	1	40001-40100
6	Translation Subroutines	17	21-3316
7	Dimension Translators No. 1 and No. 2	4	4400-5267
8	Fixed Library Catalog	1	7230-7417
9	Set-up Generation	1	7230-7417
10	Generators	66	50212-64731
11	Generation Subroutines	6	537-2043
12	Set-up Segmentation	1	7230-7417
13	Segmentor	11	674-2726
14	Op File I for Fixed Library	1	7230-7417
15	Set-up Allocation	1	7230-7417
16	Allocation	7	674-2403
17	Set-up Initialization	1	7230-7417
18	Initialization Generator	16	2000-5057
19	Set-up Processor	1	7230-7417
20	Processor	6	653-2172
21	Fixed Library Routines	17	10-3417
22	Set-up Listing	1	7230-7417
23	Listing Routine	12	653-3027

Following is a brief description of the routines and their use:

1. Reproduce UNICODE Master Tapes

The UNICODE Master Tape should be loaded on Uniservo 2 (Tape Control Unit 2, if 1105). A blank tape should be loaded on Uniservo 1 (TCU2, if 1105). If MS1 is on, a stop occurs after each typing of a section number. If MS1 is off, the tape is reproduced without stopping. Start with PAK at 40400.

Uniservo 2 is read according to the set of addresses of the first section and the figure 1 is typed. If MS 1 is set, the computer stops with PAK at 40414. If this section is to be changed, Flex, bioctal, or magnetic-tape inputs should be loaded at this point. If this loading does not alter address 40000, Master Clear and Start. If address 40000 has been altered, Master Clear, set PAK at 40414, and Start. The altered contents of section 1 is then written on Uniservo 1; Uniservo 2 is read according to the parameter of the second section; the figure 2 is typed; and the computer stops again on MS1 with PAK at 40414. Now corrections or changes may be made in section 2 before restarting as explained above. This procedure continues in a like manner through the 23 sections of UNICODE. The final stop occurs with PAK at 40400. Tapes 1 and 2 are automatically rewound.

2. Compare Two UNICODE Master Tapes

Put the UNICODE Master Tapes to be compared on Uniservos 1 and 2. Set PAK at 40600 and start.

The routine compares the two tapes and prints out any differences on the typewriter in changed-word post mortem format. If the tapes are identical, only the section numbers are typed. When comparison is finished, both tapes are rewound and the computer stops at 40600. This routine should be run after every updating to see if the new tape has been properly reproduced.

3. High-Speed Printer Octal Listing of UNICODE Master Tape

Put the UNICODE Master Tape on Uniservo 1 (TCU2, if on 1105); put a blank tape on Uniservo 3 (TCU2, if on 1105). Set PAK at 40500 and start.

The output on Uniservo 3 can be later run off on the High-Speed Printer to secure the desired listing. Both tapes must be rewound from the console after the computer stops at 40500.

4. Read Magnetic Tape

Op	u	v
Q = xxy yyy	First Address	Last Address

where x is any binary digit and yyyy is the binary representation of the Uniservo number. If on the 1105, use a Uniservo under Tape Control Unit 2.

With Q set as desired, set PAK at 41152 and start. References from within a program may be made by loading Q as desired and using the instruction, 37 41217 41152.

5. Write Magnetic Tape

Op	u	v
Q = Zxy yyy	First Address	Last Address

where Z = 1,
x is any binary digit,
yyyy is the binary representation of the Uniservo number. Use Tape Control Unit 2 if on the 1105.

Set PAK at 41000 and start. This routine may also be referenced from a program by the instruction, 37 41036 41000.

6. Bioctal Paper Tape Loader

Put the tape in the Ferranti reader, turn the reader on, set PAK at 44601, and start. The routine can also be used by the program reference, 37 44636 44601.

7. Flex Paper Tape Loader

Put the tape in the Ferranti reader, turn the reader on, set PAK at 44602, and start. The routine can also be used by the program reference, 37 44636 44602.

8. High-Speed Printer Octal Dump on Magnetic Tape

Op	u	v
Q = zxy yyy	First Address	Last Address

where z = 0 means use of Tape Control Unit 1

z = 1 means use of TCU2 if on the 1105. If on the 1103A, z must be set at 1 for any use of the routine.

x = 0 means no rewind of tape

x = 1 means rewind with interlock and printer stop.

yyyy is the binary representation of the Uniservo number.

Set PAK at 44625 and start. The output tape, when listed on the High-Speed Printer, will give an octal dump of the storage area between the addresses specified in u and v. This routine may be referenced from a program by the instruction, 37 44636 44625.

9. Read to Q

Put address which is to be read in A_v, set PAK at 44603, and start. The content of the address appears in Q and the address in A_v is increased by 1 for a read of the next address, if desired. The computer stops on 44603.

10. Write from Q

Put address which is to be filled in A_v, put content that is desired in Q, set PAK at 44604, and start. The content of Q is stored in the address specified and the address in A_v is increased by 1 for another write in the next address, if desired. The computer stops on 44604.

11. Changed-Word Post Mortem

	Op	u	v
Q =	XX	YYYYY	ZZZZZ

where XX = 00 gives punched paper tape output.

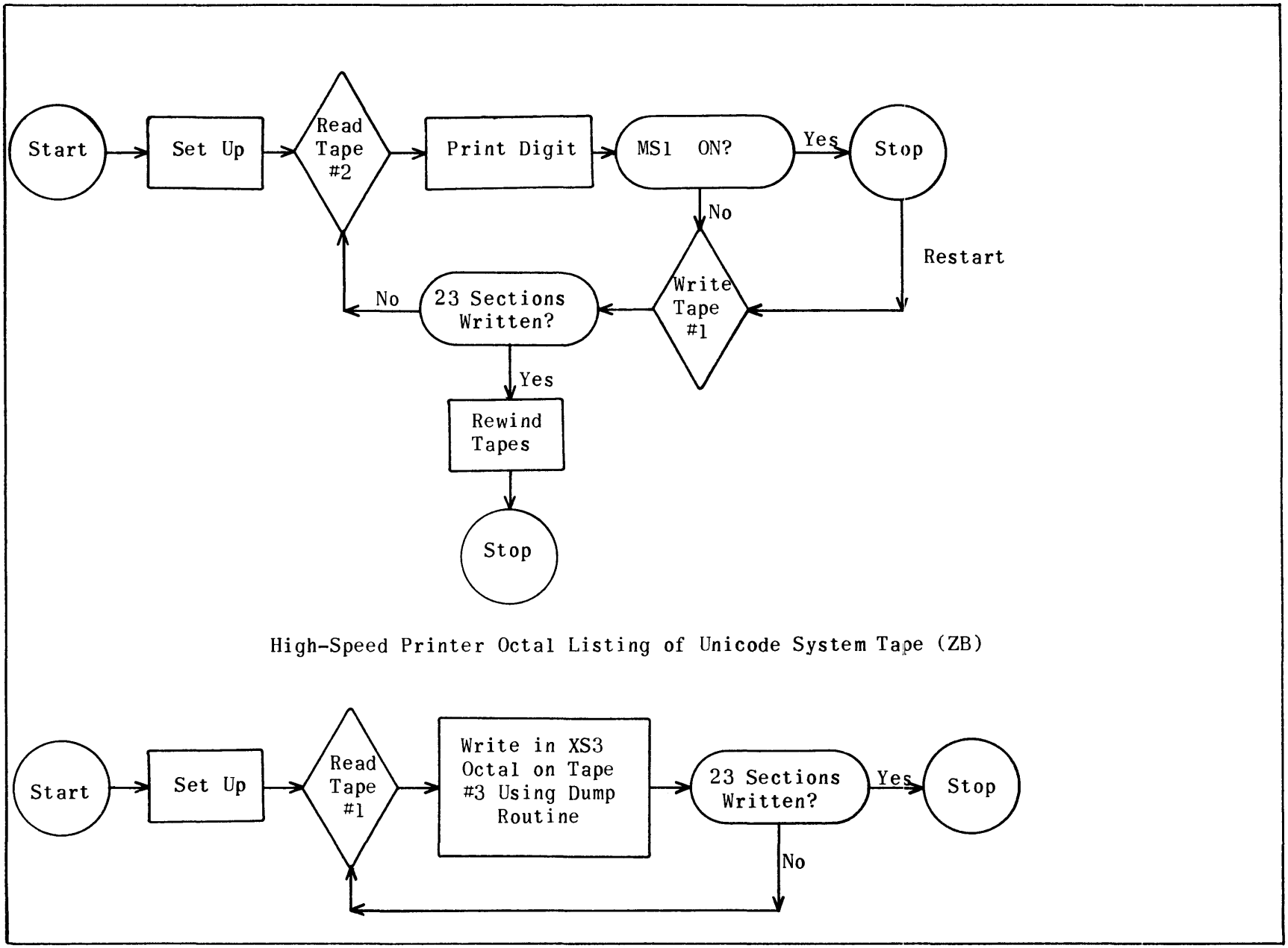
XX ≠ 00 gives typed output.

YYYYY is the first address of the area to be compared.

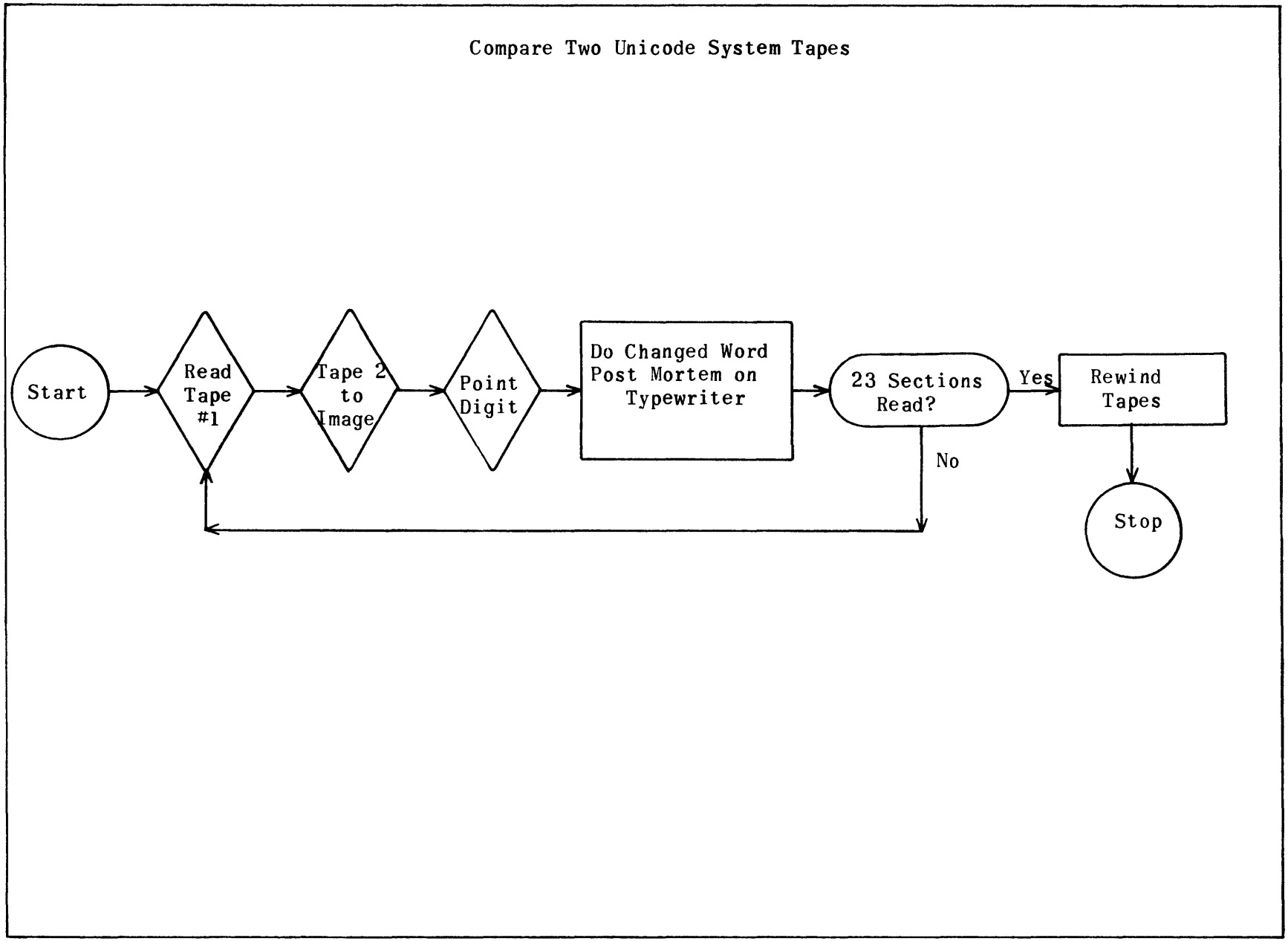
ZZZZZ is the first address of the image.

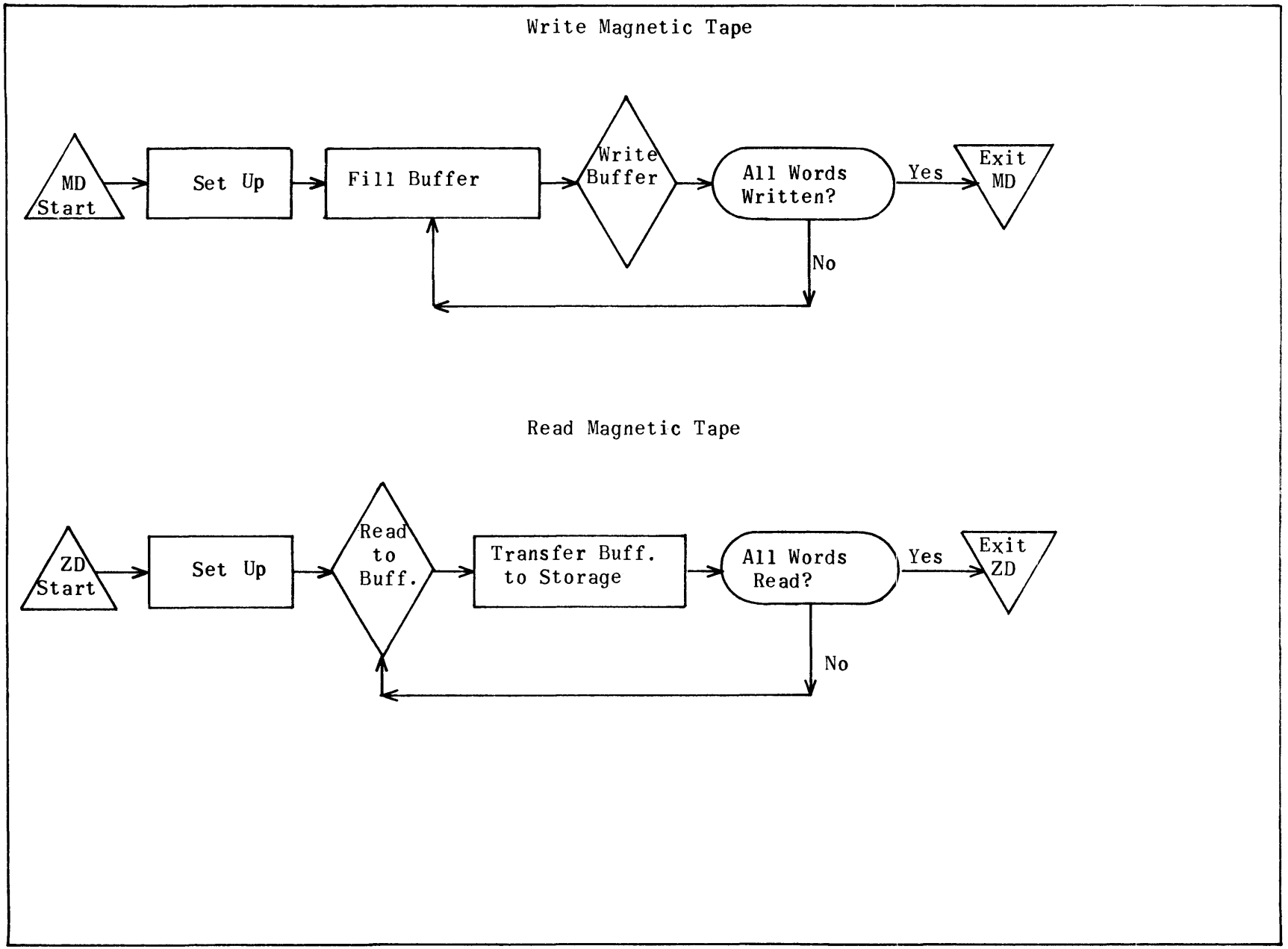
In A_v should be put the number of words to be compared. Set PAK at 44613 and start. The routine may be referenced from a program by the instruction, 37 44636 44613.

Reproduce System Tape (KK)



High-Speed Printer Octal Listing of Unicode System Tape (ZB)


Compare Two Unicode System Tapes
Start
Read Tape #1
Tape 2 to Image
Point Digit
Do Changed Word Post Mortem on Typewriter
23 Sections Read?
Yes
No
Rewind Tapes
Stop


Write Magnetic Tape
MD Start
Set Up
Fill Buffer
Write Buffer
All Words Written?
Yes
Exit MD
No
Read Magnetic Tape
ZD Start
Set Up
Read to Buff.
Transfer Buff. to Storage
All Words Read?
Yes
Exit ZD
No

Regions for System Tape Package

```
BA7230          BZ7607
CA35            CZ3134
EA50333         EZ75600
FA40001         FZ40101
GA21            GZ3316
HA4400          HZ5267
JA7230          JZ7417
KA50212         KZ64731
LA537           LZ2043
NA674           NZ2726
RA674           RZ2403
TA2000          TZ5057
VA653           VZ2172
WA10            WZ3417
YA653           YZ3027

GG40200
    22
PR40222
    10
CE40232
    10
CC40242
    40
FF40302
    27
KK40400
    31
CD40431
     5
ZB40500
    14
CP40514
     5
HH40600
   102
HJ40702
    15
VV40717
    30
 40747
```

```
ZD41152     Read
MD41000     Write
RS40000     Restart
EN44600     Other Routines
```

System Tape Package

(Parameters)

UNICODE MASTER TAPE SECTIONS		IA	FF		
1	0	41	BA	BZ	1st & 2nd blks.
2	1	41	CA	CZ	Merge
3	2	41	BA	BZ	Two blks. after merge set-up S.O.S.
4	3	41	EA	EZ	S.O. & S.O.S. prints
5	4	41	FA	FZ	FC
6	5	41	GA	GZ	S.O.S.
7	6	41	HA	HZ	Dim. #1 & #2
8	7	41	JA	JZ	Lib. Catalog
9	10	41	JA	JZ	Set-up Generation
10	11	41	KA	KZ	Generation
11	12	41	LA	LZ	Generation Subs
12	13	41	JA	JZ	Seg. Set-up
13	14	41	NA	NZ	Segmentation
14	15	41	JA	JZ	Op. file 1 for Fixed Library
15	16	41	JA	JZ	Set-up Allocation
16	17	41	RA	RZ	Allocation
17	20	41	JA	JZ	Set-up Initialization
18	21	41	TA	TZ	Initialization Generation
19	22	41	JA	JZ	Processor Set-up
20	23	41	VA	VZ	Processor
21	24	41	WA	WZ	Fixed Library routines
22	25	41	JA	JZ	Listing Set-up
23	26	41	YA	YZ	Listing
		CA	FF27		

Produce Master Tape

```
    IA   GG
 0  TU   CC       GG5          Set to print #1
 1  TU   GG17     GG11         Set address of 1st par.
 2  TP   CE4      CE3          Set index
 3  TV   CE5      GG10         Set one shot
 4  TP   CE6      RS           Set restart address (40000)
 5  TP   (30000)  CE7          No. to print
 6  RJ   PR       PR1          Print number
 7  MS   0        EN2          Stop for load
10  RJ   GG10     (30000)      One shot              other params ↓
11  TP   (30000)  Q        }
12  RJ   MD36     MD1      }   Write tape
13  RA   GG11     CE       }
14  RA   GG5      CE       }   Modify
15  IJ   CE3      GG4          n parameters
16  MS   0        GG
17  TP   FF       Q
20  RJ   MD36     MD
21  MJ   0        GG13
    CA   GG22
```

Print Digit

```
    IA   PR
 0  MJ   0        (30000)      Exit
 1  RP   2        PR3
 2  PR   0        CE2
 3  PR   0        CE1          Lower carriage
 4  SP   CE7      52       }
 5  PR   0        A        }   Print word
 6  SS   A        6        }
 7  ZJ   PR5      PR
    CA   PR10
```

Constants

```
    IA   CE
 0  0    1    0
 1  0    0    57          Shift down
 2  0    0    45          Return carriage
 3  0    0    0           Index
 4  0    0    26          Set index
 5  0    0    GG17
 6  MJ   0    GG10        Set restart
 7  0    0    0           No. to print
    CA   CE10
```

Constants

	IA	CC		
0	0	CC1	0	
1	52	0	0	Flex for 1
2	74	0	0	2
3	70	0	0	3
4	64	0	0	4
5	62	0	0	5
6	66	0	0	6
7	72	0	0	7
10	60	0	0	8
11	33	0	0	9
12	52	37000	0	10
13	52	52000	0	11
14	52	74000	0	12
15	52	70000	0	13
16	52	64000	0	14
17	52	62000	0	15
20	52	66000	0	16
21	52	72000	0	17
22	52	60000	0	18
23	52	33000	0	19
24	74	37000	0	20
25	74	52000	0	21
26	74	74000	0	22
27	74	70000	0	23
30	74	64000	0	24
31	74	62000	0	25
32	74	66000	0	26
33	74	72000	0	27
34	74	60000	0	28
35	74	33000	0	29
36	70	37000	0	30
37	70	52000	0	31
	CA	CC40		

Update System Tape or Reproduce System Tape

```
     IA   KK
 0   TU   CC        KK11          Set to print #1
 1   TU   KK26      KK6           Set address of 1st par.
 2   TP   CE4       CE3           Set index
 3   TV   CD        KK15          Set one shot
 4   MJ   0         KK5           Skip
 5   TP   CD1       RS            Set restart address
 6   TP   (30000)   Q     }       Set to read tape #2
 7   RA   Q         CD4   }
10   RJ   ZD45      ZD            Read tape
11   TP   (30000)   CE7           Set print number
12   RJ   PR        PR1           Print number
13   MS   10000     KK14          Stop for write
14   TU   KK6       KK16          Set write address
15   RJ   KK15      (30000)       One shot
16   TP   (30000)   Q     }       Write tape
17   RJ   MD36      MD1   }
20   RA   KK6       CE    }       Modify
21   RA   KK11      CE    }
22   IJ   CE3       KK5
23   EF   0         CD2           Rewind #1
24   EF   0         CD3           Rewind #2
25   MS   0         KK            Stop
26   TP   FF        Q
27   RJ   MD36      MD
30   MJ   0         KK20
     CA   KK31
```

Constants

```
     IA   CD
 0   0    0         KK26
 1   MJ   0         KK14
 2   2    200       10000          Rewind #1
 3   2    200       20000          Rewind #2
 4   01   0         0
     CA   CD5
```

Dump System Tape on Servo 3 TCU2

```
     IA   ZB
 0   TP   CP3      CP2          Set index
 1   TU   CP1      ZB3          Set par. address
 2   MJ   50000    ZB2          Test buffer.
 3   TP   (30000)  Q            Par. ⟶ Q
 4   RJ   ZD45     ZD           Read Uniservo 1
 5   TU   ZB3      ZB6 }        Par. ⟶ Q
 6   TP   (30000)  Q   }
 7   RA   Q        CP           Uniservo 3
10   RJ   EN36     EN25         Write Uniservo 3
11   RA   ZB3      CP4          Next parameter
12   IJ   CP2      ZB2          23-parameter index
13   MS   0        ZB           Stop
     CA   ZB14
```

Constants

```
     IA   CP
 0   2    0        0            TCU2 designator
 1   0    FF       0            1st parameter address
 2   0    0        0            Index
 3   0    0        26           Set index
 4   0    1        0            1 in u
     CA   CP5
```

Compare System Tapes

```
     IA   HH
 0   TU   CC       HH17      Set to print 1
 1   TU   VV27     HH3       Set 1st parameter
 2   TP   VV6      VV5       Set index
 3   TP   (30000)  Q
 4   TP   Q        VV        Set prog. address
 5   RJ   ZD45     ZD        Tape 1 ⟶ storage
 6   TU   HH3      HH7   ┐
 7   TP   (30000)  Q     ├   Set parameter in Q
10   RA   Q        VV2   ┘
11   LT   25       A         1st add. of image to A_V
12   TV   A        VV        Set image address
13   TP   Q        A         Last add. ⟶ A
14   ST   VV       A         Last - 1st ⟶ A_V
15   AT   VV4      VV1       +1 to set A parameter
16   RJ   ZD45     ZD        Read to image (No. 2)
17   TP   (30000)  CE7   ┐   Print number
20   RJ   PR       PR1   ┘
21   TP   VV       Q     ┐
22   TP   VV1      A     ├   Do CWPM
23   RJ   EN36     EN13  ┘
24   RA   HH3      VV23  ┐   Modify
25   RA   HH17     VV23  ┘
26   RJ   HH26     HH27      One shot
27   IJ   VV5      HH3       3 parameters
30   TP   FF3      Q     ┐   Read in S.O. from No. 1
31   RJ   ZD45     ZD    ┘
32   TP   CC4      CE7   ┐   Print 4
33   RJ   PR       PR1   ┘
34   TP   VV7      HJ2   ┐
35   TP   VV10     HJ3   ├   Compare S.O.
36   TP   VV17     HJ4   │
37   RJ   HJ       HJ1   ┘
40   TP   VV12     HJ2   ┐   Last of S.O.
41   RJ   HJ       HJ1   ┘
42   TP   FF4      Q     ┐   Read FC (#1)
43   RJ   ZD45     ZD    ┘
44   TP   CC5      CE7   ┐   Print 5
45   RJ   PR       PR1   ┘
46   TP   VV14     Q     ┐   Read FC (#2)
47   RJ   ZD45     ZD    ┘
50   TP   VV15     Q     ┐
51   TP   VV14     A     ├   Compare FC
52   RJ   EN36     EN13  ┘
53   RA   HH3      VV16  ┐   Set for S.O.S.
54   RA   HH17     VV16  ┘
55   TP   VV17     VV5       Set index
56   RJ   HH26     HH3   ┐   Up to generators
57   IJ   VV5      HH56  ┘
```

```
 60   TP   FF11     Q      }
 61   RJ   ZD45     ZD     }   Read generator (#1)
 62   TP   CC12     CE7    }
 63   RJ   PR       PR1    }   Print 10
 64   TP   VV7      HJ2    )
 65   TP   VV20     HJ3    )
 66   TP   VV4      HJ4    }   Compare gen.
 67   RJ   HJ       HJ1    )
 70   TP   VV22     HJ2    }
 71   RJ   HJ       HJ1    }   Last generator read
 72   RA   HH3      VV23   }
 73   RA   HH17     VV23   }   Skip generator parameters
 74   TP   VV24     VV5        Set index
 75   RJ   HH26     HH3    }
 76   IJ   VV5      HH75   }   Do rest of parameters
 77   EF   0        VV25       Rewind
100   EF   0        VV26       Rewind
101   MS   0        HH
      CA   HH102
```

```
      IA    VV
 0    0     0         0              Q for CWPM
 1    0     0         0              A for CWPM
 2    1     50000     50000          Set for image read
 3    0     0         77777          Mask
 4    0     0         1
 5    0     0         0              Index
 6    0     0         2              Set index
 7    2     1         5120           Read S.O. & gen.
10    1     50333     1              Compare S.O.
11    0     5120      0              For more reads
12    2     1         546            Last read of S.O.
13    0     0         0
14    2     1         100            Read FC
15    1     40001     1              Compare FC
16    0     2         0
17    0     0         3              Set index
20    1     50212     1              Compare gen.
21    0     0         0
22    2     1         2260           Last gen. read
23    0     1         0
24    0     0         14             Set index
25    2     200       10000 }        Rewinds
26    2     200       20000 }
27    0     FF        0
      CA    VV30

      IA    HJ
 0    MJ    0         30000          Exit
 1    MJ    0         HJ5            Start
 2    0     30000     30000          Read par. & comp. par. for A
 3    0     30000     30000          Compare par. for Q
 4    0     30000     30000          Index
 5    TP    HJ2       Q     }        Read S.O. or gen.
 6    RJ    ZD45      ZD    }
 7    TP    HJ3       Q     }
10    TP    HJ2       A     }        CWPM
11    RJ    EN36      EN13  }
12    RA    HJ3       VV11           Modify
13    IJ    HJ4       HJ5            N times
14    MJ    0         HJ             Exit
      CA    HJ15
```

Regions for Read-Write

Core (Coding for both 1103A and 1105 versions shown)

Core		Drum	
BS5267			
1			
BF5270		Drum	
740			
DD6230		MD41000	Write
40		40	
EE6270		ME41040	
14		14	
FF6304		MF41054	
5		5	
WB6311		MB41061	
25		25	
CR6336		MR41106	
5		5	
SU6343		MU41113	
4		4	
CC6347		MC41117	
20		20	
BG6367		MG41137	
13		13	
ZA6402		ZD41152	Read
51		51	
ZB6453		CB41223	
13		13	
ZC6466		DC41236	
40		40	
BB6526		DB41276	
55		55	
CA6603		CT41353	
35		35	
PC6640		PD41410	
21		21	
ZZ6661		41431	
CN740	No. of words in buffer		
CL737	No. of words -1		
LA6227	Last add. of buffer		
CM741	No. of words +1		
NN432	No. of words to bootstrap		

Write Magnetic Tape

```
          IA    MD
DD    0   MJ    0         MU          → Setup
      1   RP    NN30000   DD5  }
      2   TP    MD        DD   }      Bootstrap
      3   0     30000     30000       Par-40/00  1st add.  last add.
      4   0     30000     30000       Next add. in buffer
      5   TP    Q         DD3
      6   QT    CC2       BG2         1st add. → temp.
      7   LQ    Q         17
     10   QT    CC2       BG3         Last add. → temp. → A
     11   ST    BG2       BG4         Last - 1st → temp.
     12   AT    CC5       BG6         Last - 1st + 1
     13   TP    BG5       A           No. left → A
     14   TJ    BG6       EE          Room this load ↓   No → EE
     15   TV    DD4       DD22 )
     16   TP    CC11      Q    }      Set RP & TP
     17   QS    BG6       DD21 }
     20   TU    DD3       DD22 )
     21   RP    30000     DD23 }      Words → buffer
     22   TP    30000     30000 }
     23   TP    DD4       A    )
     24   SA    BG6       17   }      Update address
     25   SA    BG6       25   }
     26   LT    0         DD4  )
     27   RS    BG5       BG6         Set cells left
     30   ZJ    FF        DD31        Buffer full ↓  no → FF
     31   RJ    WB        WB1         Write tape
     32   RJ    CR        CR1         Clear
     33   TP    DD4       MD4  }      Set drum
     34   TP    BG5       MG5  }
     35   MJ    0         MD36        → drum
     36   RJ    MD36      MD37        Exit
     37   MS    0         MD1         Stop
          CA    MD40
```

No room this load

```
        IA   ME
EE  0   TV   DD4     EE5
    1   TU   DD3     EE5      Set addresses
    2   TP   CC11    Q
    3   QS   BG5     EE4      Set n of RP
    4   RP   30000   EE6
    5   TP   30000   30000    Fill buffer
    6   RA   DD3     BG5      Update parameter
    7   TP   CC      BG5      Clear number of cells left
   10   RJ   WB      WB1      Write on tape
   11   RJ   CR      CR1      Clear
   12   TP   DD3     Q
   13   MJ   0       DD6      Return to finish parameter
        CA   ME14
```

Room & buffer not full

```
        IA   MF
FF  0   TP   DD3     Q
    1   QJ   FF2     DD33     40 ↓ 00 → exit
    2   RJ   WB      WB1      Write tape
    3   RJ   CR      CR1      Clear
    4   MJ   0       DD33     Exit
        CA   MF5
```

Write Buffer

```
          IA    MB
WB    0   MJ    0        (30000)      Exit
      1   TP    CC14     A
      2   ST    BG5      A            No. blks.──→Q
      3   DV    CC7      BG7
      4   ZJ    WB6      WB5
      5   RS    BG7      CC6          Index = No. blks -1──→BB7
      6   SP    BG7      0
      7   AT    CC6      BG10
     10   EF    0        CC3          Set bypass (=MJ 0 WB11 on 1103A)
     11   TV    CC10     WB21         Set at BF
     12   TP    DD3      Q
     13   LQ    Q        22
     14   QT    CC17     A            Set tape No. in code word
     15   TP    CC17     Q
     16   QS    A        CC4
     17   EF    0        CC4
     20   RP    10170    WB22
     21   EW    10000    (30000)      BF + n(170)
     22   RA    WB21     CC16
     23   IJ    BG7      WB17         Write n blks.
     24   MJ    0        WB           Exit
          CA    MB25
```

Clear

```
          IA    MR
CR    0   MJ    0        30000        Exit
      1   TP    CC10     DD4          = BF
      2   TP    CC14     BG5          No. of cells left = 1 buffer load
      3   RP    CN10000  CR
      4   TP    CC1      BF           Buffer = Z's ──→exit
          CA    MR5
```

Set-up

```
          IA    MU                    SU in core
SU    0   RP    NN30000  SU2
      1   TP    MD       DD           Bootstrap
      2   RJ    CR       CR1          Clear
      3   MJ    0        DD5
          CA    MU4
```

Constants

		IA	MC		
CC	0	0	0	0	Zero
	1	74	74747	47474	Fill buffer - Z's
	2	0	77777	0	u mask
	3	0	20000	04000	Set bypass mode - TCU2
	4	02	00606	0	Write & 128/in-tape #1
	5	0	1	0	1 in u
	6	0	0	1	1 in v
	7	0	170	0	Words/blk. in u
	10	0	BF	BF	1st add. of buffer
	11	0	07777	0	Mask
	12	0	0	0	
	13	0	2	2	
	14	0	CN	0	Set number of cells left
	15	0	LA	LA	Last address in buffer
	16	0	0	170	Words/blk. in v
	17	0	1	70000	
		CA	MC20		

Explanation of Temporaries and Variables

Core BG	Drum MG			
0	0	0	0	High } Check sum
1	0	0	0	Low } Check sum
2	0	u	0	u is first address
3	0	u	0	u is last address
4	0	u	0	u = Last-first
5	0	u	0	u = No. cells left in buffer
6	0	u	0	u = Last-first +1
7	0	0	v	v = No. of blocks -1 (Index)
10	0	u	0	u = No. of blocks
11	0	0	0	
12	0	0	0	

```
        IA   ZD
ZA  0   TP   Q        ZD2
    1   MJ   0        ZD47
    2   0    30000    30000      Par.   0   1st add.   last add.
    3   LQ   Q        22
    4   QT   ZB       ZB1
    5   TP   ZB       Q          Set tape number
    6   QS   ZB1      BB36
    7   QS   ZB1      BB37
   10   TP   ZA2      Q
   11   QT   ZB2      ZB3        1st add. ---> ZB3
   12   LQ   Q        17
   13   QT   ZB2      A          Last add. ---> A
   14   ST   ZB3      A
   15   AT   ZB4      ZB5        Last - 1st + 1 ---> ZB5
   16   TJ   ZB6      ZA31       Fit ---> ZA31  No ↓
   17   TP   ZB7      ZC2
   20   RJ   ZC       ZC1        Fill buffer
   21   TP   ZA2      Q
   22   LQ   Q        25
   23   TV   Q        ZA25       Buffer ---> storage
   24   RP   CN30000  ZA26
   25   TP   BF       (30000)
   26   RA   ZA2      ZB10       Update par.
   27   TP   ZA2      Q
   30   MJ   0        ZA3
   31   TP   ZB5      Q
   32   LQ   Q        25         No. ---> V
   33   TP   ZB11     A
   34   AT   Q        ZC2        Read to buffer
   35   RJ   ZC       ZC1
   36   TP   ZA2      Q
   37   LQ   Q        25
   40   TV   Q        ZA44       Set address
   41   TP   ZB12     Q          Mask ---> Q
   42   QS   ZB5      ZA43
   43   RP   30000    ZD45       Buffer ---> storage
   44   TP   BF       (30000)
   45   RJ   ZD45     ZD46       Exit
   46   MS   0        ZD
   47   RP   NN30000  ZA3
   50   TP   MD       DD
        CA   ZD51
```

		IA	CB		
ZB	0	0	1	70000	Mask
	1	0	0	0	Tape No.
	2	0	77777	0	Mask
	3	0	(0)	0	1st address
	4	0	1	0	
	5	0	(0)	0	Last - 1st + 1
	6	0	CM	0	
	7	0	BF	LA	Standard par. to fill buffer
	10	0	CN	0	
	11	0	BF	BS	Other par.
	12	0	07777	0	Mask
		CA	CB13		

Read Tape

```
          IA    DC
ZC    0   MJ    0        30000     Exit
      1   MJ    0        ZC3       Start
      2   0     30000    30000     0 address    address
      3   TP    ZC2      Q         1st add. ──→ 35
      4   QT    ZC34     ZC35
      5   LQ    Q        17        Last - 1st +1 ──→ A_u
      6   QT    ZC34     A
      7   ST    ZC35     A
     10   AT    ZC37     A
     11   LQ    Q        6         Address ──→ BB2
     12   TV    Q        BB2
     13   DV    BB45     ZC33      No. of blks. ──→ index
     14   TP    A        ZC32      Store remainder
     15   TP    Q        A         No. of blks. ≠ zero ↓ No ──→ ZC26
     16   ZJ    ZC17     ZC26
     17   RS    ZC33     ZC36      Index - 1
     20   TU    BB45     BB2       No. of words = 170
     21   RJ    BB       BB1       Read
     22   RA    BB2      ZC31      Increase address
     23   IJ    ZC33     ZC21      n blks.
     24   TP    ZC32     A         Remainder zero ──→ exit    No ↓
     25   ZJ    ZC26     ZC
     26   TU    ZC32     BB2       Read remainder
     27   RJ    BB       BB1
     30   MJ    0        ZC        Exit
     31   0     0        170
     32   0     0        0         Remainder
     33   0     0        0         No. of blocks
     34   0     77777    0
     35   0     30000    0         1st address
     36   0     0        1
     37   0     1        0
          CA    DC40
```

Read One Block of Tape

```
          IA    DB
BB    0   MJ    0        (30000)        Exit
      1   MJ    0        BB3
      2   0     (30000)  (30000)        Par. = 00  n  address
      3   MJ    0        BB4            (on 1103A = MJ 20000 BB5)
      4   EF    0        BB43           Set bypass (1103A = MJ 0 BB5)
      5   EF    0        BB42           Set normal bias
      6   TV    BB2      BB52           Set address
      7   TP    BB44     Q       }      Set n of RP
     10   QS    BB2      BB51    }
     11   QT    BB2      Q              n ──→ Q
     12   TP    BB45     A              170 ──→ A
     13   ST    Q        A       }      170 - n ──→ RP
     14   AT    BB46     BB53    }
     15   RJ    BB53     BB50           Read
     16   ER    0        A              IOA ──→ A
     17   ZJ    BB20     BB34           Parity ↓   No ──→ 34
     20   EF    0        BB41           Set high
     21   RJ    BB53     BB47           Read
     22   ER    0        A              IOA ──→ A
     23   ZJ    BB24     BB34           Parity ↓   No ──→ 34
     24   EF    0        BB40           Set low
     25   RJ    BB53     BB47           Read
     26   ER    0        A              IOA ──→ A
     27   ZJ    BB30     BB34           Parity ↓
     30   EF    0        BB37           Move back
     31   TP    CA10     PC2     }      Print PARITY
     32   RJ    PC       PC1     }
     33   MS    0        BB1            Stop for rereads
     34   EF    0        BB42           Set normal (1103A = MJ 0 BB35)
     35   MJ    0        BB             Exit
     36   2     602      10000          Read one blk. fwd.
     37   2     14       10001          Move back one blk.
     40   2     1        70000          Low
     41   2     1        60000          High
     42   2     1        50000          Normal
     43   0     20000    04000          Set bypass
     44   0     07777    0
     45   0     170      0
     46   RP    0        (30000)
     47   EF    0        BB37           Move back one blk.
     50   EF    0        BB36           Read one blk.
     51   RP    10000    BB53    }
     52   ER    10000    (30000) }
     53   0     0        0       }      RP
     54   ER    10000    A       }      Throwaway
          CA    DB55                    BB55 in core
```

```
        IA   PD
PC   0  MJ   0       30000          Exit
     1  MJ   0       PC16           Start
     2  0    30000   0              0    address n
     3  TP   PC 2    Q
     4  QT   CA5     A45
     5  ST   CA6     PC20
     6  TU   PC2     PC7
     7  SP   30000   52
    10  PR   0       A
    11  SS   A       6
    12  ZJ   PC10    PC13
    13  RA   PC7     CA7
    14  IJ   PC20    PC7
    15  MJ   0       PC
    16  RP   4       PC3
    17  PR   0       PC4
    20  0    0       0              Index
        CA   PD21
```

Constants
(From 1103A Unicode Service Routines)

```
          IA    CT
CA    0   0     170       ZZ
      1   67    50342     65127  ⎫
      2   30    77657     36566  ⎬        Sentinel
      3   30    47776     62452  ⎪
      4   30    77777     77777  ⎭
      5   0     0         77777
      6   0     0         1
      7   0     1         0
     10   0     CA13      10
     11   0     CA23      11
     12   0     77777     0
     13   47    12203     02204
     14   20    12120     31204
     15   46    15301     21401
     16   25    42574     24704
     17   04    24013     01201
     20   04    26031     20412
     21   20    12203     02224
     22   57    42040     40404
     23   47    07033     40601
     24   04    34061     41603
     25   22    20042     42524
     26   01    20070     40130
     27   15    20040     30604
     30   24    20121     70304
     31   57    52420     40447
     32   24    01301     20157
     33   42    04040     40404
     34   2     200       10000        Rewind #1
          CA    CT35
```

4. UNICODE SAMPLE CODING

4. UNICODE SAMPLE CODING.

a. Matrix Inversion by UNICODE

Assume A is the matrix to be inverted and ${}^{i}C$ is the i-th approximation to the inverse. Set ${}^{0}C$ to A^{T}. Then for the i-th iteration (i=1, 2, ..., n where n is the rank of the matrix)

$$ {}^{i}C_{jk} = \frac{{}^{i-1}C_{jk}}{A_i \, {}^{i-1}C_i} \qquad \text{for } k=i $$

$$ {}^{i}C_{jk} = {}^{i-1}C_{jk} - {}^{i}C_{ji}(A_i \, {}^{i-1}C_k) \qquad \text{for } k \neq i $$

where A_i is the i-th row of A and ${}^{j}C_i$ is the i-th column of ${}^{j}C$. After n repetitions C_n = Inverse of A.

Next an estimate of the error is made

$$ \text{Average error} = \frac{\Sigma\Sigma(A_i \, {}^{n}C_j)-n}{n^2} $$

If the average error is greater than the error allowed by the routine, ${}^{n}C$ is used as a new value for ${}^{0}C$ and a new approximation is made. Sentence 2.7 of the program establishes the error allowed. Reducing ae (allowable error) to a smaller amount may increase the number of iterations needed to obtain the desired inversion.

To run the program for a different size matrix, change sentences 1, 2.5, and 3. In 2.5, n is given the size of the new n x n matrix. In 3, m is given a numerical value 1 less than n. The numerical value of n determines the new dimensions used for the subscripted variables in sentence 1.

The input matrix A must be prepared on a data tape. It is brought into the computer by the Automatic Data Read-In.

```
                unicode program .
                matrix inversion



1               dimension a(10,10),c(10,10),r(10)   .
2               start  .
2.5             n=10  .
2.7             ae=0.01  .
3               m=9  .
3.1             vary i 0(1)m sentences 3.2 thru 3.3   .
3.2             vary j 0(1)m sentence 3.3 then resume 3.1    .
3.3             list a(i,j), i, j, tape 4    .
4               vary i 0(1)m sentences 5 thru 6 then jump to 6.5  .
5               vary j 0(1)m sentence 6 then resume 4  .
6               c(i,j)=a(j,i)   .
6.5             l=-1  .
7               vary i 0(1)m sentences 7.5 thru 24 .
7.5             vary k 0(1)m sentences 8 thru 10 then jump to 11  .
8               r(k)=0  .
9               vary j 0(1)m sentence 10 then resume 7.5  .
10              r(k)=r(k)+(a(i,j)Xc(j,k))  .
11              vary j 0(1)m sentence 12 then jump to 13  .
12              c(j,i)=c(j,i)/r(i)   .
13              if l = -1 jump to 20  .
14              la=0  .
15              lb=1  .
16              vary k la(1)lb sentences 17 thru 18 then jump to 19  .
17              vary j 0(1)m sentence 18 then resume 16  .
18              c(j,k)=c(j,k)-(c(j,i) X r(k))   .
19              if lb = m jump to 23  .
20              la = l+2  .
21              lb = m  .
21.5            if la G m jump to 24.1  .
22              jump to sentence 16  .
23              l = l+1  .
24              resume 7  .
24.1            q=0  .
24.15           vary i 0(1)m sentences 24.2 thru 24.3 then jump to 24.35  .
24.2            vary k 0(1)m sentences 24.25 thru 24.3 then resume 24.15  .
24.25           vary j 0(1)m sentence 24.3 then resume 24.2  .
24.30           q = q + (a(i,j) X c(j,k))  .
24.35           e = (q-n)/n^2  .
24.39           type e .
24.4            if |e| G ae, jump to 6.5  .
24.5            vary i 0(1)m sentences 24.6 thru 25 then jump to 26  .
24.6            vary k 0(1)m sentence 25 then resume 24.5  .
25              list c(i,k),i,k, tape 4,  .
26              stop .
zzzzzzend of tape .
```

b. Floating Point to Fixed Point Sub-Program

This is a method of converting a floating-point number to the nearest integer. The sub-program to do this is shown operating at sentences 100 to 106. Controlling sentences of the main program which reference the sub-program are given here as sentences 50-51.

Note that the Vary loop shown is never finished. A jump out is made to exit sentences when the nearest integer is found.

```
50.  Compute Fix (Z(J)) Δ.
51.  I = KK Δ.
100. Fix (Q(KA)) Δ.
101. ZZ = |Q(KA)| Δ.
102. Vary KK 0(1) 999999 with ZZ ZZ(-1) -2 Sentence 103 Δ.
103. If ZZ < 0.5 jump to 104 Δ.
104. If Q (KA) > = 0 jump to 106 Δ.
105. KK = -KK Δ.
106. Exit Δ.
```

c. Linear Programming Application.

```
Unicode program

  linear program
  original dantzig method

1     dimension a(17),x(8,17), z(17), delta(17), c(17), psi(8), p 8), temp(17), work(8)
2     start .
2.1   lf = 7 .                          Note: Only sentences 1 - 2.3 need be changed
2.2   lg = 16 .                              for matrices of different size.
2.3   g = 16 .
3.1   read x .
3.2   read c .
3.3   read psi .
3.4   read p .
                                                          BEGINNING OF CYCLE
4     x(0,0) = x(0,0) + 1 .                               Cycle Count
5     vary i 0(1)lg sentences 6 thru 8 .
6     z(i) = 0 .                                          0) z_i
7     vary j 1(1)lf sentence 8 .
8     z(i) = z(i) + x(j,i) X psi(j) .
9     vary i 0(1)lg sentence 10 .                         1) z_i - c_i = δ_i
10    delta(i) = z(i)-c(i) .
11    temp(0) = 0 .
12    k = 0 .
13    vary i 1(1)lg with b 1(1)g sentences 14 thru 17 .
14    if temp(0) G delta(i), jump to sentence 16 .
15    resume 13 .                                         Find smallest negative δ
16    vary j 1(1)lf sentence 16.1 .
16.1  if b = p(j), jump to sentence 15 .
16.2  temp(0) = delta (i) .
17    k = i .
18    if k = 0, jump to sentence 44 .                     If no negative δ is found, go
                                                          to output
19    type k .                                            Type k = subscript of incoming
                                                          vector
20    temp(0) = 1000 .
21    vary j 1(1)lf sentences 22 thru 29 .
22    if x(j,k) G 0, jump to sentence 25 .
23    temp(j) = -1 .                                      2) θ = min_i λ_j / x_ik
24    resume 21 .
25    if x(j,0) L 0, jump to sentence 23 .                    Find mr (position in basis)
26    temp(j) = x(j,0)/x(j,k) .                               for out going vector
27    if temp(j) G= temp(0) jump to sentence 24 .
```

```
28    temp(0) = temp(j) .                                  (Continued)
29    mr = j .
30    type mr .                                            Type mr
31    psi(mr) = c(k) .                                     3) place coefficient of in-
                                                              coming vector in new position
32    p(mr) = x(0,k) .                                     4) place number of incoming
                                                              vector in new position
33    vary j 1(1)1f sentence 34 .
34    temp(j) = x(j,k) .                                   Save column k and x_rk = α
34.1  work(0) = x(mr,k) .
35    vary i 0(1)1g sentences 36 thru 42 .
36    x(mr,i) = x(mr,i)/work(0) .                          5) calculate incoming row
37    if x(mr,i) not= 0, jump to sentence 40 .
38    resume 35 .
39    resume 40 .
40    vary j 1(1)1f sentences 41 thru 42 .                 6) calculate remaining parts
41    if j = mr, jump to sentence 39 .                        of matrix by columns
42    x(j,i) = x(j,i) - x(mr,i) X temp(j) .
43    jump to sentence 4 .                                 Go to start next cycle

                                                           END OF CYCLE - START OUTPUT
44    vary j 1(1)1f sentence 45 .                          List numbers of variables in
45    list p(j),j,tape 3,((variables in basis)) .             basis
46    vary i 0(1)1g sentence 47 .                          List δ's (shadow prices)
47    list delta(i),i,tape 3, ((delta)) .
48    vary j 0(1)1f sentences 49 thru 50 .                 List whole final matrix
49    vary i 0(1)1g sentence 50 .
50    list x(j,i),j,i, tape 3 ((final tableau)),
                              (matrix),(row),(column) .
51    vary i 1(1)1g sentence 52 .                          Find and list values of
52    temp (i) = 0 .                                          variables
53    vary j 1(1)1f sentences 54 thru 56 .
54    vary i 1(1)1g sentence 55 .
55    if x(0,i) = p(j), jump to sentence 56 .
56    temp(i) = x(j,0) .
58    vary i 1(1)1g sentence 59 .
59    list temp(i),i,tape 3, ((values of variables)) .
60    vary j 1(1)1f sentences 61 thru 66 .                 Compute and list back solution
61    read a .
62    work (j) = 0 .
```

```
63    vary i 1(1)lg sentences 64 thru 65  .
64    work(j) = work(j) + temp(i) X a(i)  .
65    z(j) = work(j) - a(0)  .
66    list z(j),j,tape 3, ((back solution)),
                          (deviation), (equation)  .
67    c(0) = 0  .
68    vary i 1(1)lg sentence 69  .
69    c(0) = c(0) + temp (i) X c(i)  .
70    list c(0), tape 3, ((profit function))  .
71    type c(0)  .
71.1  list x(0,0), tape 3, ((cycle count))  .
71.2  type x(0,0)  .
72    stop  .
zzzzzzend of tape  .
```

(Continued)

Compute and list back solution

Compute and type and list profit function

Type and list cycle count

Arrangement of the data tape:

A(I) 7 sets of A (rows of original equation)

	Cycle counter	Vector numbers						
Y(J,I)	0	1	2	3	. . . . .			16
Constant terms	x_1 0	x_1 1	x_1 2	x_1 3	· · · · ·			x_1 16
	x_2 0	x_2 1	x_2 2	x_2 3	· · · · ·			x_2 16
	.	.	.	.	. . . . .			.
	.	.	.	.	. . . . .			.
	.	.	.	.	. . . . .			.
	x_7 0	x_7 1	x_7 2	x_7 3	· · · · ·			x_7 16

Coefficient matrix

Constant term	Coefficient	Coefficient	Coefficient
x_0	x_1	x_2 · · ·	x_{16}

C (I) 0 C_1 C_2 C_3 . . . C_{16} coefficients of profit function

PSI(J) 0 ψ_1 ψ_2 ψ_3 . . . ψ_7 coefficients of profit function of the vectors in the basis

P (J) 0 P_1 P_2 P_3 . . . P_7 vector numbers in basis

The slack variables and artificial variables must be inserted in the data tape.
The starting basis is the unit matrix.

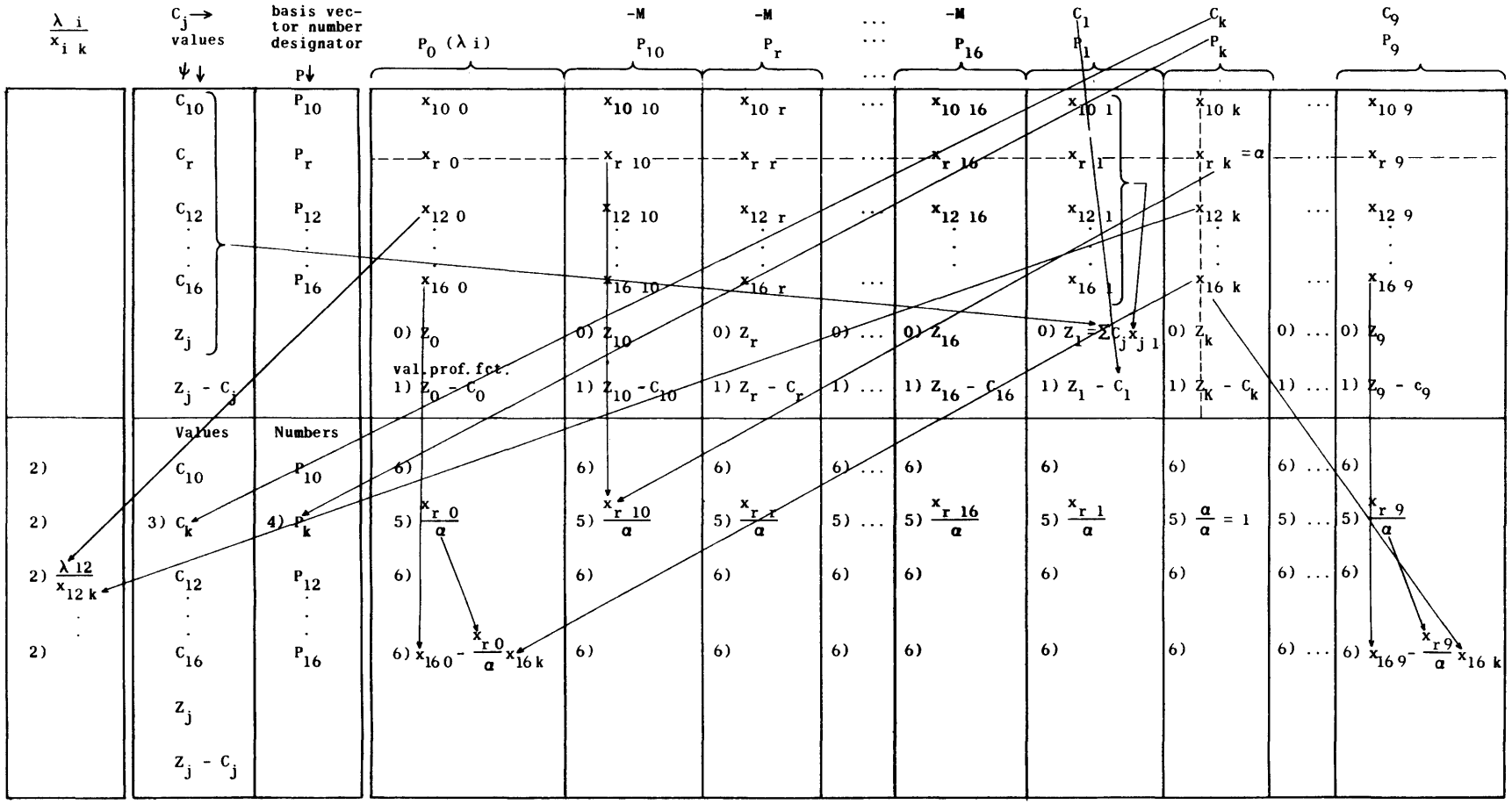
This column outside tableau need not be saved (for outgoing vector)

How the new tableau is computed from the preceding one: (Method taken from Charnes, Cooper, Henderson, "An Introduction to Linear Programming".)

x_{ik} are the values of column k in the old tableau, k being the number of the vector that was selected in the old tableau to enter the base.

x_{ri} are the values in the outgoing row.

$\lambda_i = x_{i\,0}$ are the coefficients of P_0 (the constants in the input data).



The new tableau is generated in the steps 0 - 1 - 2 - 3 - 4 - 5 - 6.

But watch!: Compute for step 6 only columns, where step 5 gave element $\neq 0$ and $x_{ik} \neq 0$; otherwise column remains unchanged. In this example the slack and artificial variables (10-16) are placed before the real variables (1-9). The sequence is unimportant but the constant vector P_0 must remain in the first column. The coded program has in the data tape the variables in their natural order 1, 2...10...16.

5. UNICODE CARD INPUT

5. UNICODE Card Input

Card input to UNICODE is possible without any change in the UNICODE Master Tape if a Card-to-Excess-Three Routine is written and added to the UNICODE Service Routines. Then the tape produced by this routine with card input may be used in UNICODE similarly to the tape produced by the Flex-to-Excess-Three Routine.

Assuming only 48 characters available on cards, certain restrictions in the writing of the UNICODE programs must be adopted. Superior figures can be eliminated by the use of the symbol POW followed by regular figures. Only one of the two signs, > and < , can be used. The space is simulated by no punch on the cards. The dash becomes the minus sign. No simulation is needed for the comma and period. Counting the 26 alphabetic, 10 numeric, and the 4 mentioned, only 8 available characters remain. The operator can obtain the essential UNICODE characters needed by an arbitrary assignment of characters which will later be changed by the conversion routine to the excess-three value desired. The following is one such possible assignment.

Excess-Three Character	Card Character	Card Punching
/	/	0 - 1
<	@	4 - 8
*	*	X - 4 - 8
+	&	Y
=	#	3 - 8
|	:	Y - 4 - 8
(	$	X - 3 - 8
)	%	0 - 4 - 8

In the above assignment, there is no greater than (>) sign. If in the IF sentence, a term such as "X > Y" is wanted, it should be written as "Y < X". Similarly, if a term "X > = Y" is wanted, it should be written

"Y < = X". If the converse of the latter is wanted, in addition, in an IF sentence, it is supplied automatically by a jump to the following sentence on failure of the < = relation to hold. Further flexibility can be introduced here by making this next sentence a JUMP sentence. It can be seen that only two clauses are now permitted in the IF sentence, but, because of the possibilities of combining the IF sentence with a subsequent JUMP sentence, there is no real loss in generality.

Each 80-column card should be considered a complete blockette. The conversion routine should use spaces to fill out the remaining 40 characters in the blockette.

If a sentence overruns from one card to another, care must be taken not to break a symbol into two parts. One must always end the punching of a card on a divider or a space. Any number of spaces may be put between symbols of a UNICODE sentence. Thus, 40 or more spaces between symbols in a sentence will not cause trouble. However, putting only part of a symbol on one card will destroy its meaning.

The first 6 positions of a regular UNICODE program card are reserved for the line number. The first 6 positions of an <u>overrun</u> program card must not be punched. On an overrun Data card, no indentions are needed. Of course, no figures may be broken at the end of a Data card. Thus, on Data cards, scientific notation numbers may not be broken into separate parts by use of spaces because spaces indicate the start of a new figure on Data lists. On the other hand, on a program card, the symbols making up a scientific notation number may be separated by spaces. Use of a comma following the last figure on a Data card will save computer time because it stops further analysis of the blockette and ignores the spaces used to fill the line.

The PRINT sentence, being limited to 6 blockettes, is somewhat limited in size by the use of cards, but this restriction is easily overcome by writing two or more PRINT sentences in succession.

The rules of tape preparation explained in the UNICODE Manual must be simulated when using cards. Thus, whenever blockettes of spaces are needed to fill out a block, the right number of blank cards must be inserted.

Card input for UNICODE programs and Data can be used at a computer installation which has no direct provision for card input to the computer if there is available a Card-to-Magnetic Tape Converter. If the cards have been properly prepared in accordance with the conventions described here, they can be run through the Card-to-Magnetic Tape Converter and this output tape then translated to a UNICODE tape by means of a short conversion routine.

Though this latter routine is at present specifically adjusted to the arbitrary assignment of characters given earlier in this paper, it can be readjusted to any other assignment merely by modifying the order or content of two 8-line tables.

Cards to UNICODE Tape

This routine has as input a magnetic tape obtained from cards via the Card-to-Magnetic Tape Converter. The characters on this tape are changed, as needed, to the excess-three equivalents which are recognized by UNICODE.

The space-fill option for no punch should be used in producing the input tape on the Card-to-Tape Converter. Further conventions followed in writing the programs and Data lists and punching the cards are described in the write-up, UNICODE Card Input. The arbitrary assignment of card characters given in the table therein is assumed to be standard. No special translation is needed for the first three characters of this table after the Card-to-Tape conversion. However, the remaining five characters have to be translated from one excess-three value to another before they are acceptable to UNICODE.

When it is known that none of these five characters have been used, there is no need to use this routine. The tape from the Card-to-Tape Converter may be used directly as an input to UNICODE. This will happen most frequently with Data lists. If a Data list is preceded by a Data Index, which uses equal (=) signs, of course, it will still be necessary to run the tape from the Card-to-Tape Converter through this conversion routine. Other than the equals sign which only occasionally occurs in a Data Index, only one character on Data lists, the semicolon, has no counterpart among the usual card characters. The semicolon should always be replaced by the comma in Programs or Data lists using cards. Thus, many sets of Data list cards, when properly punched and run through the Card-to-Tape Converter, will produce a tape completely acceptable to UNICODE.

If a Card-to-Magnetic Tape Converter is available, it will be found a more economical procedure to prepare all Data lists in this way because of the greater ease of correction. This will be true regardless of whether or not the program has been prepared on cards.

To translate the five characters mentioned above, the following two tables of excess-three values are used in this routine:

Card-to-Tape Produced Excess-Three Value	
00 00000 00016	&
00 00000 00035	#
00 00000 00062	:
00 00000 00055	$
00 00000 00075	%

UNICODE Accepted Code for Character	
00 00000 00063	+
00 00000 00076	=
00 00000 00042	|
00 00000 00017	(
00 00000 00043	)

Each of these tables is contained in an 8-line space. Thus, they can be increased in size up to eight lines and altered as needed to produce a different conversion. The order within the tables is significant. Thus, recognition on an input tape of the third value in the first table would give a substitution on the UNICODE tape of the third value of the second table.

There are two versions of this routine; one for the 1103A and one for the 1105. With the 1105 version an MJ2 setting indicates that Tape Control Unit 2 is being used. No MJ2 setting indicates use of TCU1. No settings are needed with the 1103A version.

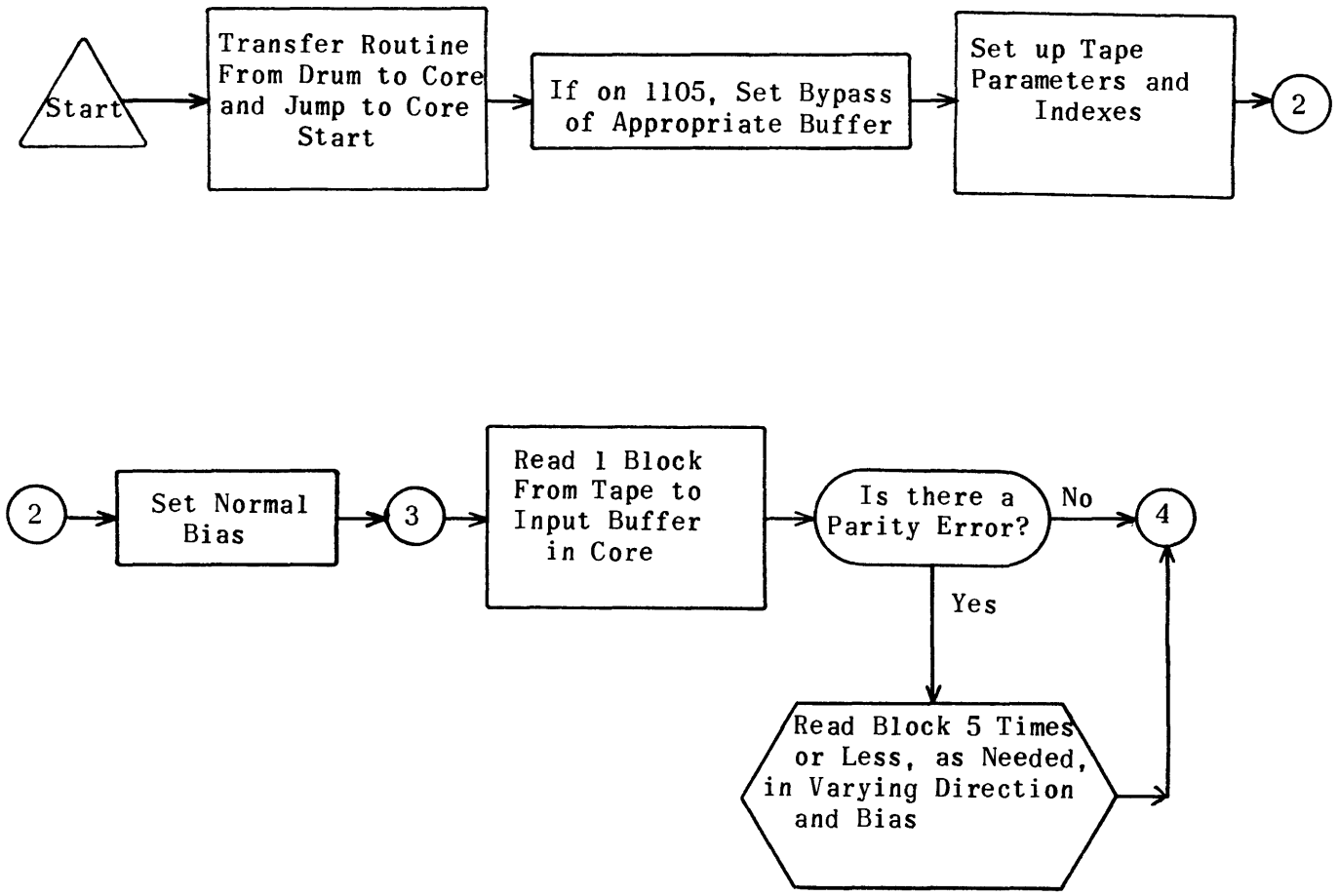
To operate either version, after checking whether an MJ2 setting is needed, put the tape produced by the Card-to-Magnetic Tape Converter on any Uniservo. The number of this Uniservo should be placed in A_v. In Q_v put the number of the Uniservo on which the UNICODE tape is to be written. The octal number of input blocks must be in Q_u.

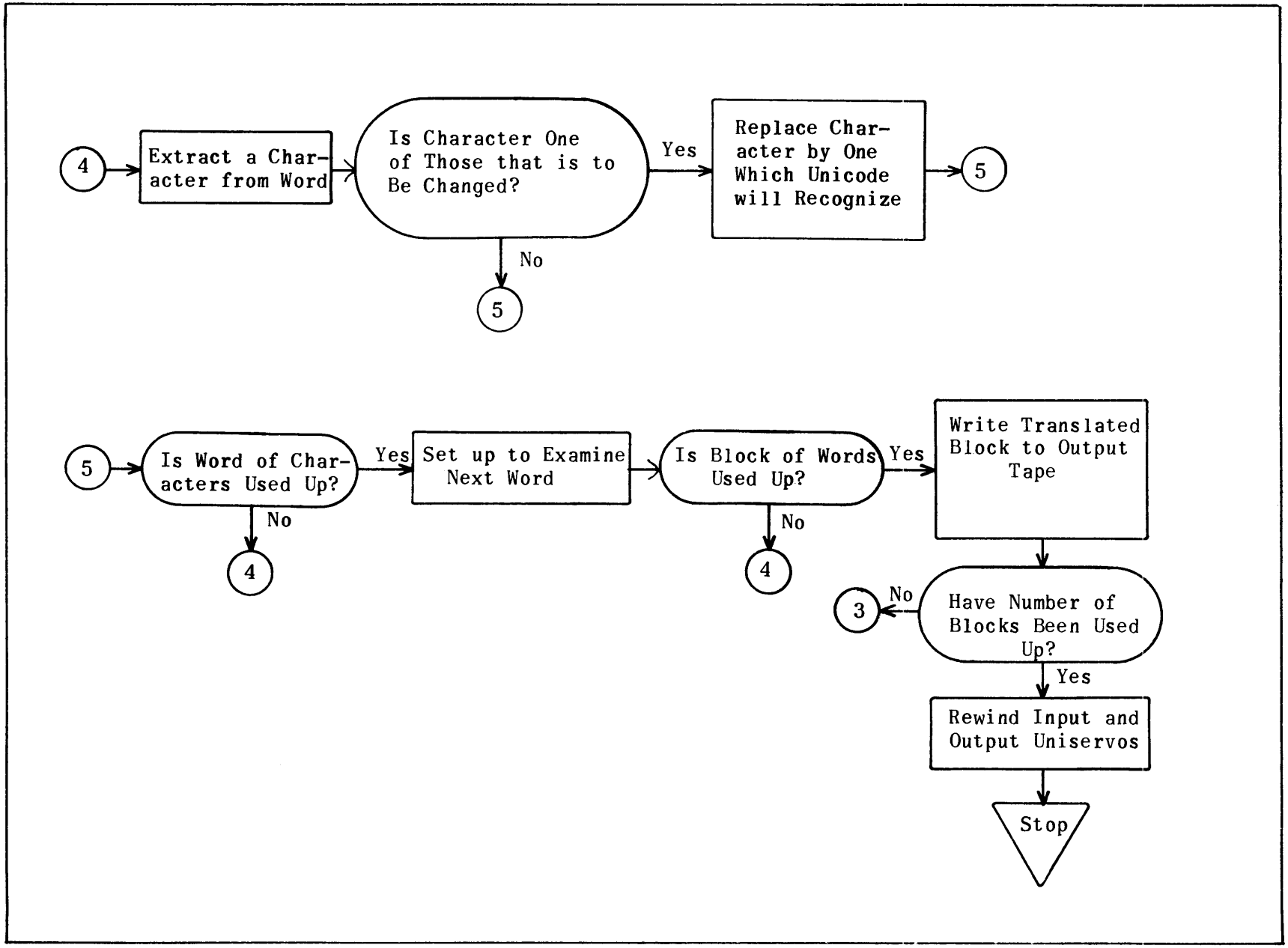
Start with beginning address at 77460. The routine transfers itself to core where it operates from 6300-6736. Both tapes are rewound on completion of the translation. The computer stops with PAK at 6300, which address, since the core is not restored, can be used for a start of any additional translation. No check is made for illegal parameters. A failure to recover from a parity error after 5 additional reads with varying bias and direction will cause a computer stop with the print-out: PY. Another 6-read trial can be obtained at this point by pushing the START button.

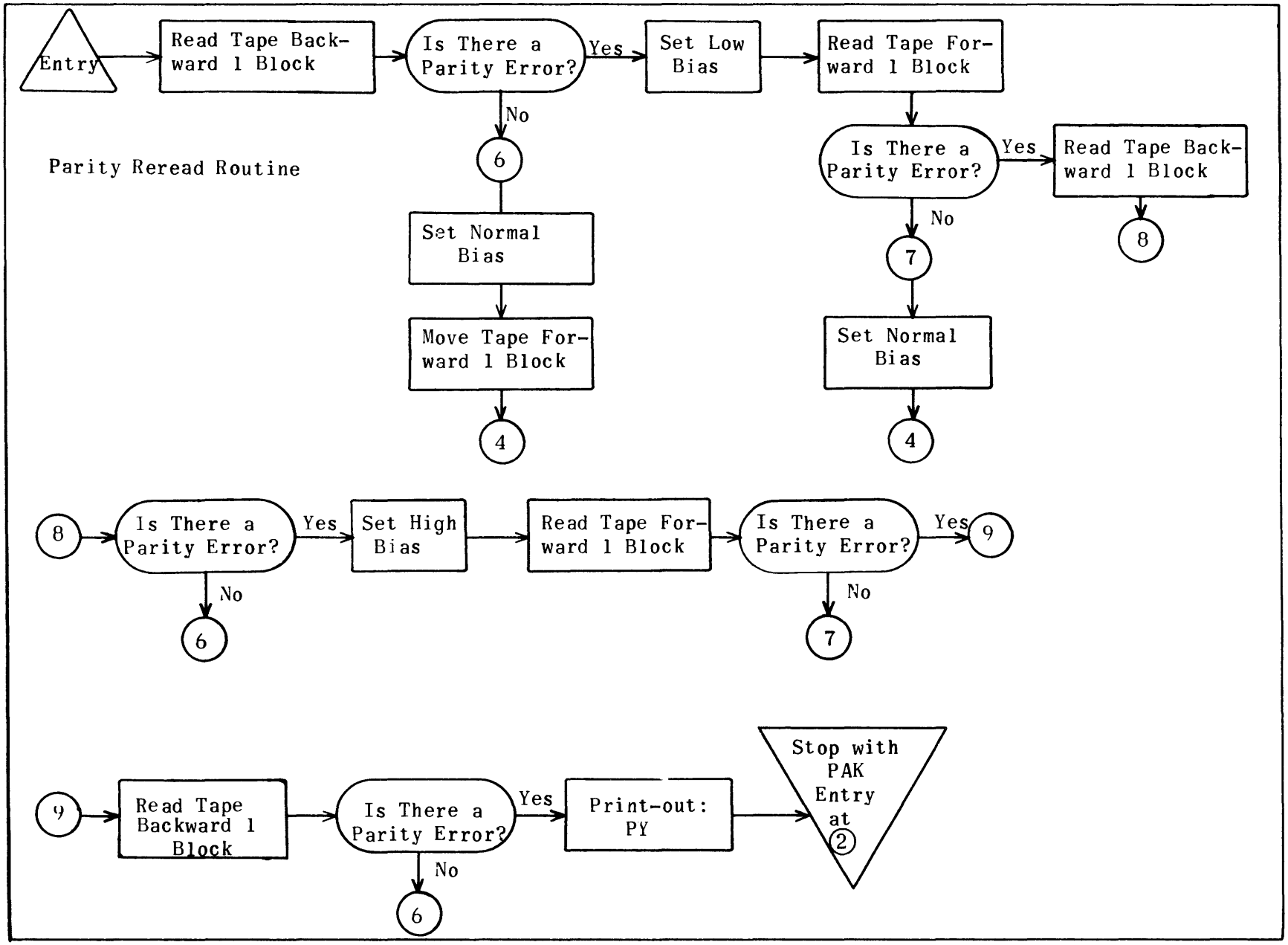
The flex copy of this routine loads in the drum from 77460-77714. It may be added to the bioctal tape of UNICODE Service Routines. It has not been included on any of the tapes of the UNICODE System but flow charts and annotated coding for it are shown following this write-up.

On both versions of the routine the first table is located at 77675 and the second table at 77705. A change in the number of significant values in the tables would necessitate a change in the repeat instruction that controls use of the first table. On the 1103A version this instruction reads 75 20005 06345 and is located at 77515. On the 1105 version this instruction reads 75 20005 06362 and is located at 77532. For example, an increase to 8-line sets of significant values in the tables in the 1105 version would require a change of the content of 77532 to 75 20010 06362.

Flow Charts for Routine to Convert Card-To-Tape Converted Tape to UNICODE Tape (1103A or 1105)




4
Extract a Character from Word
Is Character One of Those that is to Be Changed?
Yes
Replace Character by One Which Unicode will Recognize
5
No
5
5
Is Word of Characters Used Up?
Yes
Set up to Examine Next Word
Is Block of Words Used Up?
Yes
Write Translated Block to Output Tape
No
4
No
4
3
No
Have Number of Blocks Been Used Up?
Yes
Rewind Input and Output Uniservos
Stop


Entry
Read Tape Backward 1 Block
Is There a Parity Error?
Yes
Set Low Bias
Read Tape Forward 1 Block
No
6
Parity Reread Routine
Is There a Parity Error?
Yes
Read Tape Backward 1 Block
8
No
7
Set Normal Bias
Move Tape Forward 1 Block
4
Set Normal Bias
4
8
Is There a Parity Error?
Yes
Set High Bias
Read Tape Forward 1 Block
Is There a Parity Error?
Yes
9
No
6
No
7
9
Read Tape Backward 1 Block
Is There a Parity Error?
Yes
Print-out: PY
Stop with PAK Entry at 2
No
6

Regions for 1103A or 1105 Routine To Convert Card-To-Tape
Converted Tape to UNICODE Tape

```
RE DH77460
RE HL6300
RE BL6377
RE TC6460
RE NU6513
RE BE6522
RE NE6523
RE UC6533
RE TS6547
```

1105 Routine to Convert Card-to-Tape Converted Tape to UNICODE Tape

```
        IA DH
        RP  30233 HL        }
        TP  DH2 HL          }   Transfer Routine from drum to core
HLO     LA  A   14          }
  1     TP  A   UC6         }   Number of input Uniservo to UC6 in shifted position.
  2     TP  Q   UC4             Storing q in UC4
  3     MJ  20000 HL5           Is MJ2 set?  If so, jump to bypass buffer 2
  4     MJ  0 HL11              Jump to bypass buffer 1
  5     MJ  50000 HL5           Is Buffer 2 active?
  6     TP  TC1 UC13            Set up TCU2 selection
  7     EF  0 TC13              Bypass buffer 2
 10     MJ  0 HL14
 11     MJ  40000 HL11          Is buffer 1 active?
 12     TP  TC UC13             Set up TCU1 selection
 13     EF  0 TC12              Bypass buffer 1
 14     TP  TC26 Q          }
 15     RP  10010 HL17      }   Put TCU selection in tape handling parameter
 16     QS  UC13 TC2        }
 17     SP  UC6 0           }
 20     AT  TC2 UC          }   "Read forward 1 block from input servo" set up in UC
 21     TP  TC32 Q          }
 22     QT  UC4 UC12        }   u of Initial Q transferred to A_v
 23     LQ  A   25          }
 24     ST  TC23 UC5            Index of number blocks formed in UC5
 25     TP  TC22 UC12       }
 26     TV  UC4 UC12        }   v of Initial Q transferred to UC4 in shifted
 27     LA  UC12 14         }   position.  This is output servo number
 30     TP  A UC4           }
 31     AT  TC3 UC1             "Write 1 block to output servo" set up in UC1
 32     17 0 TC5                Set normal bias
 33     17 0 UC             }   Read forward 1 block to core
 34     75 10170 HL36       }
 35     76 10000 TS         }
```

```
36    76 0 A
37    ZJ HL40 HL41          Is there a parity error?
40    RJ BL BL1             Recover from parity error by attempted rereads
                            at different bias
41    TU TC21 HL45          Set up start of analysis of block at 1st address of
42    TU TC21 HL47          buffer, TS
43    TP TC20 UC3           Index of greatest possible (162) number of words
                            set up.  Rest of block words are spaces
44    TP TC16 UC2           Index for number characters in word
45    LQ TS 6
46    TP TC14 Q             Character to A for examination
47    QT TS A
50    RP 20005 HL62         Is character one of 1st table?
51    EJ NU HL52
52    SN Q 17               r ⟶ u of substituting command
53    SA HL50 0
54    TU A HL61
55    LQ HL45 Q25           Location in block to v of substituting command
56    TV Q HL61
57    RA HL61 TC15          r + BE to u of Substituting command to get proper
                            character of 2nd table
60    TP TC14 Q             Character substitution (2nd table char. for 1st
61    QS 30000 30000        table char.)
62    IJ UC2 HL45           Index jump on no. characters in word
63    RA HL45 TC17          Set-ups to examine next word
64    RA HL47 TC17
65    IJ UC3 HL44           Index jump on number possible  non-space words in
                            block
66    17 0 UC1
67    75 10170 HL71         Write 1 translated block on Output Uniservo
70    77 10000 TS
71    IJ UC5 HL33           Index on number of blocks
72    RA UC6 TC4            Rewinds input Uniservo
73    17 0 UC6
74    RA UC4 TC4            Rewinds output Uniservo
75    17 0 UC4
76    MS 0 HL               Stop with core starting address in PAK
```

```
BL          0   MJ   0 30000
            1   TV   TC25 BL6          Set-up to read from last address of buffer
Parity      2   TP   TC24 UC10         Index of number of words in buffer to UC10
Reread      3   TP   TC10 A
Routine     4   AT   UC6  UC7          Read backward 1 block input Uniservo
            5   17 0 UC7
            6   76 10000 TS167
            7   RS   BL6  TC23
           10   IJ   UC10 BL6          Providing 170 external reads into descending
           11   76 0 A                 addresses
           12   ZJ   BL13 BL52         Is there a parity error?
           13   17 0 TC6               Set low bias
           14   17 0 UC
           15   75 10170 BL17          Read forward 1 block input Uniservo
           16   76 10000 TS
           17   76 0 A                 Is there a parity error?
           20   ZJ   BL21 BL57
           21   TV   TC25 BL24         Set-up to read into last address of buffer
           22   TP   TC24 UC10         Index of number of words in buffer to UC10
           23   17 0 UC7               Read backward 1 block input Uniservo
           24   76 10000 TS167
           25   RS   BL24 TC23         Providing 170 external reads
           26   IJ   UC10 BL24
           27   76 0 A                 Is there a parity error?
           30   ZJ   BL31 BL52
           31   17 0 TC7               Set high bias
           32   17 0 UC
           33   75 10170 BL35          Read forward 1 block input Uniservo
           34   76 10000 TS
           35   76 0 A                 Is there a parity error?
           36   ZJ   BL37 BL57
           37   TV   TC25 BL42         Set-ups for reading backward
           40   TP   TC24 UC10
           41   17 0 UC7
           42   76 10000 TS167         Read backward 1 block input Uniservo
           43   RS   BL42 TC23
           44   IJ   UC10 BL42
```

```
        45   76 0 A
        46   ZJ BL47 BL52          Is there a parity error?
        47   75 10003 BL51
        50   PR 0 TC27             Print out"carriage return" PY
        51   MS 0 HL32             Stop with PAK at complete restart of at-
                                   tempt to read block
        52   17 0 TC5              Set normal bias
        53   TP TC11 A
        54   AT UC6 UC11           Move forward Input servo 1 block
        55   17 0 UC11
        56   MJ 0 BL               Exit from parity read
        57   17 0 TC5              Set normal bias
        60   MJ 0 BL               Exit from parity read
     TC 0    1 0 0                TCU1 indicator
         1   2 0 0                TCU2 indicator
Constants 2  0 602 0              Read forward 1 block & stop
         3   0 646 0              Write 1 block and stop
         4   0 200 0              Rewind tape
         5   0 1 50000            Normal bias
         6   0 1 60000            Low bias
         7   0 1 70000            High bias
        10   0 612 0              Read backward 1 block
        11   0 4 1                Move forward 1 block
        12   0 10000 04000        Bypass buffer 1
        13   0 20000 04000        Bypass buffer 2
        14   0 0 77               Mask
        15   0 BE 0               Address just before 2d table
        16   0 0 5                Index for characters in a word
        17   0 1 0                Adder
        20   0 0 161              Index for greatest possible number (162)
                                   words in block
        21   0 TS 0               Starting address of buffer
        22   0 0 0                Zero
        23   0 0 1                Adder
        24   0 0 167              Index for number words in block
```

```
     25    0 0 TS167                 Address of last word in block
     26    77 0 0                    Mask
     27    0 0 45                    Carriage return
     30    0 0 15                    P
     31    0 0 25                    Y
     32    0 77777 0                 Mask
NU   0     0 0 16   &                First Table Card-to-Tape Produced
                                     Excess-Three Values
      1    0 0 35   #
      2    0 0 62   :
      3    0 0 55   $
      4    0 0 75   %
      5    0 0 0
      6    0 0 0
      7    0 0 0
NE    0    0 0 63   +                Second Table Excess-Three Values
                                        Accepted by Unicode
      1    0 0 76   =
      2    0 0 42   |
      3    0 0 17   (
      4    0 0 43   )
      5    0 0 0
      6    0 0 0
      7    0 0 0
           CA DH235
```

Temporaries Used

```
UC   0    Read forward 1 block of tape
     1    Write 1 block on output tape
     2    Index for characters in a word
     3    Index for total words possible in block (162)
     4    Holds initial value of Q.  Later holds output tape no. shifted
     5    Index for number of blocks
     6    Holds initial A_V shifted of input tape number
     7    Read backward 1 block
    10    Index to read backward
    11    Move forward 1 block
    12    Temporary to get output servo no. shifted to right position
    13    Holds Tape Control Unit No. in "Operation" part of word.
```

TS Buffer of 170_8 lines used for reading into and writing from after translation to UNICODE characters.

1103A Routine to Convert Card-to-Tape Converted Tape to UNICODE Tape

```
      IA DH
      RP 30233 HL
      TP DH2 HL
HLO   LA A 14
  1   TP A UC6
  2   AT TC UC
  3   TP Q UC4
  4   TP TC25 Q
  5   QT UC4 UC12
  6   LQ A 25
  7   ST TC11 UC5
 10   TP TC10 UC12
 11   TV UC4 UC12
 12   LA UC12 14
 13   TP A UC4
 14   AT TC1 UC1
 15   17 0 TC13
 16   17 0 UC
 17   75 10170 HL21
 20   76 10000 TS
 21   76 0 A
 22   ZJ HL23 HL24
 23   RJ BL BL1
 24   TU TC7 HL30
 25   TU TC7 HL32
 26   TP TC6 UC3
 27   TP TC4 UC2
 30   LQ TS 6
 31   TP TC2 Q
 32   QT TS A
 33   RP 20005 HL45
 34   EJ NU HL35
```

Transfer routine to core from drum and jump to start in core

Number of input Uniservo to temporary in 5th-digit position

Parameter for tape read to temporary

Store q in temporary

Forming index for number blocks in temporary

Output tape number in 5th position to temporary

Forming parameter for tape write in temporary

Set normal bias

Read 1 block of tape

Providing 170_8 external reads

Is there a parity error?

Jump to parity reread routine

Set up examining loops to beginning of block

Index for 162_8 (maximum possible non-space) words in block set up

Index for 6 characters in line set up

Is a character one of those in first table?

```
35  SN Q 17
36  SA HL33 0
37  TU A HL44
40  LQ HL30 Q25
41  TV Q HL44
42  RA HL44 TC3
43  TP TC2 Q
44  QS 30000 30000
45  IJ UC2 HL30
46  RA HL30 TC5
47  RA HL32 TC5
50  IJ UC3 HL27
51  17 0 UC1
52  75 10170 HL54
53  77 10000 TS
54  IJ UC5 HL16
55  RA UC6 TC12
56  17 0 UC6
57  RA UC4 TC12
60  17 0 UC4
61  MS 0 HL
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
    0 0 0
```

Replace character by the corresponding one in second table

Jump back to beginning of loop to examine next character in line

Set-ups to examine next line in block

Jump back to loop to examine next line. Is block used up?

Write completely translated block to output tape

Provide 170_8 external writes

Have number of blocks been exhausted?

Rewind input Uniservo

Rewind output Uniservo

Stop with core starting address in PAK

Unused fill lines put in so this routine will occupy same regions as 1105 counterpart

```
        BL 0    MJ   0 30000
Parity      1   TV   TC20 BL6          Set-up to read from last address of buffer
Reread      2   TP   TC17 UC10         Index of number of words in buffer to temporary
Routine     3   TP   TC16 A
            4   AT   UC6  UC7          Read backward 1 block of input Uniservo
            5   17   0    UC7
            6   76   10000 TS167
            7   RS   BL6  TC11         Providing 170(8) external reads into descending
            10  IJ   UC10 BL6          core addresses
            11  76   0    A
            12  ZJ   BL13 BL52         Is there a parity error?
            13  17   0    TC14         Set low bias
            14  17   0    UC           Read forward 1 block input tape
            15  75   10170 BL17
            16  76   10000 TS          170 external reads provided
            17  76   0    A
            20  ZJ   BL21 BL57         Is there a parity error?
            21  TV   TC20 BL24
            22  TP   TC17 UC10
            23  17   0    UC7
            24  76   10000 TS167
            25  RS   BL24 TC11         Read backward 1 block of input tape to
            26  IJ   UC10 BL24         descending core addresses
            27  76   0    A
            30  ZJ   BL31 BL52         Is there a parity error?
            31  17   0    TC15         Set high bias
            32  17   0    UC
            33  75   10170 BL35
            34  76   10000 TS          Reading forward 1 block input tape
            35  76   0    A
            36  ZJ   BL37 BL57         Is there a parity error?
            37  TV   TC20 BL42
            40  TP   TC17 UC10
            41  17   0    UC7
            42  76   10000 TS167       Read backward 1 block of input tape to
            43  RS   BL42 TC11         descending core addresses
            44  IJ   UC10 BL42
```

```
        45   75 0 A                  }
        46   ZJ BL47 BL52            }  Is there a parity error?
        47   75 10003 BL51           }
        50   PR 0 TC22               }  Print-out: PY
        51   MS 0 HL15                  Stop with PAK at place where read can start all
                                        over
        52   17 0 TC13                  Set normal bias
        53   TP TC21 A               }
        54   AT UC6 UC11             }  Move forward 1 block
        55   17 0 UC11               }
        56   MJ 0 BL                    Exit from reread routine
        57   17 0 TC13                  Set normal bias
        60   MJ 0 BL                    Exit
     TC  0   2 602 0                    Parameter for reading 1 block forward
Constants 1   2 656 0                    Parameter for writing 1 block
         2   0 0 77                     Mask
         3   0 BE 0                     Address of word before 2nd table
         4   0 0 5                      Index set-up for 6 characters of line
         5   0 1 0
         6   0 0 161                    Index set-up for maximum possible (162_8)
                                        words in block
         7   0 TS 0                     Beginning address of buffer
        10   0 0 0
        11   0 0 1
        12   2 200 0                    Rewind parameter
        13   2 1 50000                  Normal bias parameter
        14   2 1 60000                  Low bias parameter
        15   2 1 70000                  High bias parameter
        16   2 612 0                    Read backward 1 block
        17   0 0 167
        20   0 0 TS167                  Address of last word in buffer
        21   2 4 1                      Move forward 1 block parameter
        22   0 0 45                     Carriage return
        23   0 0 15                     P
        24   0 0 25                     Y
        25   0 77777 0                  Mask
```

	0 0 0		Dummy fill lines put in to maintain correspondence with 1105 routine
	0 0 0		
	0 0 0		
	0 0 0		
	0 0 0		
NU 0	0 0 16	&	First table - Card-to-Tape Produced Excess-Three Values
1	0 0 35	#	
2	0 0 62	:	
3	0 0 55	$	
4	0 0 75	%	
5	0 0 0		
6	0 0 0		
7	0 0 0		
NE 0	0 0 63	+	Second Table-Excess-Three Codes Accepted by Unicode
1	0 0 76	=	
2	0 0 42	|	
3	0 0 17	(	
4	0 0 43	)	
5	0 0 0		
6	0 0 0		
7	0 0 0		

CA DH^{235}

Temporaries Used

UC	0	Read forward 1 block of tape from input Uniservo
	1	Write 1 block on output tape
	2	Index for characters in a word
	3	Index for total words possible in block (162_8)
	4	Holds initial value of Q. Later holds output tape No. shifted
	5	Index for number of blocks
	6	Holds initial A_V shifted of input tape number
	7	Read backward 1 block
	10	Index to read backward
	11	Move forward 1 block
	12	Temporary to get output servo No. shifted to right position
	13	Not used on 1103A version

TS Buffer of 170_8 lines used for reading into and writing from after translation to Unicode characters.

6. STATISTICAL MISCELLANY

6. STATISTICAL MISCELLANY

a. Call words of UNICODE

1. Regular call words

7 7 - - -	Subscripted variable
7 6 X - -	Subscripted 'dummy' variable for Subprogram - X denotes the number of subscripts
7 5 - - -	Subscripted 'dummy' variable for functions
6 7 - - -	Constant (Floating or Fixed Point)
6 6 - - -	Function (Floating)
6 5 - - -	Floating point variable
6 4 - - -	Fixed point variable
6 3 - - -	Non-subscripted dummy variable in Subprogram
6 2 - - -	Non-subscripted dummy variable for functions
6 1 - - -	Dummy function in Subprogram
5 - - - X	Library routine - X denotes the number of operands
4 - - X X	Pseudo operation - X X denotes the number of operands
2 7 - - -	Statement of Main Program
2 6 - - -	VARY statement of Main Program
2 5 - - -	Equation defining 6 6 - - -, 6 5 - - -, or 6 4 - - - type variable (before START)
2 4 - - -	Equation defining 7 7 - - - type variable (before START)
2 3 - - -	END∧OF∧TAPE
2 2 - - -	Sentence of Subprogram (including VARY statement)
7 1- - -	Absolute address call word for those addresses 1000 thru 1777

2. SUPPLEMENTARY CALL WORDS AND ASSOCIATED PRELUDE ENTRIES

The following supplementary call words will be unique only within a given generated routine.

Call words	
10---	Relative constants
20---	Fixed constants
60---	Fixed temporary storage
70---	Working temporary storage

The rightmost three octal digits of the call words specify the relative running location of the item within the designated region, the first item of a region being designated by the digits, "000". Thus, the call word, 10003, would be used as a relative address to reference the fourth relative constant in order of increasing memory address. These call words facilitate the generation of machine instructions referencing items in the constant or temporary regions without predetermining the actual addresses of these items relative to the beginning of the running routine.

Relative constants are those which are relatively coded and must be modified during processing. Modifying "10" lines appearing in the Relative Constant region must not be counted when determining the three call-word digits specifying the relative location of constants within the region. Fixed Constants are those constants which are coded in absolute and are not to be modified. Fixed temporaries are those in which information is stored at some time during the execution of the routine and is retained for the duration of the routine. On the other hand, working temporaries* would be those which are used and reused to store different information during a given execution of the routine.

The number of entries in each of the constant or temporary regions, exclusive of "10" lines, must be included in the third word of the Prelude for

* NOTE: Distinguishing between two types of temporary storage was necessary in the generation of equation routines. In general, this may not be necessary.

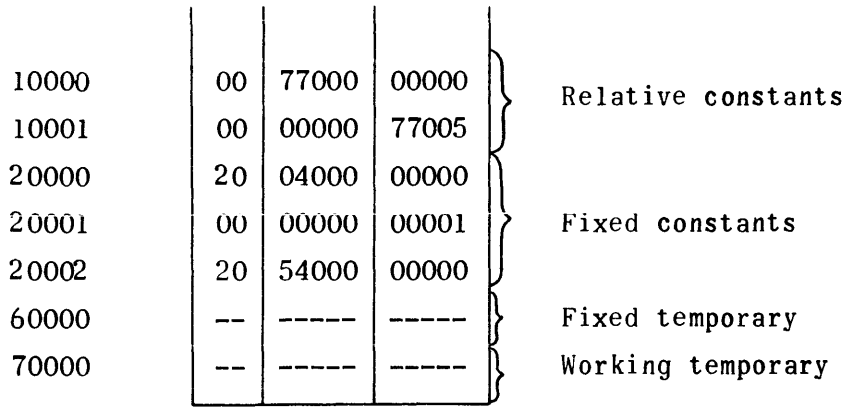
the routine. This word will be divided into four groups of three octal digits each. Each group will contain the number of entries in the corresponding region, packed to the right and filled with zeros within the group. The word will be divided as follows:

Number of Fixed Temporaries	Number of Working Temporaries	Number of Relative Constants	Number of Fixed Constants
3 octal digits	3 octal digits	3 octal digits	3 octal digits

When any of the regions is not used, a corresponding zero entry must be made in the third word of the Prelude.

An example of the use of supplementary call words in a generated routine would be:

Relative Address	Generated Routine Op	"u"	"v"	
	--	-----	-----	Prelude
	--	-----	-----	
	00	10010	02003	
	--	-----	-----	
	--	-----	-----	
	--	-----	-----	
1000	45	00000	30000	Body of routine
1001	15	10000	01002	
1002	66	30000	20000	
1003	11	31000	60000	
1004	67	20002	60000	
1005	11	31000	70000	
1006	65	60000	70000	
1007	11	31000	70000	
1010	66	60000	20000	
1011	64	31000	70000	
1012	16	10001	01014	
1013	21	01014	20001	
1014	11	31000	30000	
1015	45	00000	01000	


10000
00
77000
00000
Relative constants
10001
00
00000
77005
20000
20
04000
00000
20001
00
00000
00001
Fixed constants
20002
20
54000
00000
60000
--

Fixed temporary
70000
--

Working temporary

3. USE OF CALL WORDS TO REFERENCE SUB-PROGRAM INPUT LIST

The inputs for a sub-program are designated by means of dummy variables appearing in the heading of the sub-program. These inputs may be floating or fixed-point single-valued quantities, subscripted variables, or functions. The dummy variables will be assigned dummy call words depending on the type of input. The leftmost two octal digits of the call words indicate the type of input, the third octal digit specifies the number of subscripts in the case of subscripted variables, and the rightmost two octal digits indicate the relative location of the particular entry in the sub-program input list. These call words are used in the generated routines of a sub-program to reference the entries in the sub-program input list.

The input file for a dummy subscripted variable will contain the call word of the subscripted input variable, and the values of its multipliers, modulus and subscripts in that order. The dummy subscripted variable will have a 76--- type call word; the multipliers, modulus, and subscripts will have 63--- type call words.

SUBSCRIPTED DUMMY VARIABLE INPUT FILE*

Dummy Call word	Op	u	v	
764--	00	77---	77---	Call word of input variable in u and v
630--	--	-----	-----	Numerical value of first multiplier
630--	--	-----	-----	" " " second "
630--	--	-----	-----	" " " third "
630--	--	-----	-----	" " " modulus
630--	--	-----	-----	" " " first subscript
630--	--	-----	-----	" " " second "
630--	--	-----	-----	" " " third "
630--	--	-----	-----	" " " fourth "

*Note: Length of file varies with number of subscripts.

The input file for a dummy function will contain the call word for the input function, and the call word of the equation defining the function. The dummy function will have a 61---type call word.

DUMMY FUNCTION INPUT FILE

Dummy Call word	Op	u	v	
61---	00	66---	66---	Call word of input function in "u" and "v"
61---	00	25---	25---	Call word of equation defining input function in "u" and "v"

The input file for a dummy floating or fixed-point single-valued variable contains only the value of the input quantity.

DUMMY SINGLE-VALUED VARIABLE INPUT FILE

Dummy Call word	Op	u	v	
63---	--	-----	-----	Numerical value of input quantity

The files for the inputs for a sub-program must appear consecutively in the sub-program input list in the order in which the corresponding dummy variables appear in the sub-program heading; the file for the first (leftmost) input variable beginning at the initial address of the input list.

Consider the following:

DIMENSION . . . . . .X(2,4)
SUB-PROGRAM REFERENCE . . . . . .Compute INT(X(1,3), F, L, 0.5)
SUB-PROGRAM HEADING . . . . . .INT(W(I,J), G, K, T)

where:

(1) INT is a Pseudo Operation

(2) "X" is a subscripted input variable with call word, 77003.

(3) "F" is an input function with call word, 66001, and a defining equation with call word, 25005.

(4) "L" is a fixed-point input variable with value, 2.

(5) "0.5" is a floating-point constant input.

The entries in the sub-program input list and the associated dummy call words would be:

Dummy Variable	Dummy Call word*	Op	u	v	
W	76200	00	77003	77003	Call word of X in "u" and "v"
	63001	00	00000	00004	Value of multiplier for "X"
	63002	00	00000	00010	Value of modulus for "X"
I	63003	00	00000	00001	Value of first subscript for "X"
J	63004	00	00000	00003	Value of second subscript for "X"
G	61005	00	66001	66001	Call word of "F" in "u" and "v"
	61006	00	25005	25005	Call word of equation for "F" in "u" & "v"
K	63007	00	00000	00002	Value of "L"
T	63010	20	04000	00000	Floating-point constant "0.5"

All sub-program references utilize a common input region, with the current input list overlaying the previous list during running of the object program. The total number of entries in the input list for any sub-program reference must not exceed 100 octal, since only the rightmost two octal digits of the dummy call words are available for designating the relative location of an entry within the sub-program input list.

*Note: The last two octal digits of the dummy call words indicate the relative location of the entry within the input list.

4. USE OF CALL WORDS TO REFERENCE ARGUMENT LIST FOR FUNCTIONS

The dummy arguments for a function are specified on the left of the function equation and may be non-subscripted variables or subscripted variables with only one subscript. These dummy arguments will be assigned dummy call words which will be used in the generated routines to reference the arguments for the function. Argument files pertinent to the real arguments to be used in the evaluation of a function will be transferred to an Argument List for the function by the COMPUTE statement referencing the function. The rightmost two octal digits of the dummy call words indicate the relative location of the particular entry in this Argument List; the leftmost two digits specify the type of entry, i.e., subscripted or non-subscripted.

The argument file for a dummy subscripted argument will contain the call word of the real subscripted argument, the value of the modulus for the argument, and the value of the subscript. The dummy argument will be assigned a 75--- type call word, the dummy subscript and modulus 62--- type call words.

SUBSCRIPTED ARGUMENT FILE

Dummy Call word	Op	u	v	
750--	00	77---	77---	Call word of subscripted argument in "u" and "v"
620--	--	-----	-----	Numerical value of modulus
620--	--	-----	-----	Numerical value of subscript

The argument file for a dummy non-subscripted argument will contain only the value of the real argument. The dummy argument will have a 62--- type call word.

NON-SUBSCRIPTED ARGUMENT FILE

Dummy Call word	Op	u	v	
620--	--	-----	-----	Numerical value of real argument

The argument files for a given reference to a function equation must appear consecutively in the Argument List in the same order as the corresponding dummy

arguments appear on the left of the equation; the file for the first (leftmost) argument beginning at the initial address of the Argument List.

As an example, consider the following:

```
DIMENSIONS . . . . . S(10), Y(4) .
FUNCTION EQUATION . . . . . F(X(I), Z, W, Y(J)) = - - - - - .
FUNCTION REFERENCE . . . . . COMPUTE F(S(3), 0.5, T, Y(J)) .
```

where: (1) T is a floating-point variable with value, 16.0.
(2) J is a fixed-point variable with value, 2.
(3) S is a subscripted argument with call word, 77004.
(4) Y is a subscripted argument with call word, 77001.
(5) "3" is a fixed-point constant.
(6) "0.5" is a floating-point constant.

The entries in the Argument List and the associated dummy call words would be:

Dummy Variable	Dummy Call word	Argument List Op	u	v	
X	75000	00	77004	77004	Call word of "S" in "u" and "v".
	62001	00	00000	00012	Value of modulus for "S".
I	62002	00	00000	00003	Value of subscript for "S".
Z	62003	20	04000	00000	Floating-point constant "0.5".
W	62004	20	54000	00000	Value of T, i.e., floating-pt const. 16.0.
Y	75005	00	77001	77001	Call word of "Y" in "u" and "v".
	62006	00	00000	00004	Value of modulus for "Y".
J	62007	00	00000	00002	Value of Subscript "J".

A common region is used for the Argument Lists for all function equations, with each list overlaying the previous list during the running of the object program. The maximum number of entries in an Argument List is twelve, since a maximum of four arguments is allowed and the maximum argument file contains three entries.

B. Fixed Locations During Compilation

FIXED ADDRESS	TRANSLATION	GENERATION	SEGMENTATION	ALLOCATION	INITIALIZATION GENERATION	PROCESSING	LISTING
00005	00 20000 0	X 2 n n n = # words in LIST I x 00 if n = 0 x = 20 if all Library Call words > 50177 (Standard) x = 40 if all Library Call words < 50200 (Fixed) x = 60 if both types occur					

00006	00 2 n s
	n = # words Dimension List
	s = # arrays
				00 2m s
				m = # words modified Dim. List
				n = # words Dimension List
				s = # arrays
					00 2n s	00 2m s	00 2m s

00007	00 0 n n = # single-valued variables	00 0 n	00 x n x = 1st address for single valued variables n = # single-valued variables If n = 0, x = 0				
00010	00 2 n n n = # words Constant Pool	00 2 n n	00 2 n x x = 1st address of constants n = # constants If n = 0, x = 10000 (20000, 30000)				
00011	00 2 n n n = # words Reference List						
00012		00 0000x 0 x = 0 if no LIST or READ x = 1 if LIST only x = 2 if READ only x = 3 if both	00 0000x a a = Loading address for segments				
00013	00 0 n n = # blocks Corrected Problem						

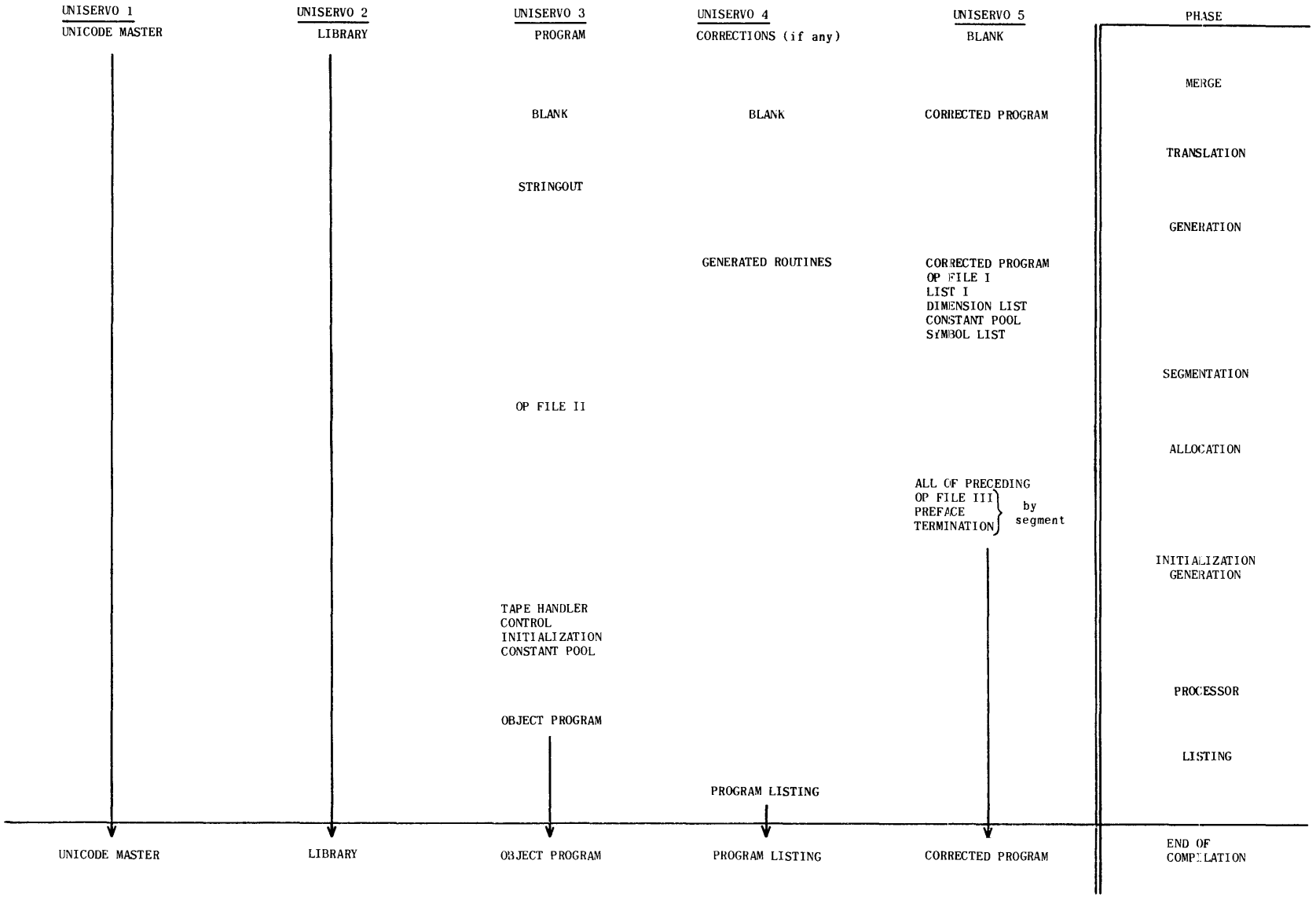
00014		u v w x y
		u = # blocks Constant Pool + Label + End
		v = # blocks DIM LIST + Label + End
		w = # blocks Symbol LIST + Label + End
		x = READ,LIST indicator
		y = # blocks thru Constant Pool					

00015				0 2n s Description as above	0 2m s		

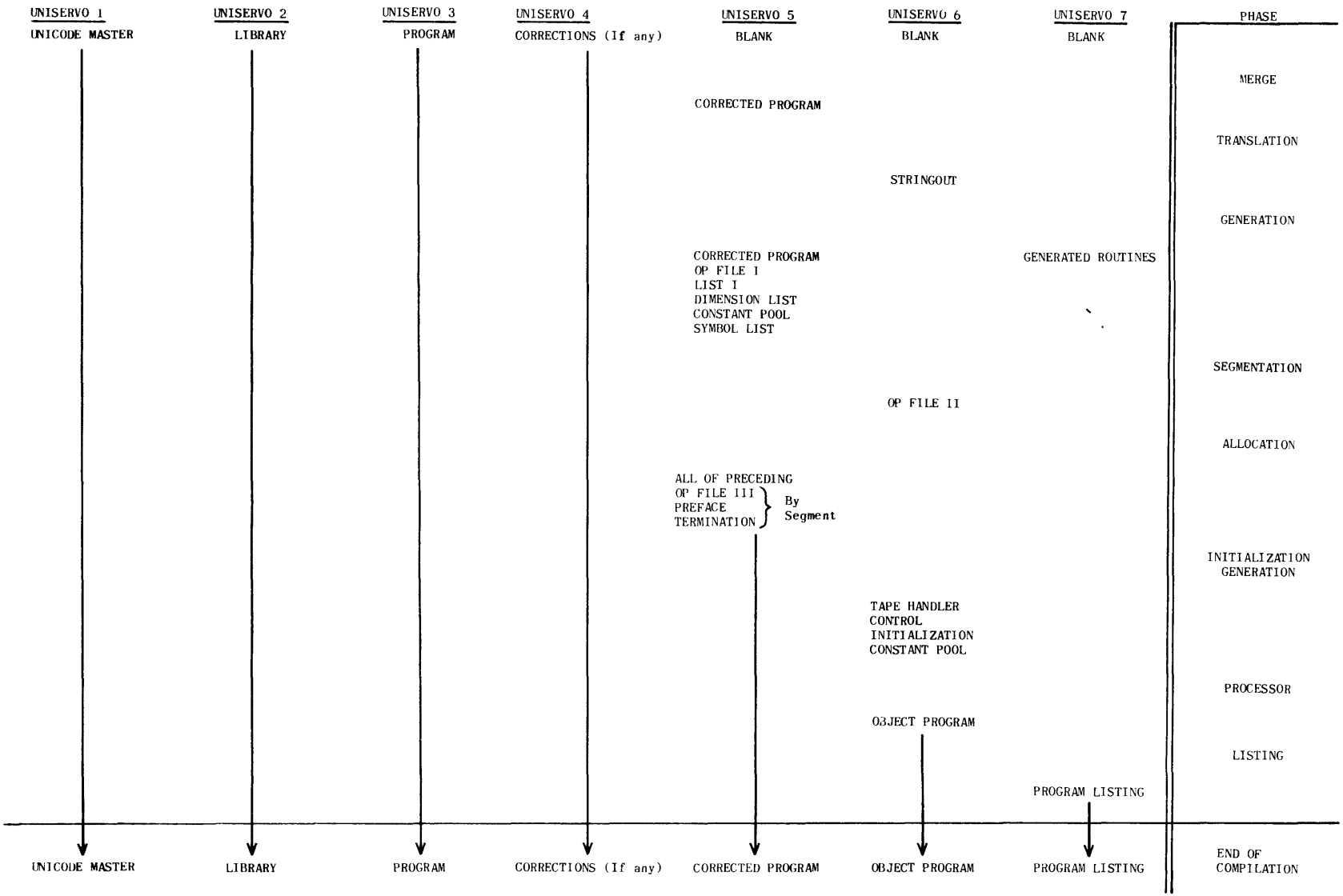
c. Uniservo Usage

On the following pages are given the two layouts of Uniservo usage during compilation of a UNICODE program; one using five Uniservos, the other seven. These charts show the important contents of each tape at the beginning of compilation and at the end of each phase. Entries only appear on the chart when the preceding phase has added to or changed the contents of the tape. The bottom line shows the contents of all tapes at the end of compilation.

UNICODE COMPILATION - Five Uniservo Layout



UNICODE COMPILATION - Seven Uniservo Layout



d. Corrections to UNICODE Manual (U-1451 Rev. 3)

Page 4: Insert after tenth line of first paragraph the following:

This size restriction on fixed-point constants written in a UNICODE program does not prevent the generation, use, or output of fixed-point numbers greater than 999999 in the running of an Object Program.

Page 7: Change first line of third paragraph to read:

Superior symbols available are minus sign, divide sign, and figures, all superior

Page 10: Change first two lines of list under first paragraph opposite "Hierarchy of Operations" title to read as follows:

1. Library Functions
2. Exponentiation

Add paragraph at bottom of page 10:

$Z = \sin y^2$ is interpreted as $Z = (\sin y)^2$. A term such as $\sin^2 y$ causes an error print-out. To get the sine of y^2, one must write $\sin (y^2)$. Similar usage applies to other library functions.

Page 23: Change second and third line from the bottom of page to read:

DOT (A(I), B(J))

where A(I) and B(J) are dummy vectors. No "dummy" operand in

Also add at the bottom of the page the sentence:

No dummy operand (including subscripts) may have the same designation as another dummy operand in the same heading, although different dummies may be equated to the same real variable by the COMPUTE statement which references the sub-program.

e. Use of Second and Third Memory Cores in Object Program

To modify the UNICODE Master Tape so that more than one core of 10000_8 addresses is available to the Object Program, update a Master Tape until the number 13 for the Segmentation Phase has been typed. Stop the updating at this point and load into absolute address 2555 the upper threshold desired. At present this address holds 00 00000 10000. It may be changed to 00 00000 20000 for 2 cores or 00 00000 30000 for 3 cores or any value above 10000 for Object Programs needing more than one core but less than the full memory.

After this correction, resume the updating of the Master Tape as explained in the System Tape Package write-up by starting at 40414. Upon completion of the updating, a changed-word post mortem (start at 40600) should show only the changing of this one word. If it does, the updated Master Tape will now generate Object Programs using the new memory area designated.

III. TRANSLATION AND CORRECTION

1. UNICODE SENTINEL BLOCKS

III. Translation and Correction

1. UNICODE Sentinel Blocks (ZZ)

The first two blocks of the system tape contain coding to:

A. Read in the merge and jump to it (start at 77000)

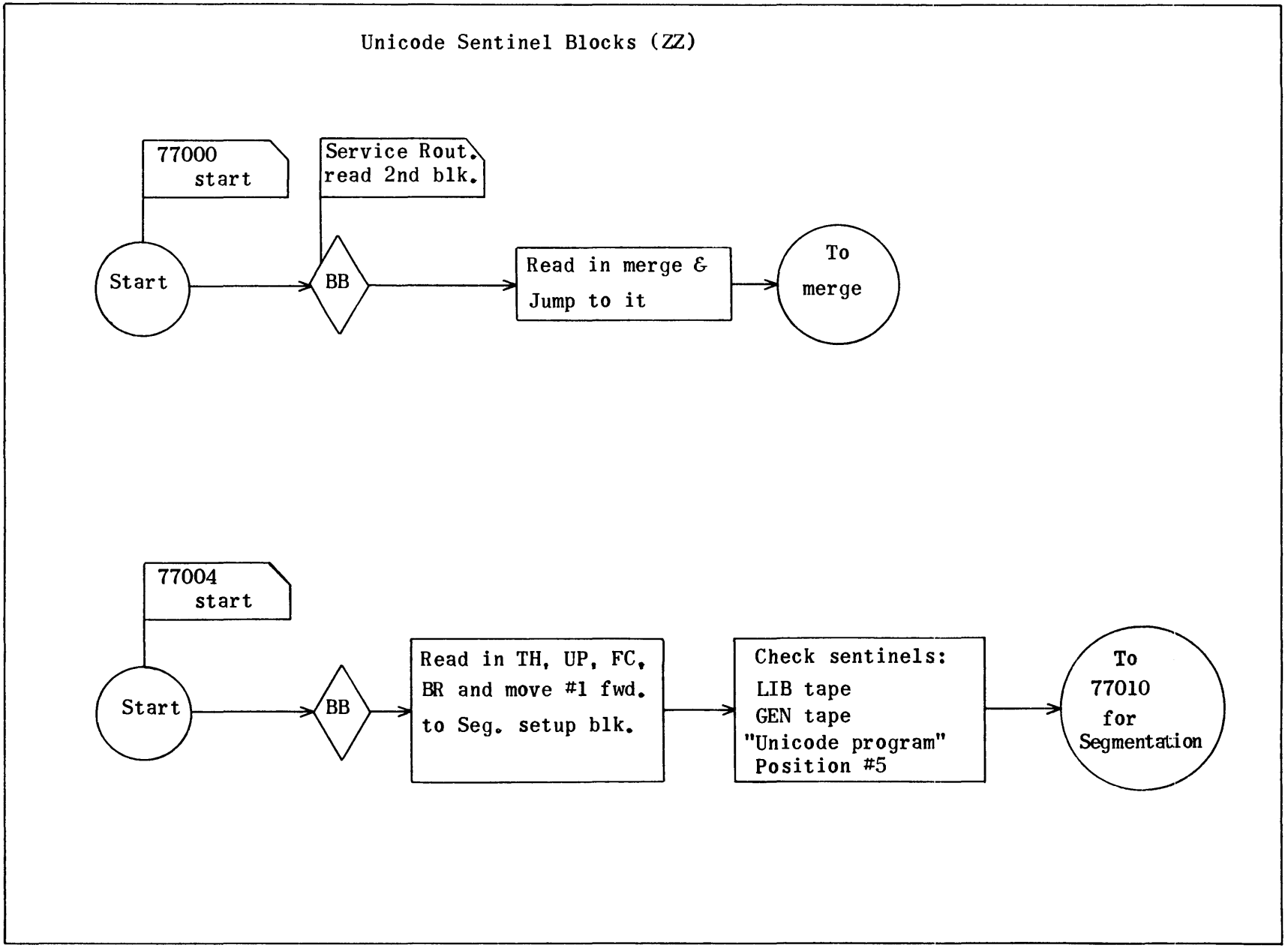
B. If start was at 77004, read in:

1. Flex Codes (FC)
2. Tape Handler (TH)
3. Print Text (UP)
4. Alarm Routine

then move tape on Uniservo 1 forward to Segmentation Set-Up and check Sentinels on following tapes:

1. Library (#2)
2. Generated Routines (#4 or #7)
3. "UNICODE Program" (#5)

then move #5 forward to Op. File I and jump to 77010. This will read in Segmentation Set-Up and jump to it, thus initiating Pass III.


Unicode Sentinel Blocks (ZZ)
77000
start
Service Rout.
read 2nd blk.
Start
BB
Read in merge &
Jump to it
To
merge
77004
start
Start
BB
Read in TH, UP, FC,
BR and move #1 fwd.
to Seg. setup blk.
Check sentinels:
LIB tape
GEN tape
"Unicode program"
Position #5
To
77010
for
Segmentation

Unicode System Tape Sentinel Blocks
(1105 & 1103A Regions and Coding)

Region	Address	Name
ZZ	7230	Sentinel and Reads
CC	7267	Constants
CE	7344	
ZU	7407	Check Library Sentinel
ZW	7437	Move No. 5 forward
ZX	7444	Check for "Unicode Program"
ZY	7474	Check Generated Tape Sentinel
ZC	7534	Read N blocks
TI	7610	Tape Image
BB	7116	Read Subroutine
TH	21	Tape Handler
TN	20	Indicates 5 or 7 servos
UP	421	Print text
UW	513	
FC	40001	Flex Codes
ZA	77000	Service Routine
XA	35	Beginning of Merge
XB	3134	End of Merge
BR	537	Start of Generation subs

Unicode System Tape Sentinel Blocks
(1103A)

```
     IA   ZZ                    = 7230
0    MJ   0       ZZ6           77000 Start
1    MJ   0       ZZ13          77004 Start
2    67   50342   65127         U N   I C  O  D
3    30   77657   36566         E 77  S Y  S  T
4    30   47776   62452         E M  77 T  A  P
5    30   77777   77777         E 77 77 77 77 77
6    TP   CC1     BB2  }
7    RJ   BB      BB1  }        Read in subs
10   TP   CC2     ZC2  }
11   RJ   ZC      ZC1  }        Read in merge
12   MJ   0       XA            --> Merge
13   TP   CC1     BB2  }
14   RJ   BB      BB1  }        Read in subs
15   EF   0       CC3           Move fwd to FC
16   TP   CE42    BB2  }
17   RJ   BB      BB1  }        Read in FC
20   RP   30100   ZZ22 }
21   TP   TI      FC   }        FC --> drum
22   TP   CC4     ZC2  }
23   RJ   ZC      ZC1  }        Read in GTH, UP
24   TP   CE31    TH3  }
25   RJ   TH2     TH   }        Move fwd to BR
26   TP   CE30    TH3  }
27   RJ   TH2     TH   }        Read in BR
30   TP   CE27    TH3  }
31   RJ   TH2     TH   }        Move fwd to seg. setup
32   MJ   0       ZU            --> LIB check
     CA   ZZ33
```

Unicode System Tape Sentinel Blocks
(1105)

```
    IA   ZZ                    = 7230
 0  MJ   0       ZZ6           77000 Start
 1  MJ   0       ZZ13          77004 Start
 2  67   50342   65127         U  N   I   C   O   D
 3  30   77657   36566         E  77  S   Y   S   T
 4  30   47776   62452         E  M   77  T   A   P
 5  30   77777   77777         E  77  77  77  77  77
 6  TP   CC1     BB2    }
 7  RJ   BB      BB1    }      Read in subs
10  TP   CC2     ZC2    }
11  RJ   ZC      ZC1    }      Read in merge
12  MJ   0       XA            ⟶ Merge
13  TP   CC1     BB2    }
14  RJ   BB      BB1    }      Read in subs
15  MJ   20000   ZZ20          TCU2 → ZZ20  TCU1 ↓
16  EF   0       ZZ36          Move fwd to FC
17  MJ   0       ZZ21          Skip 20
20  EF   0       CC3           Move fwd to FC
21  TP   CE42    BB2    }
22  RJ   BB      BB1    }      Read in FC
23  RP   30100   ZZ25   }
24  TP   TI      FC     }      FC ⟶ drum
25  TP   CC4     ZC2    }
26  RJ   ZC      ZC1    }      Read in GTH, UP
27  TP   CE31    TH3    }
30  RJ   TH2     TH     }      Move fwd to BR
31  TP   CE30    TH3    }
32  RJ   TH2     TH     }      Read in BR
33  TP   CE27    TH3    }
34  RJ   TH2     TH     }      Move fwd to seg. setup
35  MJ   0       ZU            ⟶ LIB Check
36  1    4       10153         Move fwd to FC  TCU1
    CA   ZZ37
```

Check LIB Tape
(1103A & 1105)

```
     IA    ZU
 0   TP    CC5     TH3     }
 1   RJ    TH2     TH      }      Read LIB sent.
 2   TP    TI24    A       )
 3   EJ    CC6     ZU5     )
 4   MJ    0       ZU7     }      Check LIB sent.
 5   TP    TI25    A       )      Word 21 = Δ Δ L I B Δ
 6   EJ    CC7     ZU16    )      Word 22 = Δ T A P E Δ
                              Error ↓ O.K. → ZU16
 7   TP    CC10    TH3     }
10   RJ    TH2     TH      }      Rewind #2
11   TP    CC35    UP3     }
12   RJ    UP2     UP      }      "LIB on 2"
13   TP    CC      A              A = 0
14   MS    0       ZU15
15   ZJ    ZY      ZU             Not required → GEN check
16   TP    CC5     TH3     }
17   RJ    TH2     TH      }      Read 2nd blk. of LIB tape
20   TP    TI2     A       )
21   DV    CC12    Q       )
22   ZJ    ZU23    ZU24    }      No. blks. in catalog → A
23   RA    Q       CC13    )
24   SP    Q       25      )
25   AT    CC11    TH3     }
26   RJ    TH2     TH      }      Move past LIB catalog
27   MJ    0       ZY             → GEN sent. check
     CA    ZU30
```

Check Generation Tape Sentinel
(1103A & 1105)

```
     IA   ZY
 0   TP   CC15   TN          TN = 0 3 0
 1   MJ   10000  ZY3         Five servos ↓ 7 → ZY3
 2   TP   CC     TN          TN = 0
 3   TP   TN     A
 4   AT   CC16   TH3   }     Read GEN sent.
 5   RJ   TH2    TH    }
 6   TP   CC20   A     )     Rewind gen. tape
 7   AT   TN     TH3   |
10   RJ   TH2    TH    )
11   TP   CC17   A     }     Sent. O.K. → ZY21  No ↓
12   EJ   TI24   ZY21  }
13   TP   CC22   CC33        Set for 7 servo print
14   MJ   10000  ZY16        Five servos ↓ 7 → ZY16
15   TP   CC21   CC33        Set for 5 servo print
16   TP   CC23   UP3   }     "GEN tape not on servo 4 or 7"
17   RJ   UP2    UP    }
20   MS   0      ZY          Stop for re-entry
21   TP   TI25   CE          Sent. O.K., store # blk. on 5
22   TP   CE25   TH3   }     Read 1 blk. #5
23   RJ   TH2    TH    }
24   RJ   ZX     ZX1         Pick up Sentinel
25   RP   30003  ZY27  )
26   CC   CE3    CE6   |
27   SP   CE3    0     |     Check Sentinel
30   SA   CE4    0     |
31   SA   CE5    0     |
32   ZJ   ZY33   ZW    )     No ↓ O.K. → ZW
33   TP   CE24   TH3   }     Rewind #5
34   RJ   TH2    TH    }
35   TP   CE32   UP3   }     "Mount Unicode Program #5 etc."
36   RJ   UP2    UP    }
37   MS   0      ZY22
     CA   ZY40
```

Check "Unicode Program"
(1103A & 1105)

```
     IA   ZX
 0   MJ   0        (30000)      Exit
 1   TU   CE11     ZX7          }
 2   TV   CE11     ZX20         }  Setup
 3   TP   CC       CE1          }
 4   RP   30004    ZX6          }
 5   TP   CE17     CE13         }  Set indexes
 6   TP   CE20     CE14         One word index
 7   LQ   (30000)  6            }
10   QT   CE23     CE2          }  Char. ⟶ CE2
11   EJ   CC13     ZX24         Δ ⟶     No ↓
12   IJ   CE15     ZX14         }  14 non Δ's ⟶
13   MJ   0        ZX24         }           No ⟶
14   SP   CE1      6            }
15   AT   CE2      CE1          }  Collect char. in CE1
16   TP   CE1      CE5          }
17   IJ   CE16     ZX24         CE1 full ↓ No ⟶
20   TP   CE1      (30000)      Store CE1
21   RA   ZX20     CC13         Modify
22   TP   CC       CE1          Clear temp
23   TP   CE22     CE16         Reset index
24   IJ   CE14     ZX7          One word index
25   RA   ZX7      CE12         Modify
26   IJ   CE13     ZX6          20 word index
27   MJ   0        ZX           Exit
     CA   ZX30
```

Position #5
(1103A & 1105)

```
     IA   ZW
 0   RS   CE       CC13         # blocks on 5
 1   SP   A        25
 2   AT   CE26     TH3          Move #5 fwd to op  file I
 3   RJ   TH2      TH
 4   MJ   0        ZA10         ⟶ seg, setup
     CA   ZW5
```

Constants and Variables
(1103A & 1105)

	IA	CC		
0	0	0	0	Zero
1	0	170	ZZ170	Read in subs
2	0	XA	XB	Read in merge
3	2	4	10153	Move fwd to FC TCU2 & 1103A
4	0	TH	UW23	Read in GTH, UP, etc.
5	50	102	TI	Read LIB sent. & 2nd blk.
6	01	01463	42501	Δ Δ L I B Δ
7	01	66245	23001	Δ T A P E Δ
10	10	2	0	Rewind #2
11	30	2	0	Move past LIB catalog
12	0	0	170	
13	0	0	1	
14	0	0	07777	Mask
15	0	3	0	Set TN
16	50	104	TI	Read GEN sent. (4 or 7)
17	01	01323	05001	ΔΔGEN Δ
20	10	4	0	Rewind GEN (4 or 7)
21	07	22010	16566	4 (print with 5 servos)
22	12	22010	16566	7 (print with 7 servos)
23	0	CC24	11	Par. for print GEN sent. error
24	47	51675	06601	M O U N T Δ
25	67	50342	65127	U N I C O D
26	30	01323	05030	E Δ G E N E
27	54	24663	02701	R A T E D Δ
30	54	51676	63450	R O U T I N
31	30	65015	15001	E S Δ O N Δ
32	65	30547	05101	S E R V O Δ
33	0	0	0	4 . Δ Δ S T filled in
34	24	54662	27777	A R T . 77 77
35	0	CC36	17	
36	47	51675	06601	M O U N T Δ
37	67	50342	65127	U N I C O D
40	30	01463	42554	E Δ L I B R
41	24	54730	15150	A R Y Δ O N
42	01	65305	47051	Δ S E R V O
43	01	05220	10134	Δ 2 . Δ Δ I
44	31	01505	10146	F Δ N O Δ L
45	34	25542	45473	I B R A R Y
46	01	54305	36734	Δ R E Q U I
47	54	30270	16530	R E D Δ S E
50	66	01240	15051	T Δ A Δ N O
51	66	01760	17430	T Δ = Δ Z E
52	54	51220	10101	R O . Δ Δ Δ
53	01	01016	56624	Δ Δ Δ S T A
54	54	66227	77777	R T . 77 77 77
	CA	CC55		

Constants & Variables (Cont.)

	IA	CE			
0	0	0	(0)		# blk. servo 5
1	0	0	0		Build word
2	0	0	0		Char.
3	0	0	3	}	U N I C O D } build
4	0	0	4	}	E P R O G R } build
5	0	0	5		0 0 0 0 A M
6	67	50342	65127		U N I C O D
7	30	52545	13254		E P R O G R
10	0	0	02447		0 0 0 0 A M
11	0	TI	CE3		
12	0	1	0		
13	0	0	0	)	
14	0	0	0	}	Indexes
15	0	0	0	}	
16	0	0	0	)	
17	0	0	23	)	
20	0	0	5	}	Set indexes
21	0	0	16	}	
22	0	0	5	)	
23	0	0	77		Mask
24	10	5	0		Rewind #5
25	50	105	TI		Read 1 blk. #5
26	30	5	0		Move #5 fwd n blks. to Op File
27	30	501	0		Move #1 fwd to segmentor
30	50	101	BR		Read in BR
31	30	11001	0		Move #1 fwd to BR routine
32	0	CE33	7		
33	47	51675	06601		M O U N T Δ
34	67	50342	65127		U N I C O D
35	30	01525	45132		E Δ P R O G
36	54	24470	15150		R A M Δ O N
37	01	65305	47051		Δ S E R V O
40	01	10220	10165		Δ 5 . Δ Δ S
41	66	24546	62277		T A R T . 77
42	0	170	TI		Read in FC
	CA	CE43			

Read n Blocks to Storage
(1103A & 1105)

	IA	ZC			
0	MJ	0	(30000)		Exit
1	MJ	0	ZC3		
2	0	(30000)	(30000)		0 1st address last address
3	TP	ZC2	Q		Par. → Q
4	QT	ZC34	ZC35		Store 1st address in u
5	LQ	Q	17		Last address → Q_u
6	QT	ZC34	A		Last address → A_u
7	ST	ZC35	A	}	Last - 1st + 1 → A_u
10	AT	ZC37	A	}	
11	LQ	Q	6		1st add. → Qv
12	TV	Q	BB2		Set up 1st address
13	DV	ZC40	ZC33		$n/170_8$ → index
14	TP	A	ZC32		Store remainder
15	TP	Q	A	}	Quotient = 0 → ZC26 no ↓
16	ZJ	ZC17	ZC26	}	
17	RS	ZC33	ZC36		Index - 1
20	TU	ZC40	BB2		No. of words = 170_8
21	RJ	BB	BB1		Read blk
22	RA	BB2	ZC31		Address + 170_8
23	IJ	ZC33	ZC21		n blks
24	TP	ZC32	A	}	Remainder = 0 → exit no ↓
25	ZJ	ZC26	ZC	}	
26	TU	ZC32	BB2		Remainder → par.
27	RJ	BB	BB1		Read remainder
30	MJ	0	ZC		Exit
31	0	0	170		Constant
32	0	0	0		Remainder
33	0	0	0		Index
34	0	77777	0		Mask
35	0	(30000)	0		1st address
36	0	0	1		Constant
37	0	1	0		
40	0	170	0		
	CA	ZC41			

2. TAPE MERGE

2. Tape Merge

The tape merge routine is used to produce a corrected program tape, or, if no corrections are desired, to copy the program onto Uniservo 5. MJ3 is used to determine which of the two options is taken. If it is set, a corrected tape is produced; if not, the tape is copied.

Corrected Tape

Correction tape is read in 16_8 blocks at a time. While corrections are being read in, a table is built up which contains an entry for each correction item.

For a change or insert sentence, the table entry consists of the line number and a directory word which contains the drum address where the item will be stored, and the number of words the correction contains. The sign bit of the directory word is set to negative so that the entry will be recognized as a change or an insert.

For a delete sentence, the table entry contains the first and last line number of the group of items to be deleted.

The Line-Number processor is used to process all line numbers in the table. The line numbers of insert and change items are processed between read-in of successive blockettes and the processed line number included in the table entry. The line numbers of delete items are processed after 16_8 blocks have been read in and tape has been stopped.

Since there is not enough time available between blockettes to print out errors, a modified version of the Line Number Processor is used. When an error is encountered, the line number and the type of error are stored so that errors can be printed out after the tape is stopped. Also, the output given by Line-Number Processor for an illegal line number is set to zero so that the line will not appear on the corrected tape.

After 16_8 blocks have been read in, the corrections are transferred to the drum, the line numbers of delete items are processed, and any Line-Number Processor errors are printed out.

If more corrections remain to be read in, the routine repeats the process.

When all corrections have been read in and all table entries have been formed, the table is sorted so that the corrections can be brought down from drum in ascending order. At this time, all table entries with item numbers of zero are thrown away.

After the table has been sorted, read-in of program tape is begun. Items are read in from the program tape seven blocks at a time. Line numbers are processed between the read-in of successive blockettes.

When seven blocks have been brought in from program tape, the read is stopped. Next all change and insert items referring to the program items are transferred from the drum to the correction buffer. The line-number position of all sentences to be deleted is filled with zero.

Next the routine starts to write the corrected program onto tape 5. All insertions and changes are made as the tape is written. Program items with line numbers of zero are not written out on corrected tape.

When the last complete block which can be produced from the group of items in core has been written, the tape is stopped and any remaining blockettes are stored to be written on the next pass. The routine then goes back to read more items from program tape.

When no more program or correction sentences remain to be written, the routine fills in the final block with END OF TAPE sentences and stops the tape. Next all tapes are rewound and the routine exits to ZA10.

Copy Tape

Program tape is read in 24_8 blocks at a time and written on tape 5 until all of tape has been copied. Then tapes are rewound and routine exits to ZA10.

Print-Outs

If merge is selected, routine prints:

MERGE
CORRECTION TAPE

Following the correction tape print-out, all errors detected on correction tape are listed. Next the routine prints:

PROGRAM TAPE

and all errors occurring on program tape are listed.

When either the merge or the copy tape is completed, the routine prints

PROGRAM NOW ON TAPE 5

If only 5 servos are available, the routine prints

PUT 1500 FT. TAPES ON S. 3 AND 4

and stops so tapes can be changed.

When the computer is restarted, the routine exits to ZA10.

Error Print-Outs

If an out-of-sequence item appears on the program tape, the line-number position of the sentence is filled with a zero so that the sentence will be dropped and the routine prints

SENTENCE ' - - - ' OUT OF SEQUENCE

Label blocks of both the correction tape and the program tape are checked to determine whether correct tape is being read. If label is incorrect, routine prints:

WRONG TAPE SERVO 3 (or 4) SHOULD BE UNICODE PROGRAM
(or UNICODE CORRECTION)

After the Error print-out, the routine rewinds the tape and stops set to re-enter after tape has been changed.

If a parity error occurs, and the parity reread routine fails to correct the error, the routine prints:

PARITY REREAD FAILS. TO REREAD START. TO IGNORE SET
A NOT = 0 AND START.

and stops set to re-enter.

If parity error is ignored, routine prints:

PARITY ERROR IGNORED

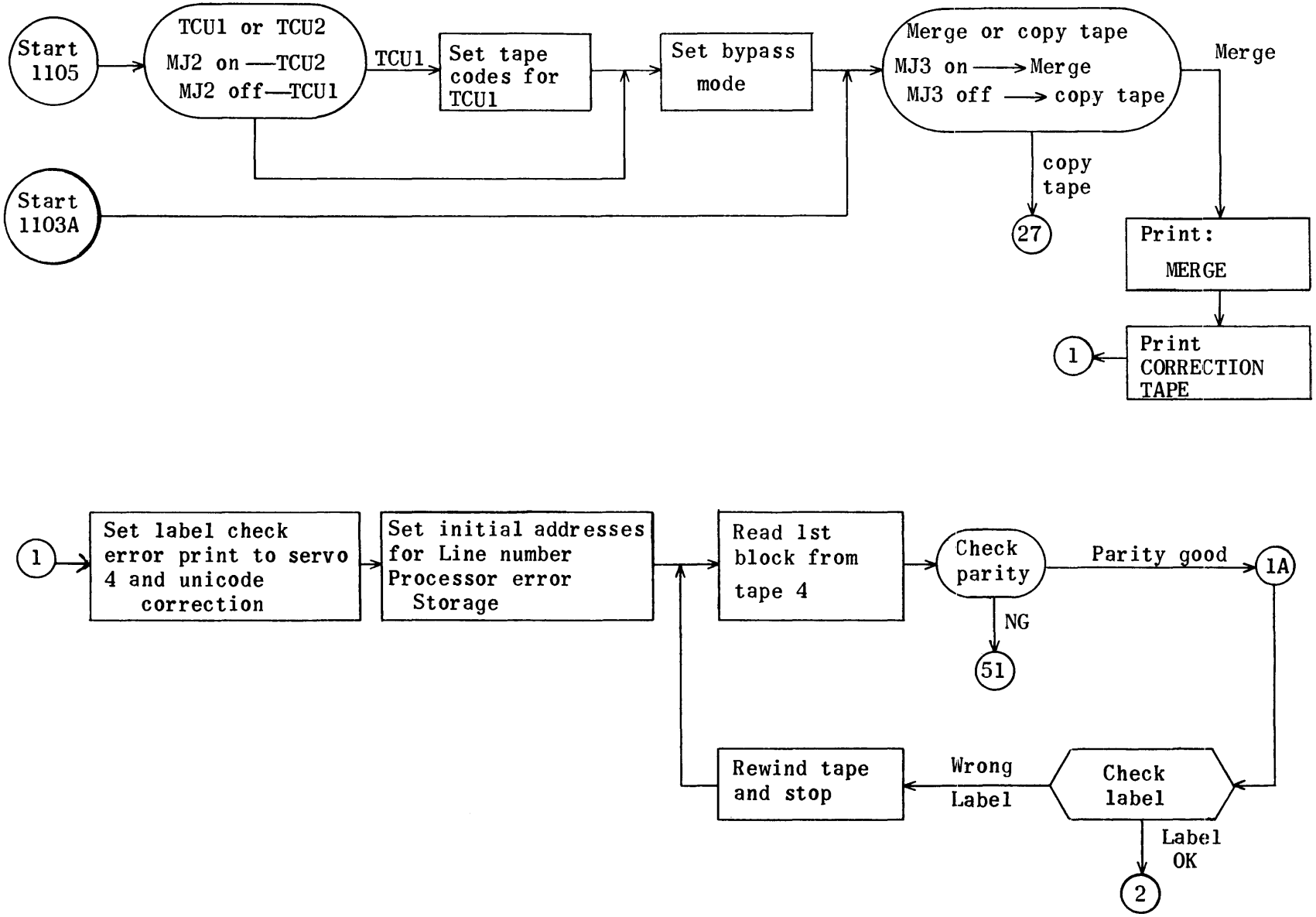
repositions tape and goes on with next block.

If more than 300 corrections appear on correction tape, the routine prints:

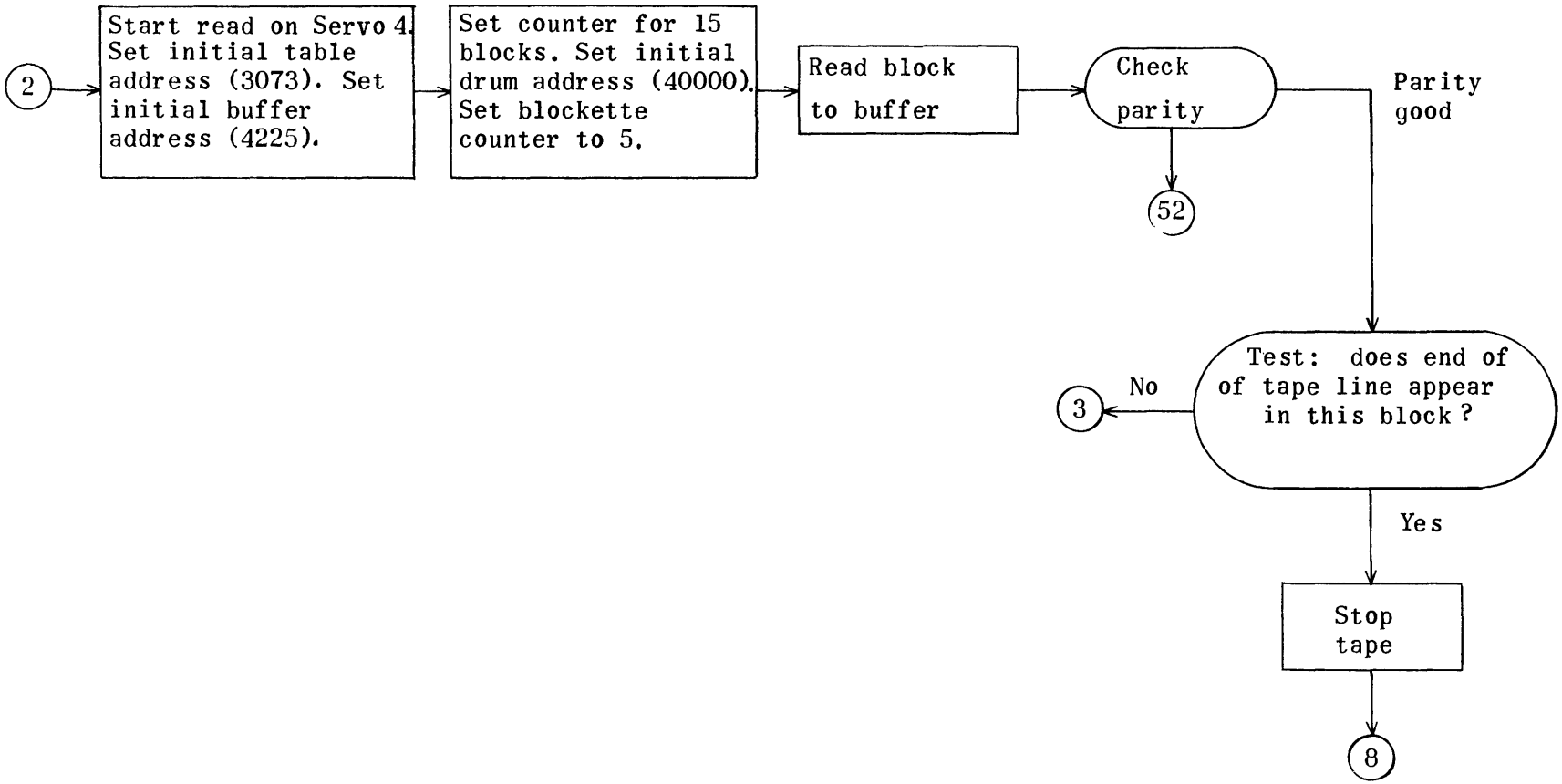
TOO MANY CORRECTIONS, 300 IS MAXIMUM

After print-out, routine ignores any remaining corrections and goes on to write corrected tape.

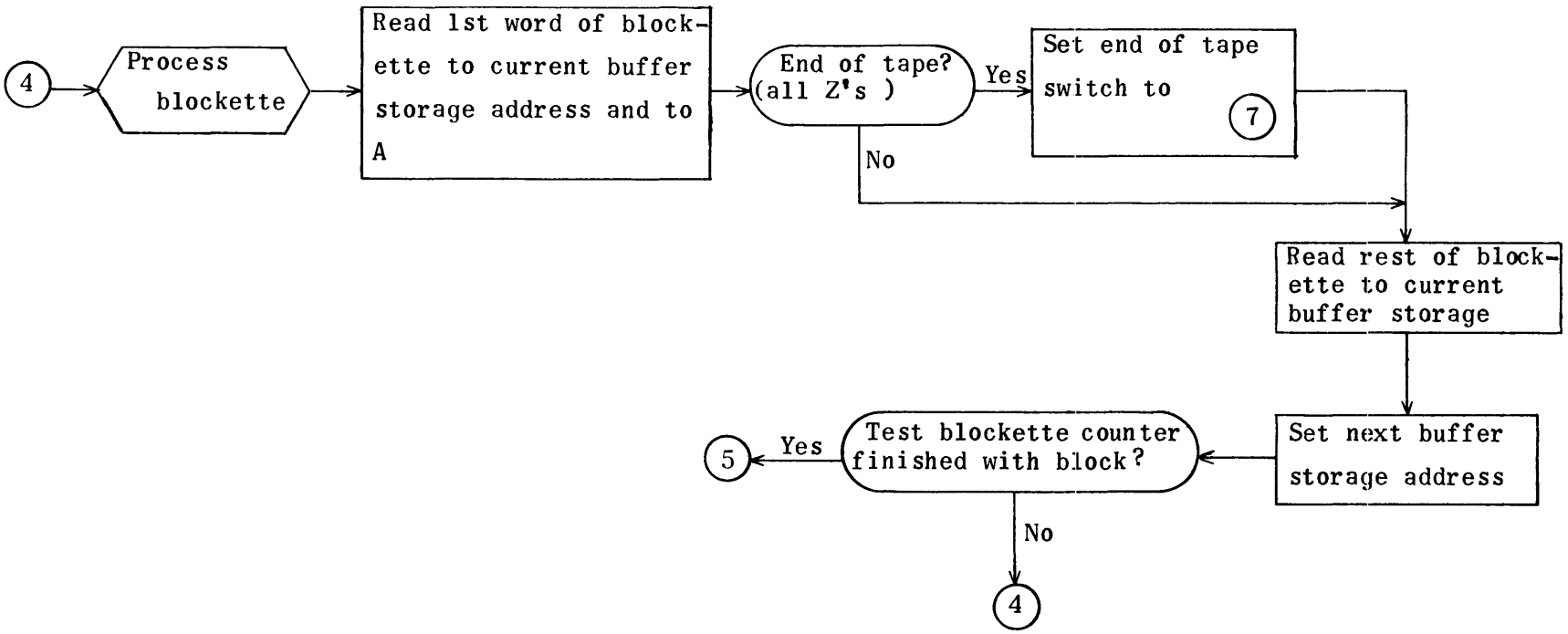
Merge Routine



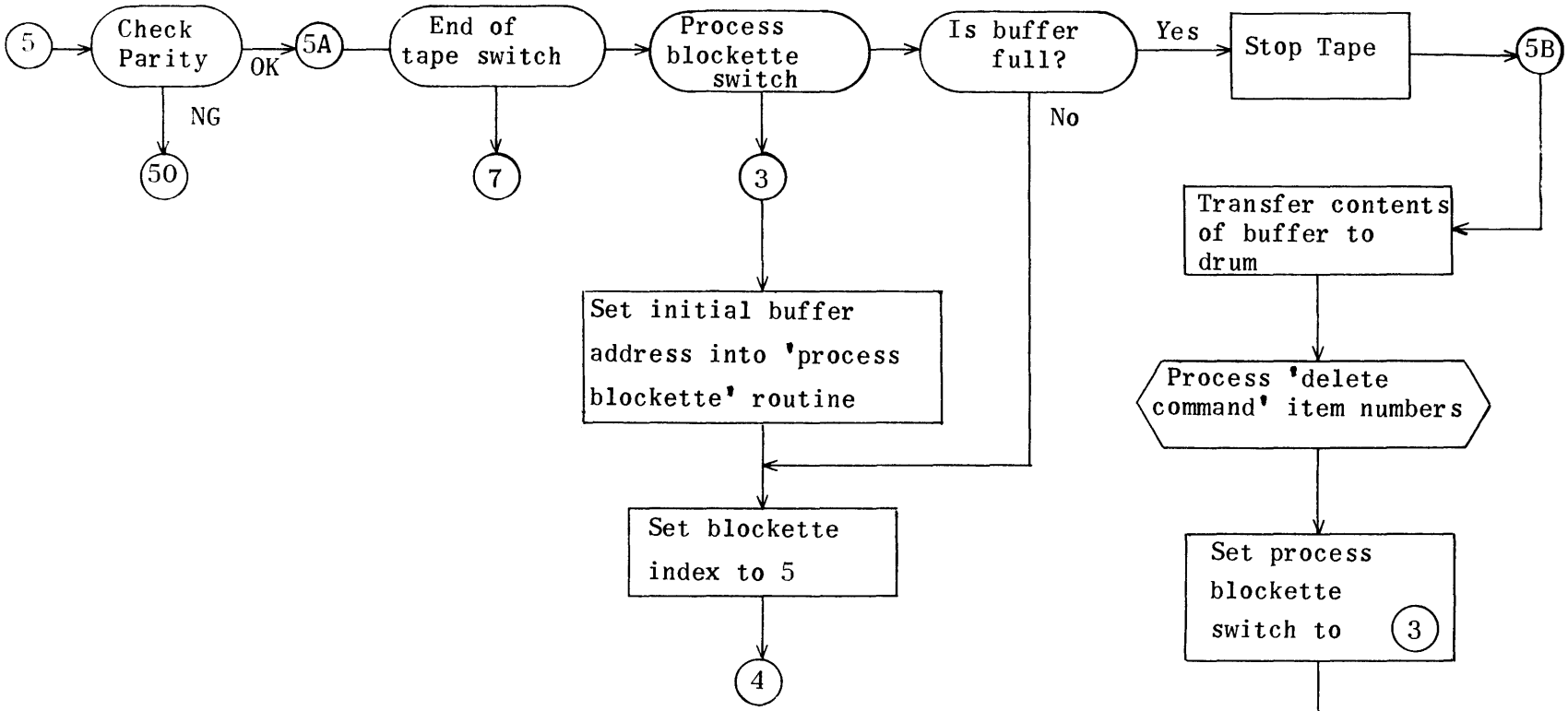
Merge Routine (Cont.)

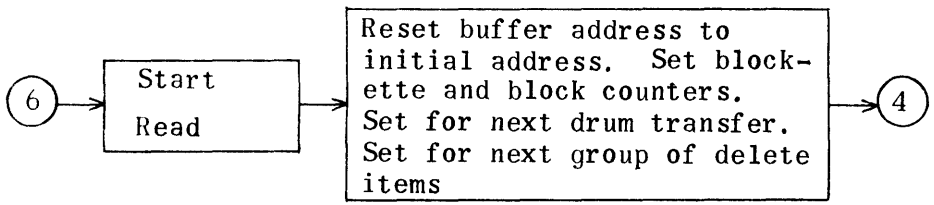


Merge Routine (Cont.)

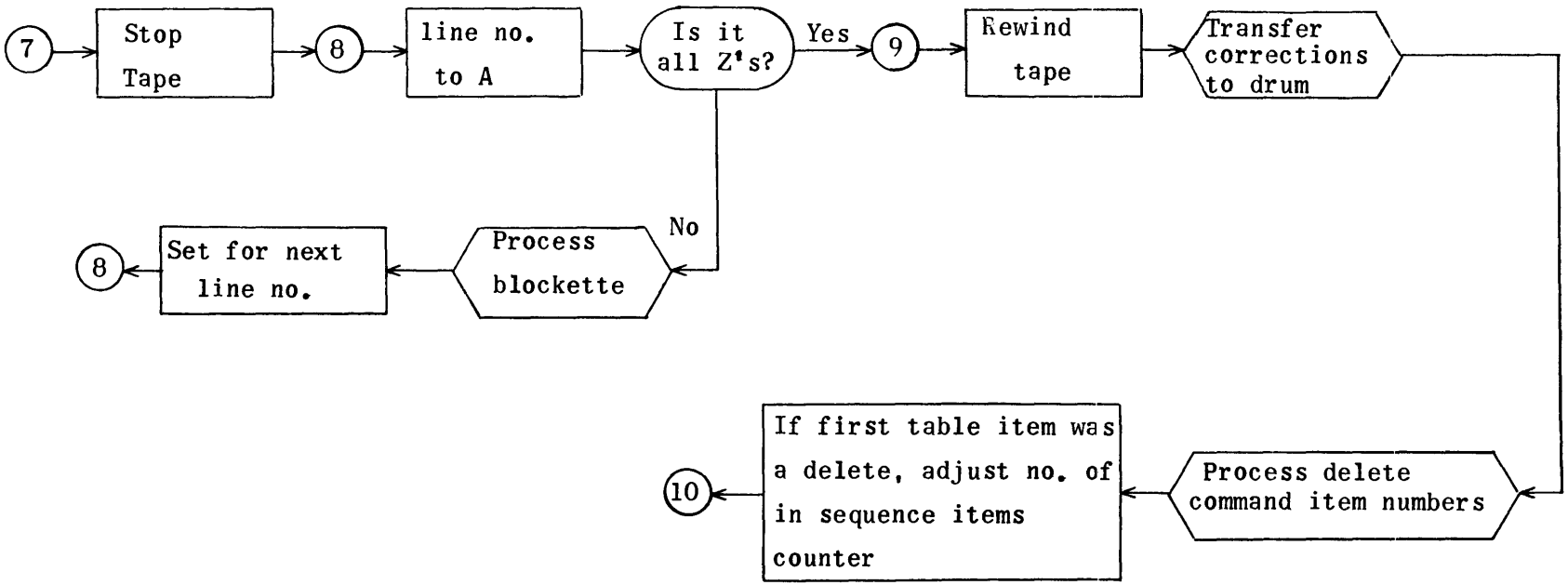


Merge Routine (Cont.)

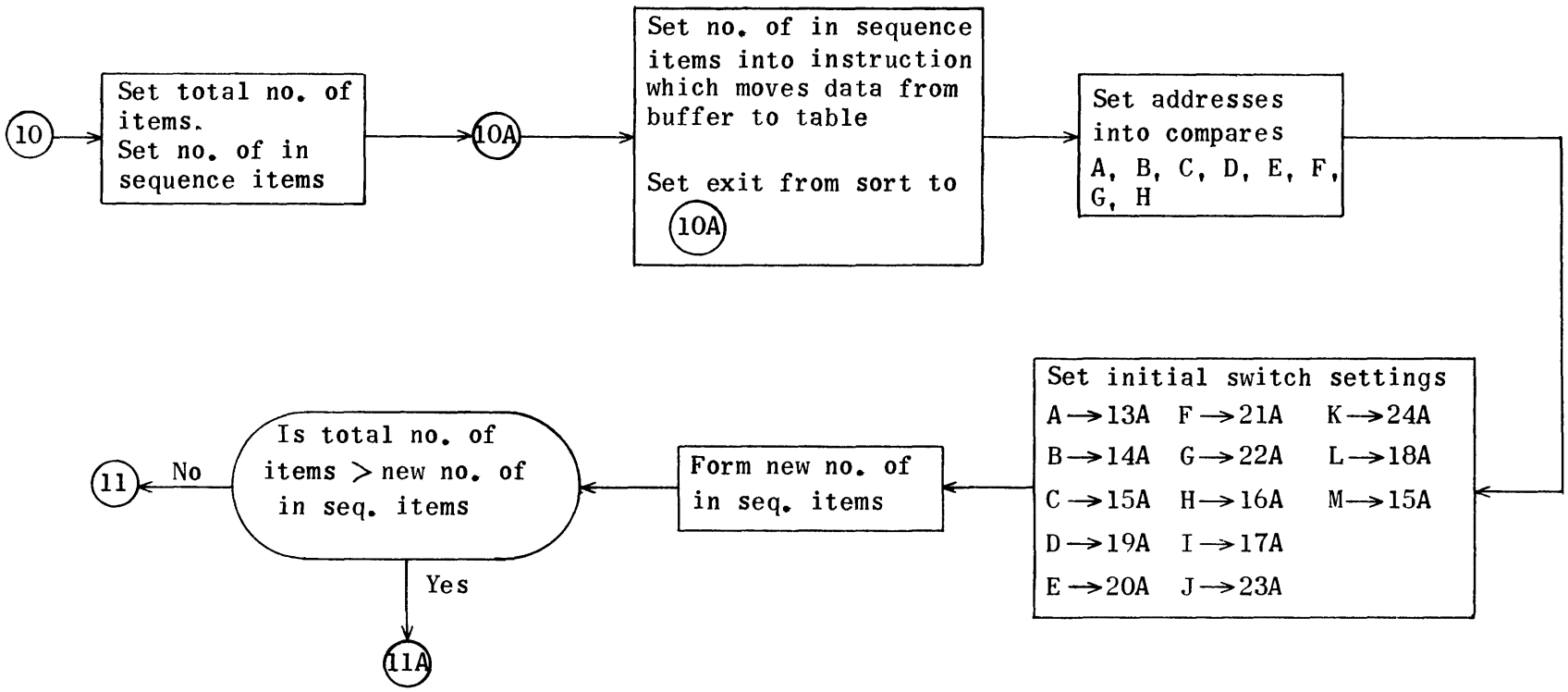




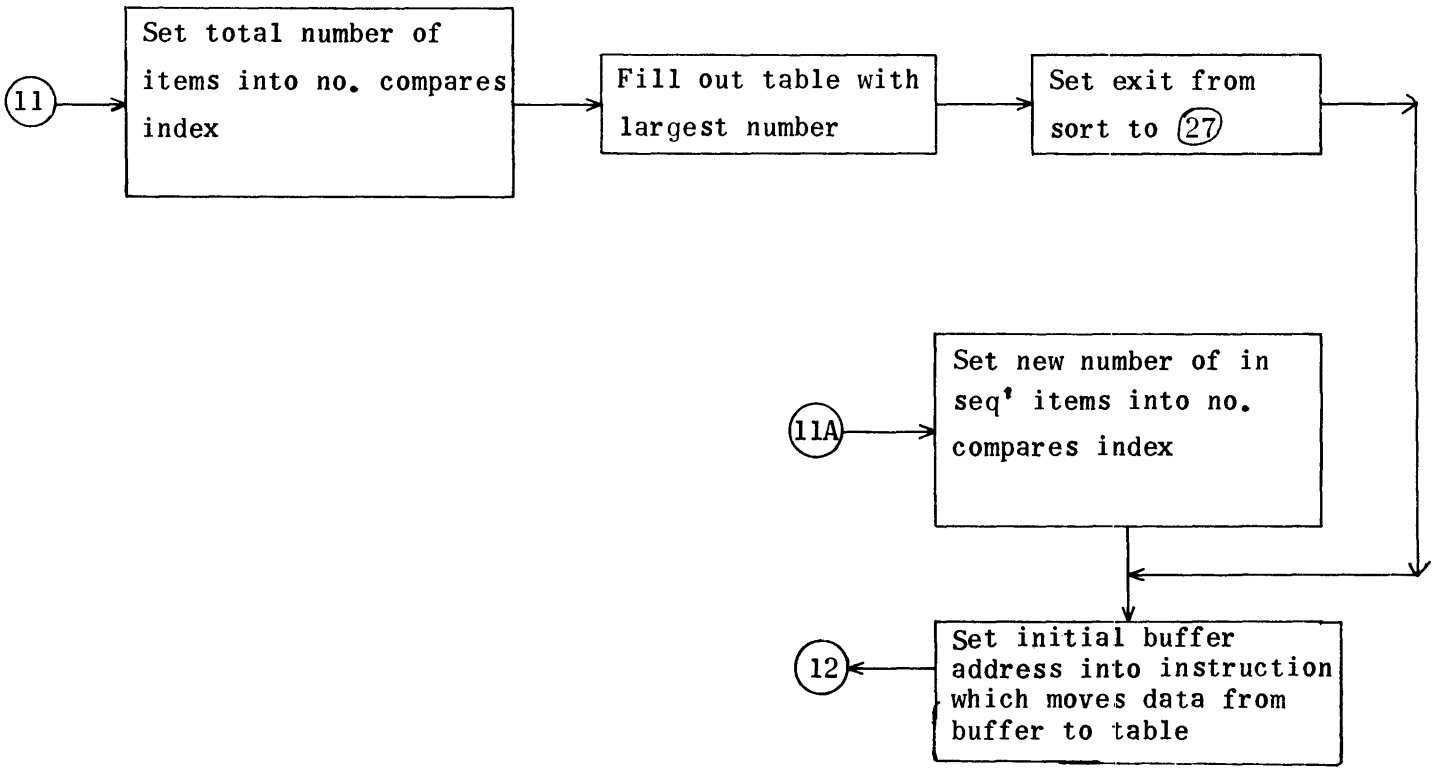
Merge Routine (Cont.)



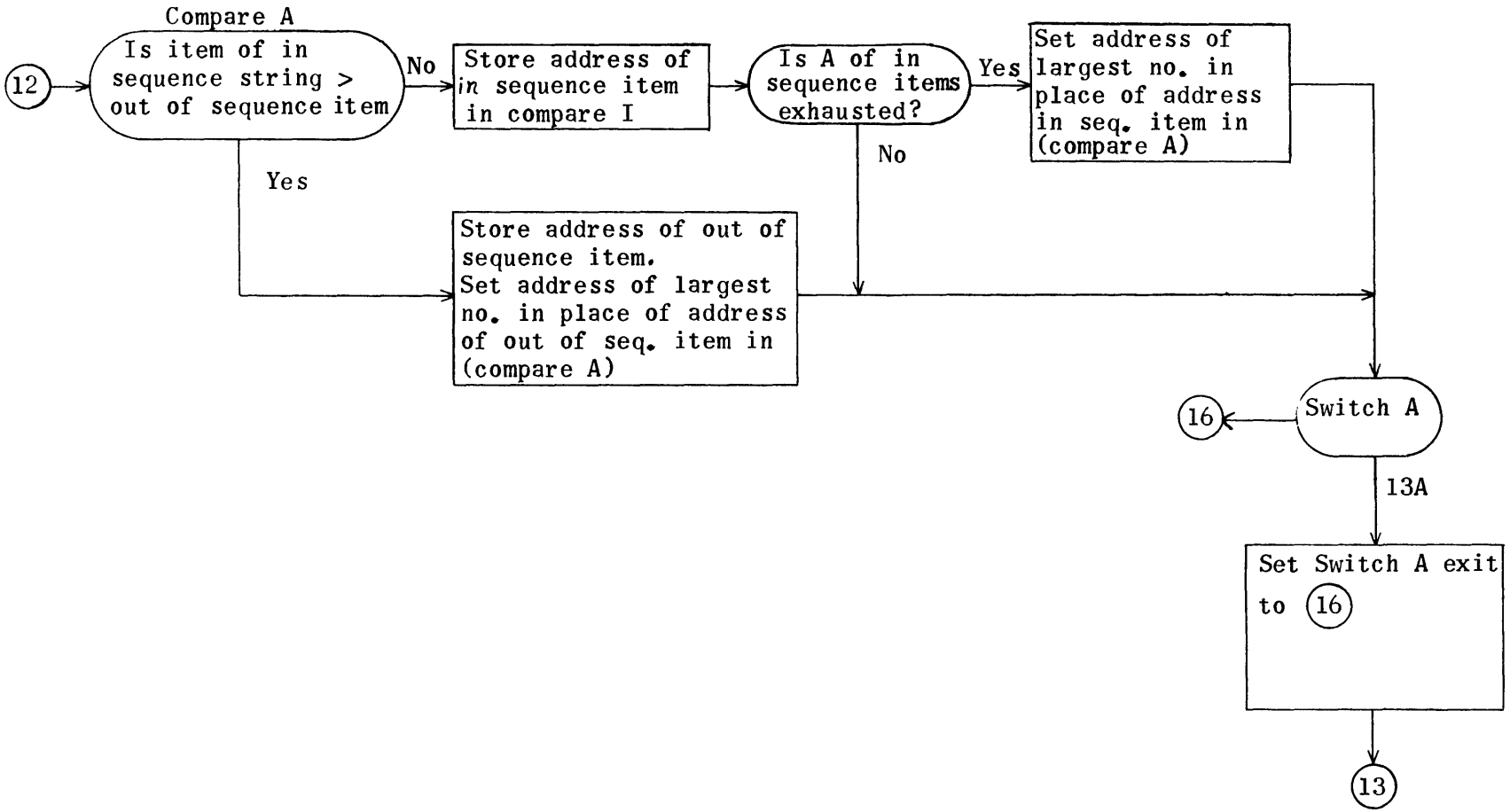
Merge Routine (Cont.)

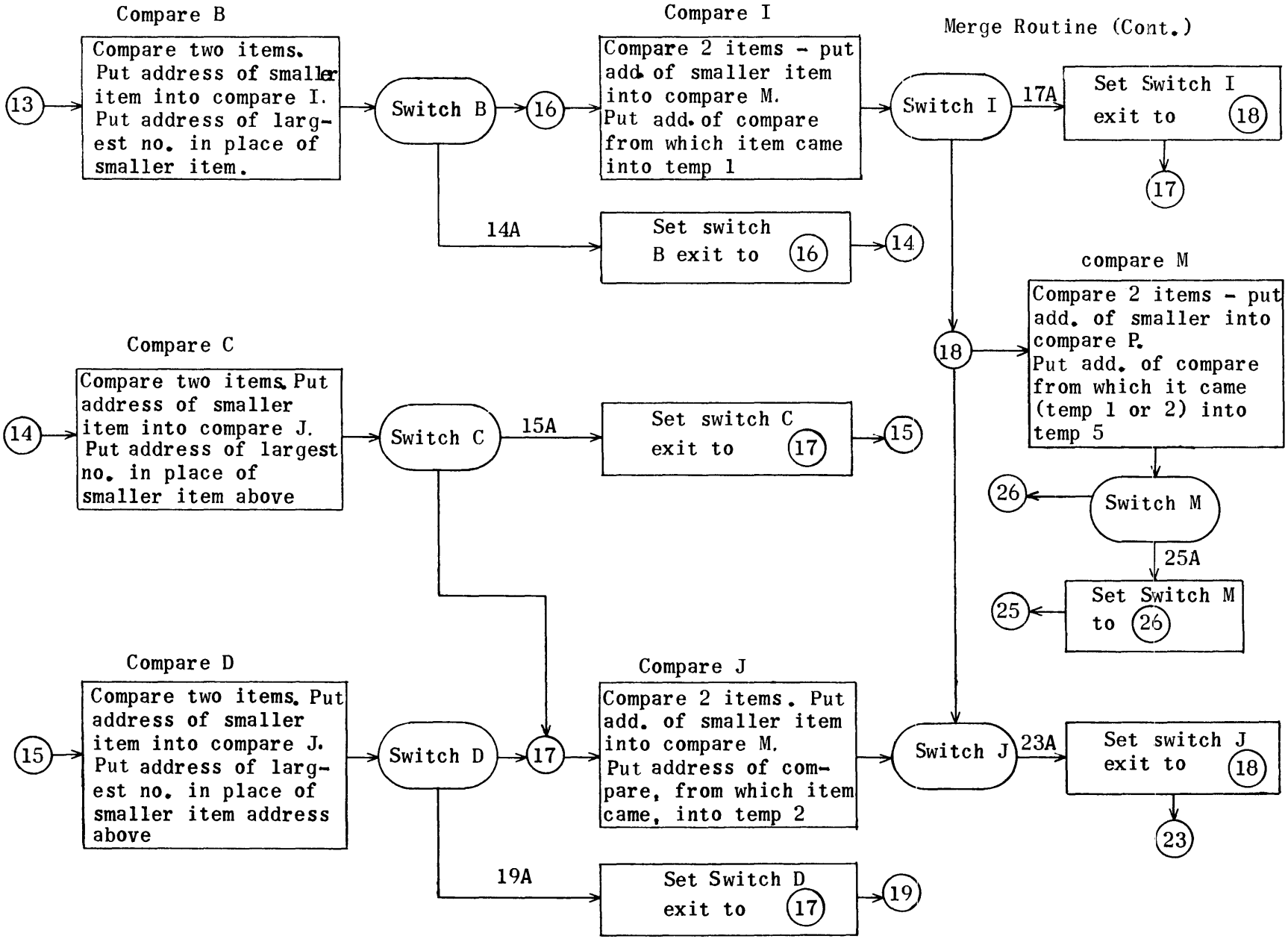


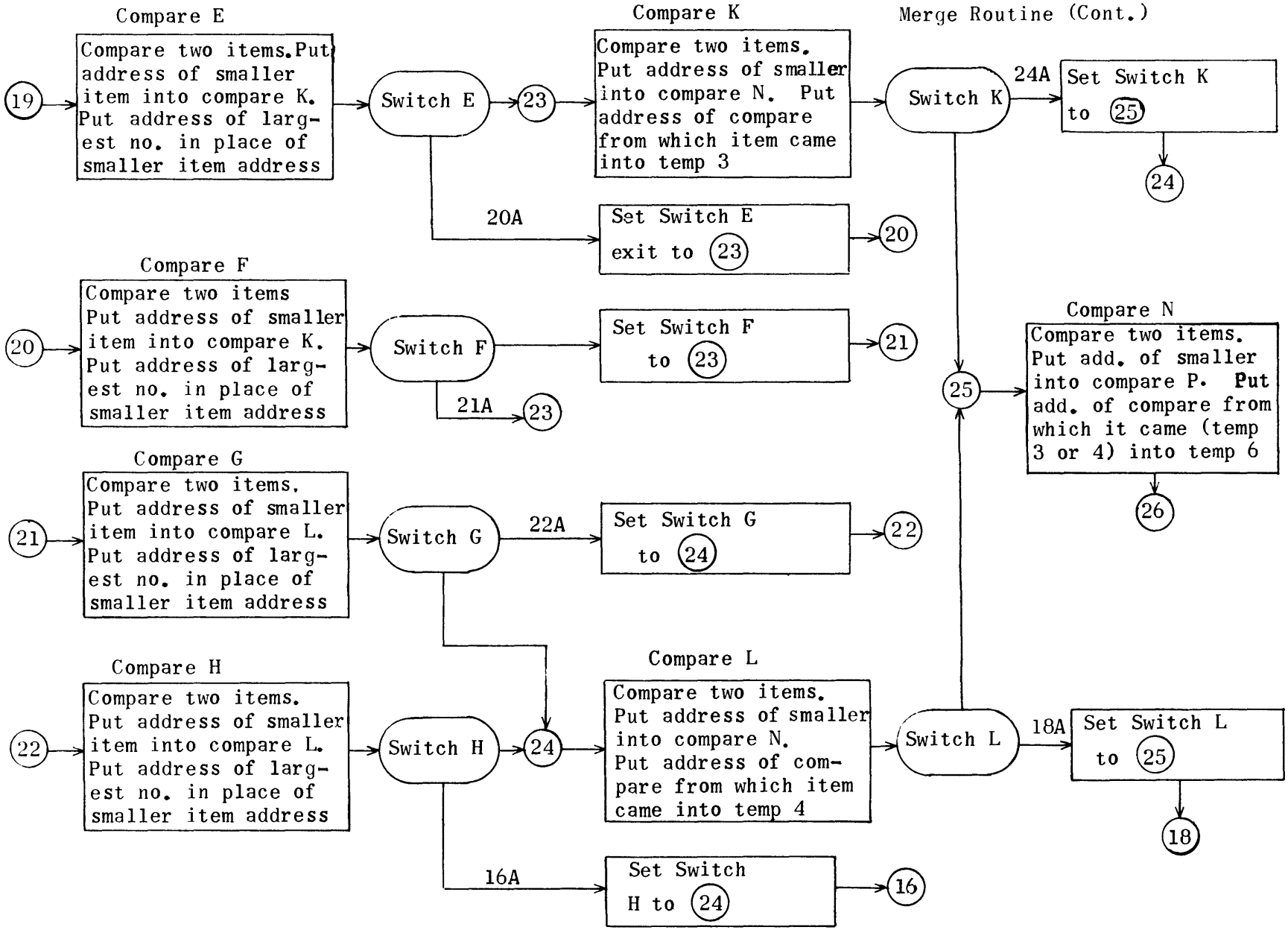
Merge Routine (Cont.)



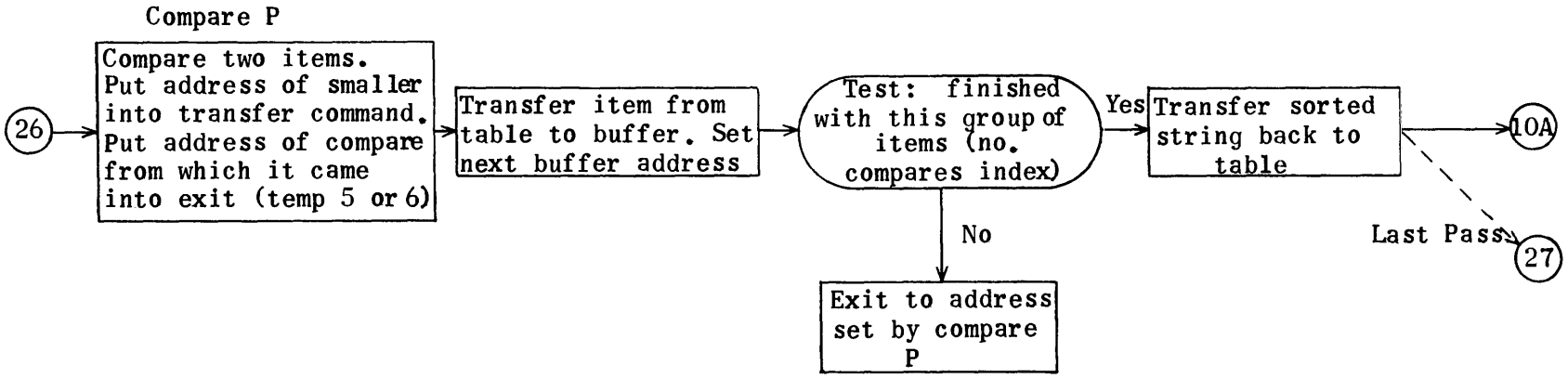
Merge Routine (Cont.)



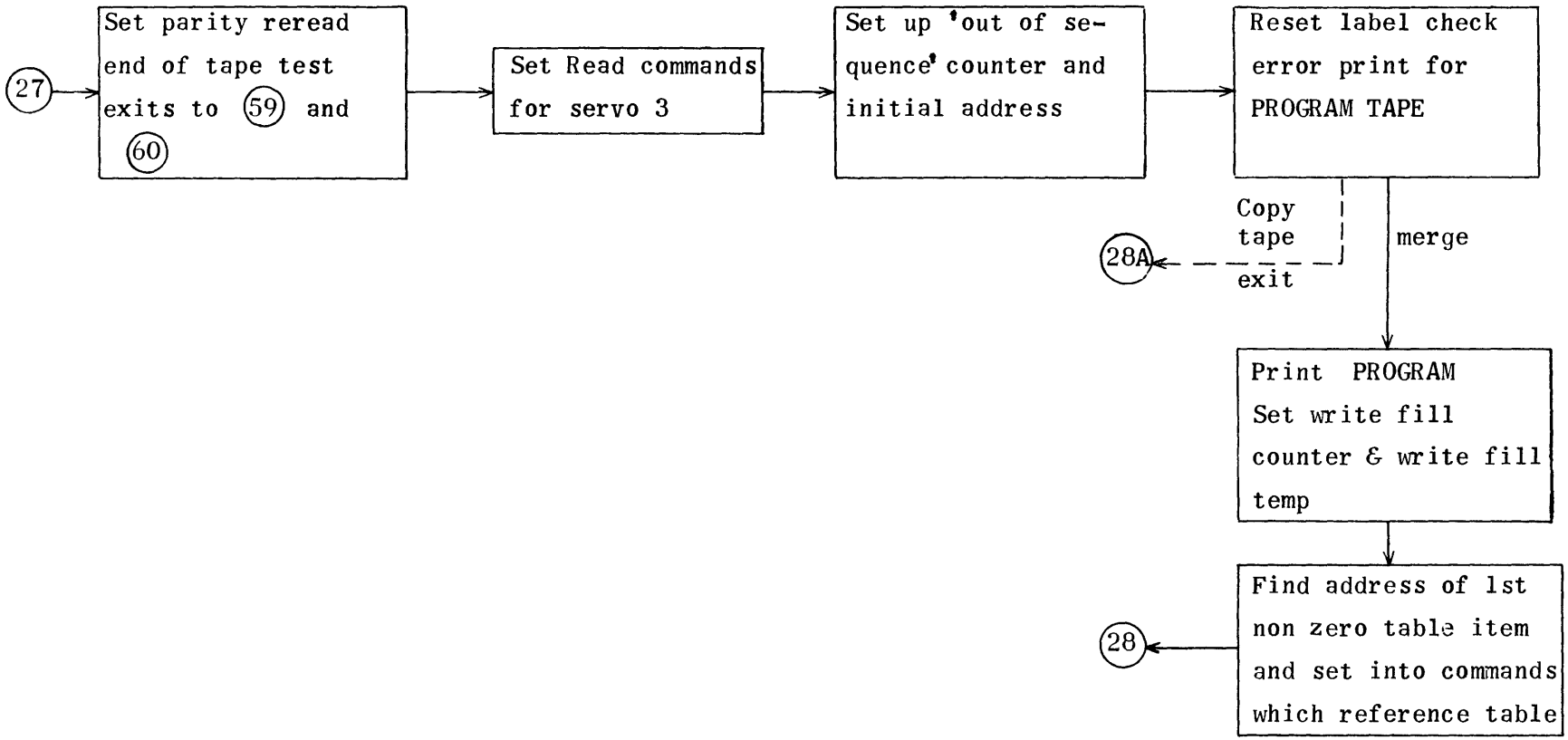

Merge Routine (Cont.)
Compare B
Compare two items. Put address of smaller item into compare I. Put address of largest no. in place of smaller item.
Switch B
16
Compare I
Compare 2 items - put add. of smaller item into compare M. Put add. of compare from which item came into temp 1
Switch I
17A
Set Switch I exit to 18
17
14A
Set switch B exit to 16
14
compare M
Compare 2 items - put add. of smaller into compare P. Put add. of compare from which it came (temp 1 or 2) into temp 5
18
Compare C
Compare two items. Put address of smaller item into compare J. Put address of largest no. in place of smaller item above
Switch C
15A
Set switch C exit to 17
15
26
Switch M
25A
25
Set Switch M to 26
Compare D
Compare two items. Put address of smaller item into compare J. Put address of largest no. in place of smaller item address above
Switch D
17
Compare J
Compare 2 items. Put add. of smaller item into compare M. Put address of compare, from which item came, into temp 2
Switch J
23A
Set switch J exit to 18
23
19A
Set Switch D exit to 17
19


Compare E
Compare two items. Put address of smaller item into compare K. Put address of largest no. in place of smaller item address
19
Switch E
23
Compare K
Compare two items. Put address of smaller into compare N. Put address of compare from which item came into temp 3
Merge Routine (Cont.)
Switch K
24A
Set Switch K to 25
24
20A
Set Switch E exit to 23
20
Compare F
Compare two items Put address of smaller item into compare K. Put address of largest no. in place of smaller item address
20
Switch F
Set Switch F to 23
21
21A
23
Compare N
Compare two items. Put add. of smaller into compare P. Put add. of compare from which it came (temp 3 or 4) into temp 6
25
26
Compare G
Compare two items. Put address of smaller item into compare L. Put address of largest no. in place of smaller item address
21
Switch G
22A
Set Switch G to 24
22
Compare H
Compare two items. Put address of smaller item into compare L. Put address of largest no. in place of smaller item address
22
Switch H
24
Compare L
Compare two items. Put address of smaller into compare N. Put address of compare from which item came into temp 4
Switch L
18A
Set Switch L to 25
18
16A
Set Switch H to 24
16

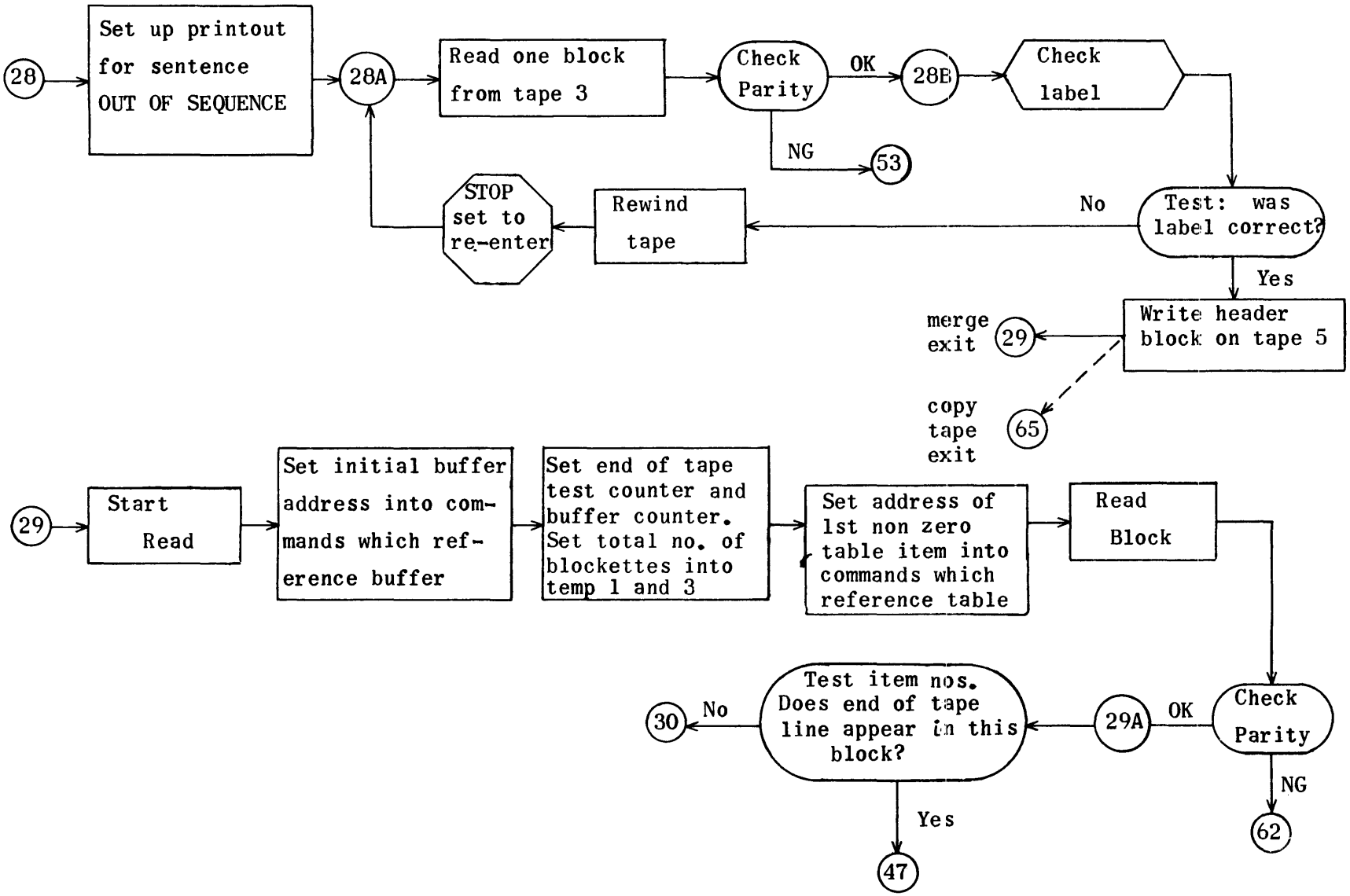
Merge Routine (Cont.)

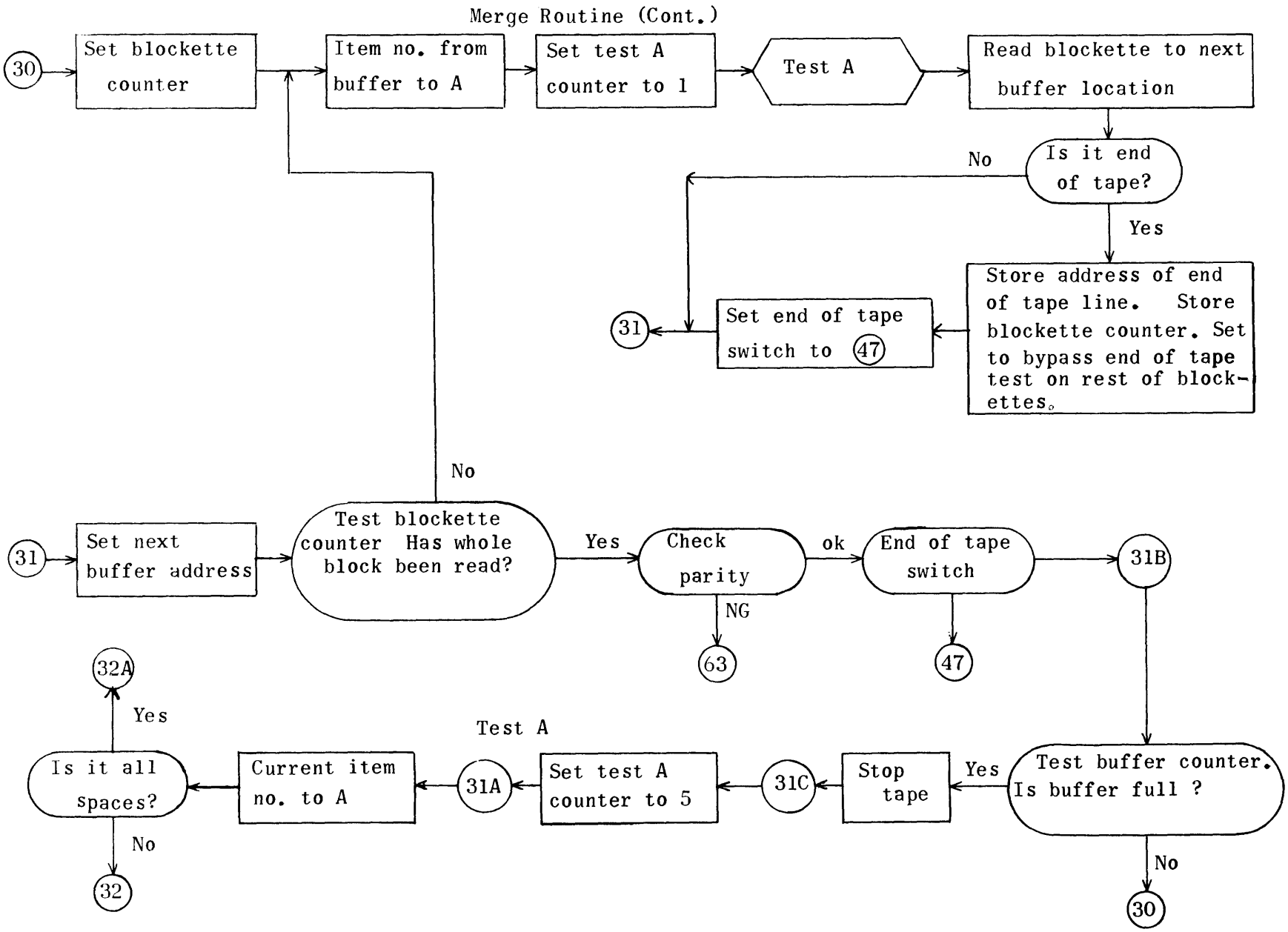


Merge Routine (Cont.)

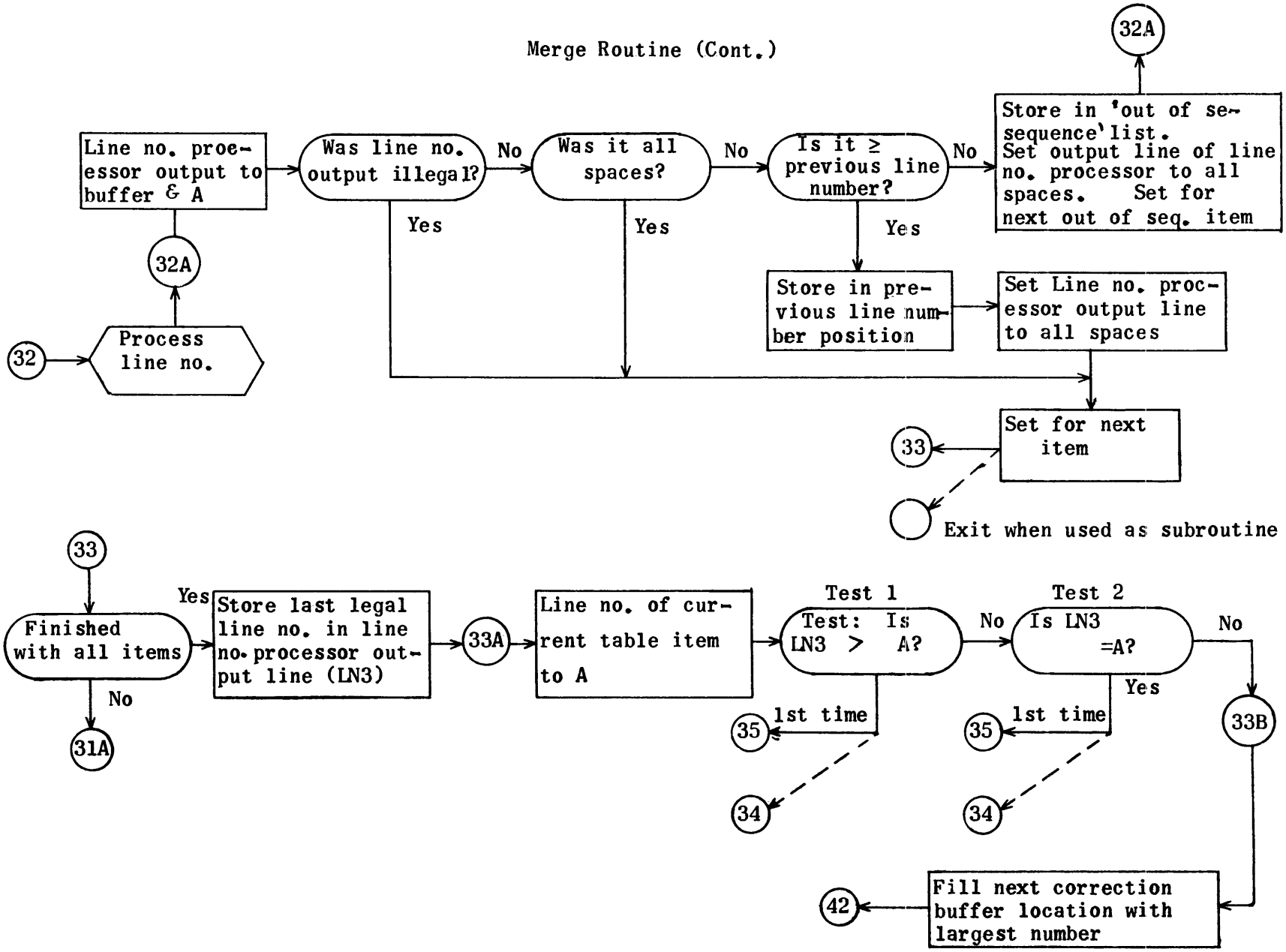


Merge Routine (Cont.)

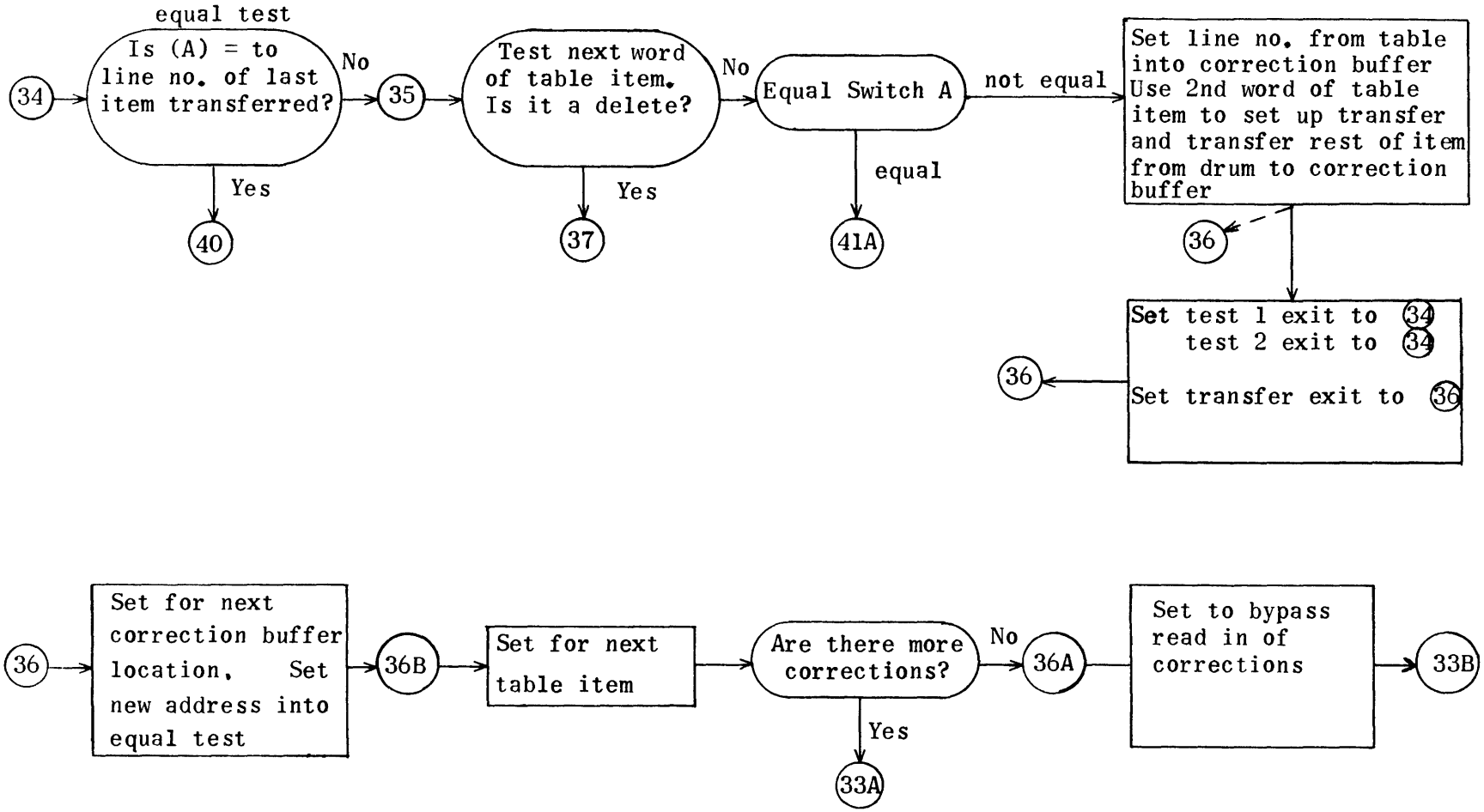


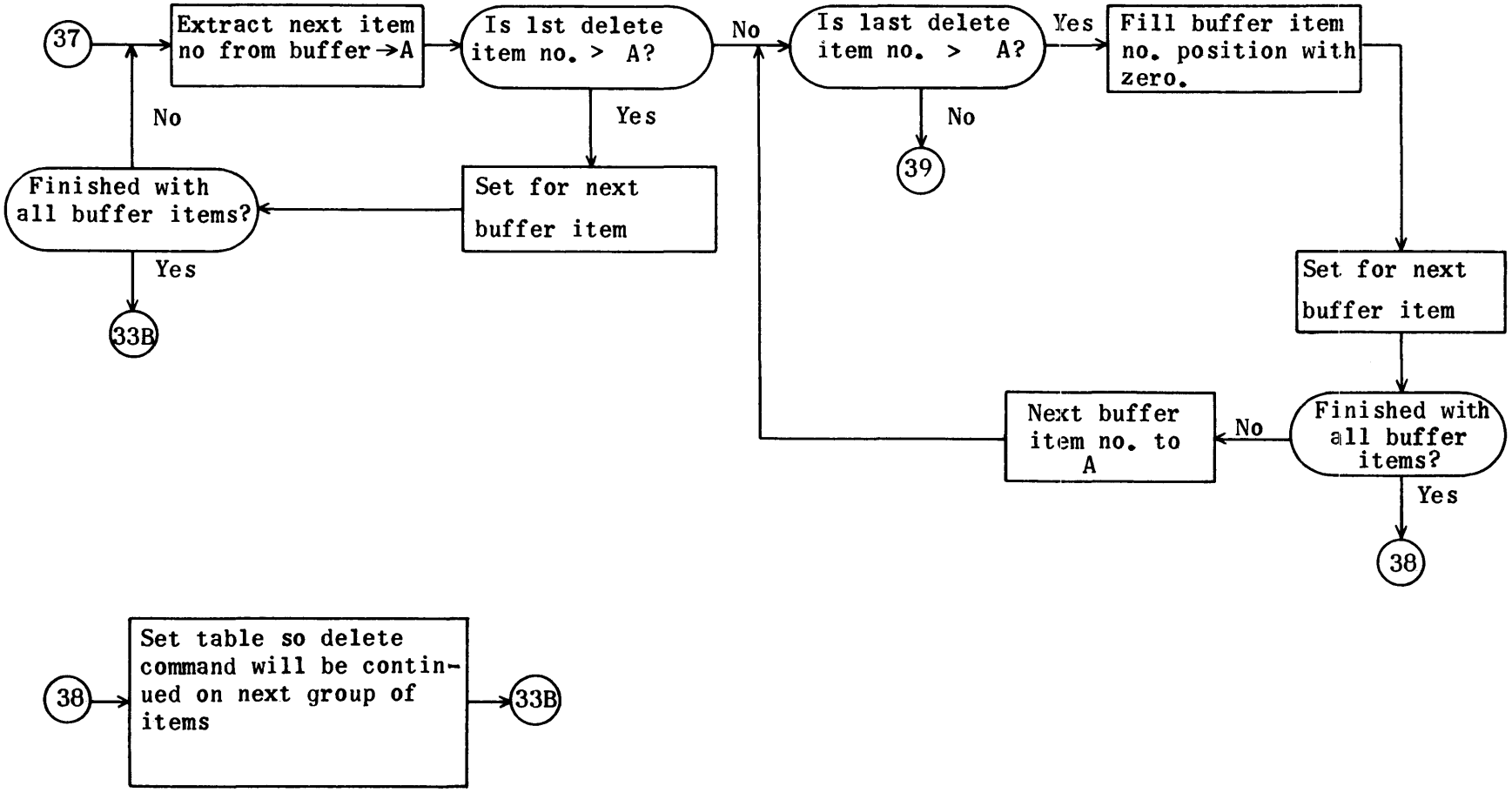

Merge Routine (Cont.)
30
Set blockette counter
Item no. from buffer to A
Set test A counter to 1
Test A
Read blockette to next buffer location
Is it end of tape?
No
Yes
Store address of end of tape line. Store blockette counter. Set to bypass end of tape test on rest of blockettes.
Set end of tape switch to 47
31
No
31
Set next buffer address
Test blockette counter Has whole block been read?
Yes
Check parity
ok
End of tape switch
31B
NG
63
47
32A
Yes
Test A
Is it all spaces?
Current item no. to A
31A
Set test A counter to 5
31C
Stop tape
Yes
Test buffer counter. Is buffer full ?
No
32
30

Merge Routine (Cont.)

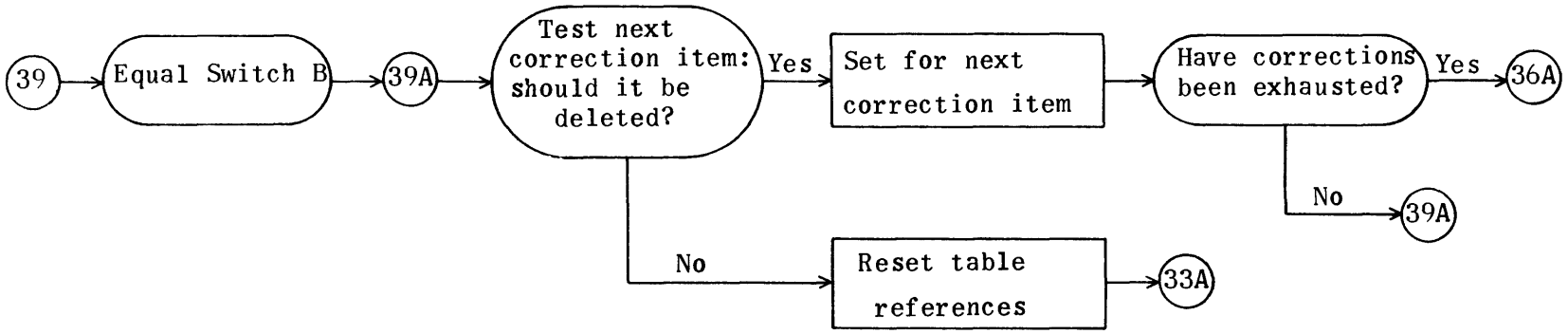


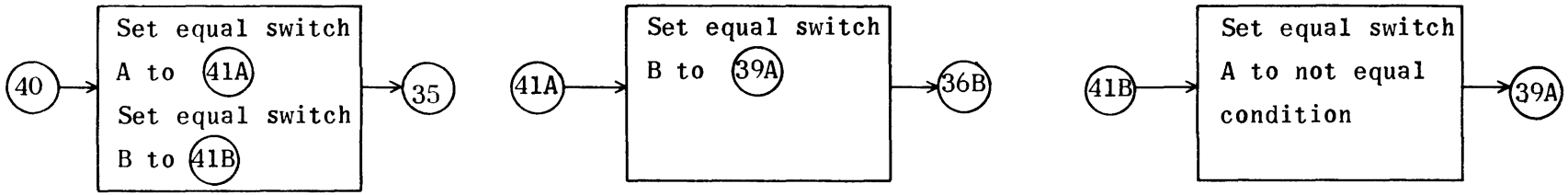
Merge Routine (Cont.)



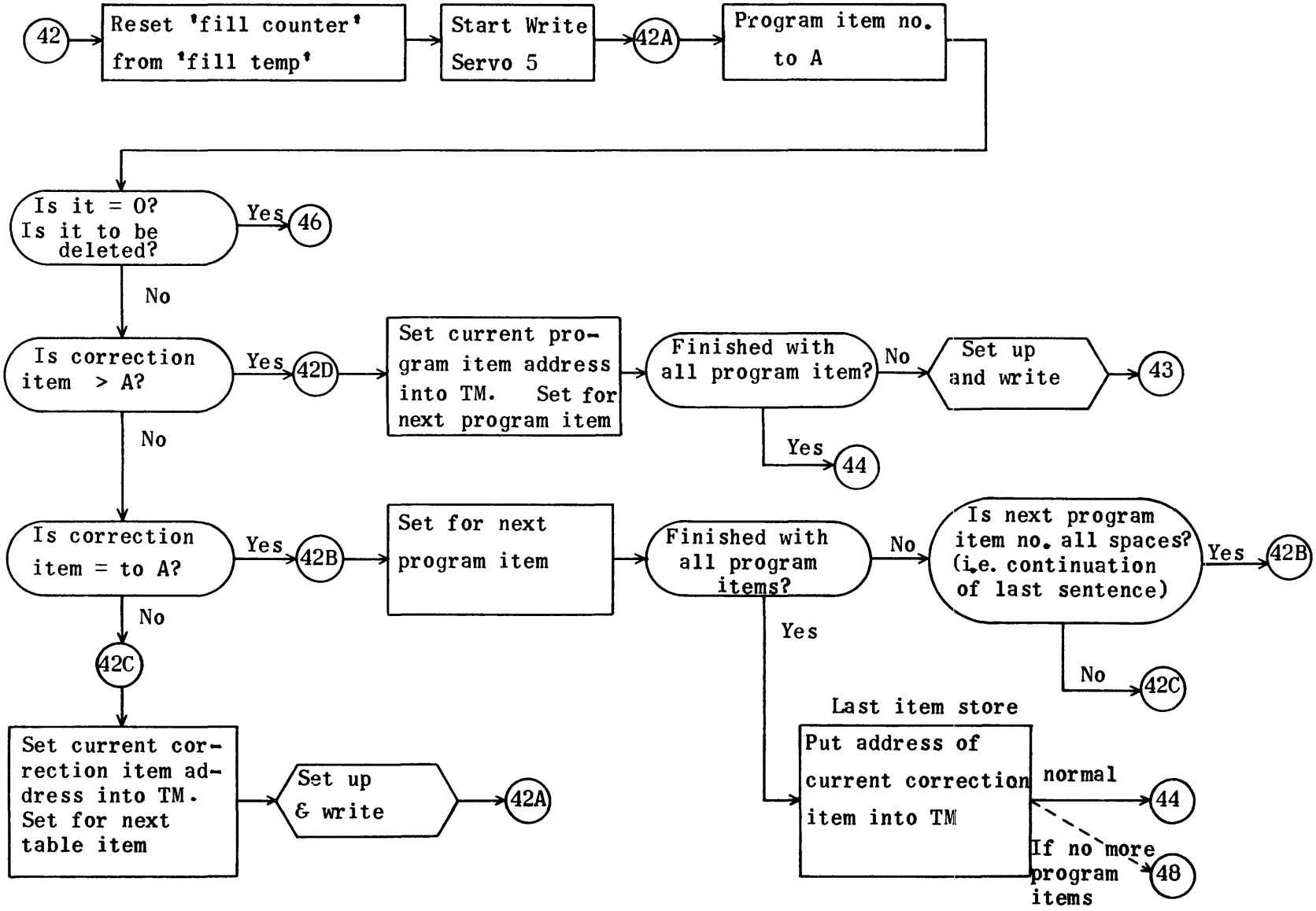

37
Extract next item no from buffer→A
Is 1st delete item no. > A?
No
Is last delete item no. > A?
Yes
Fill buffer item no. position with zero.
No
39
Yes
Set for next buffer item
Finished with all buffer items?
No
Yes
33B
Set for next buffer item
Next buffer item no. to A
No
Finished with all buffer items?
Yes
38
38
Set table so delete command will be continued on next group of items
33B

Merge Routine (Cont.)

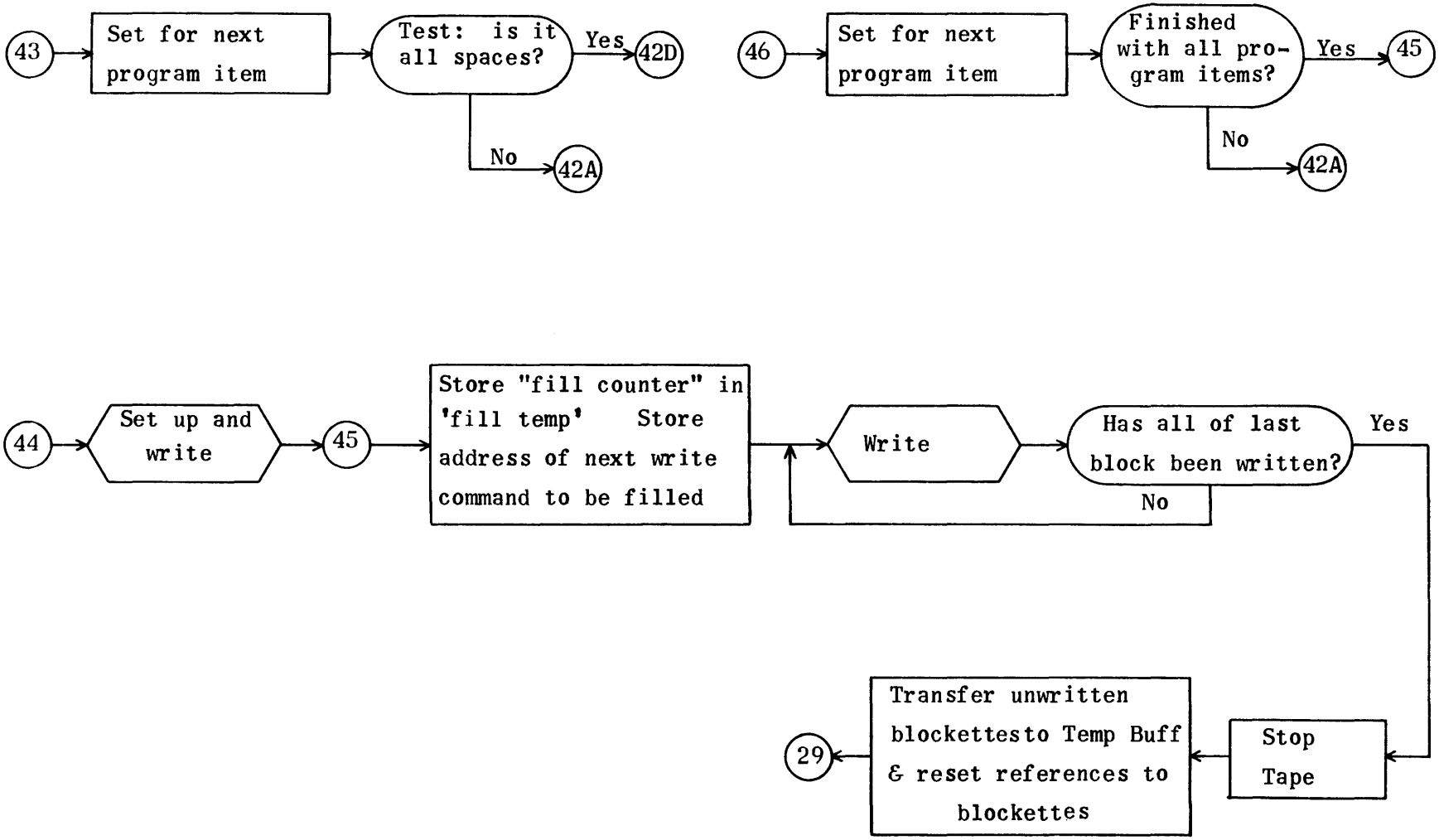




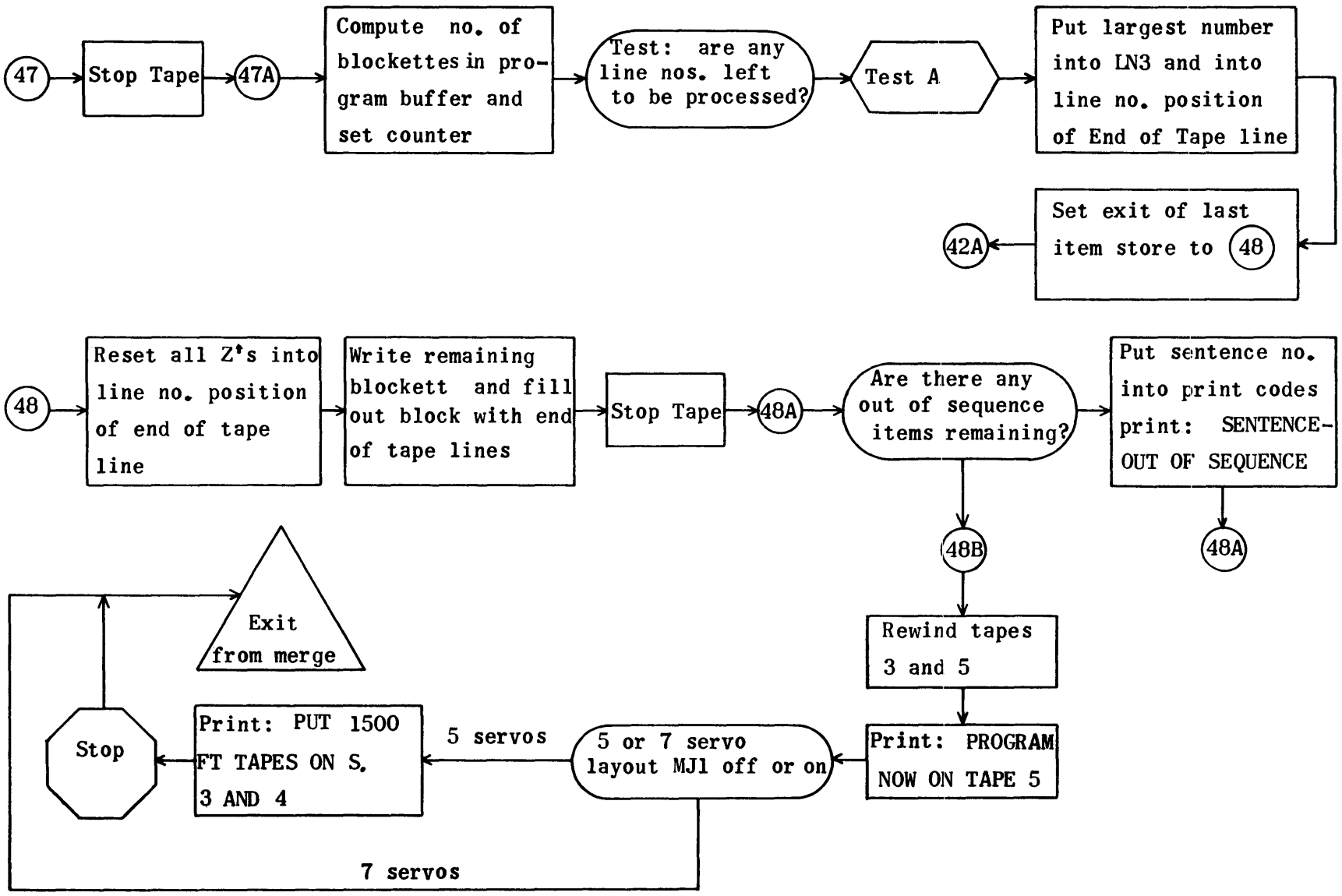
Merge Routine (Cont.)



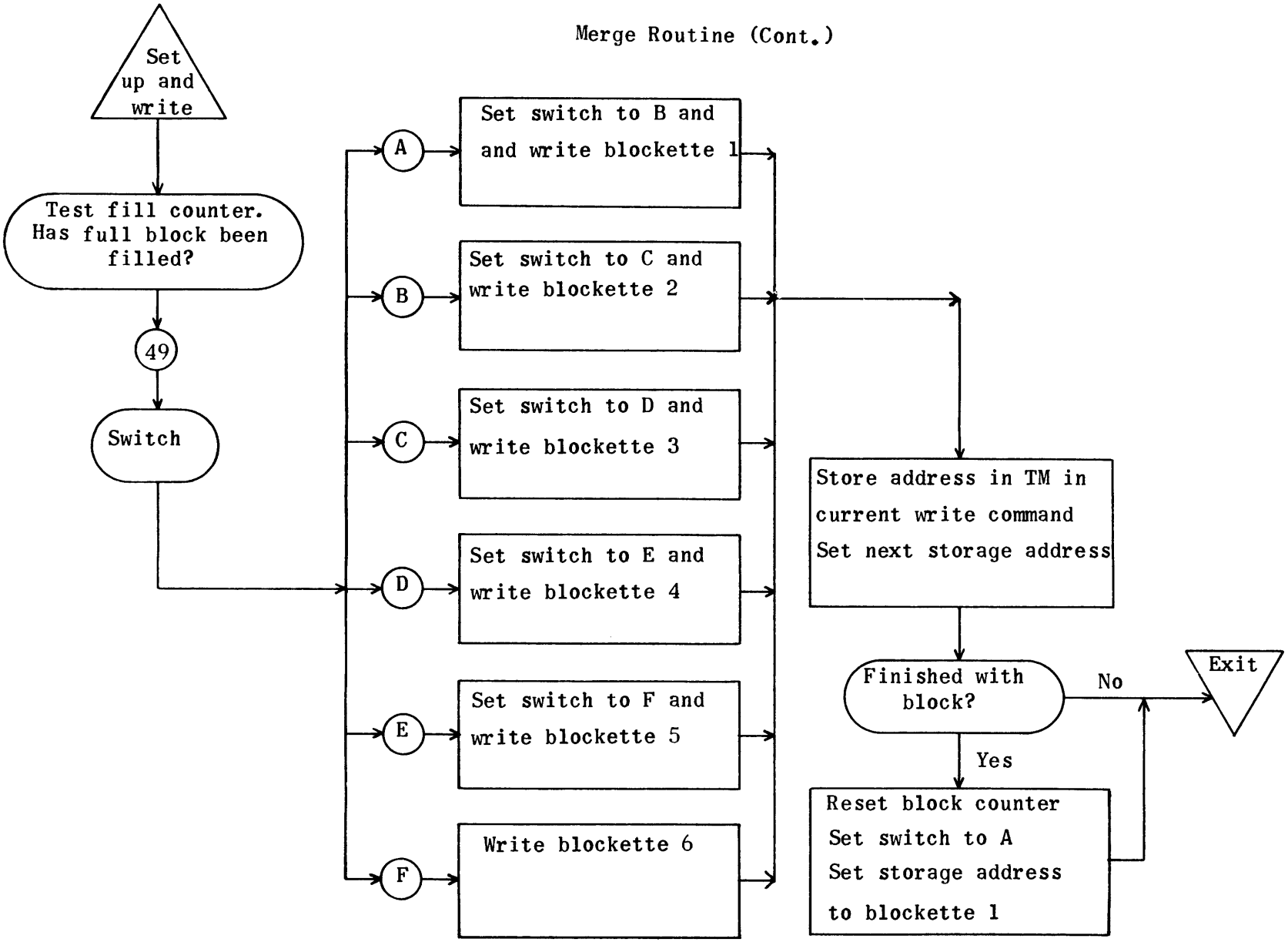
Merge Routine (Cont.)

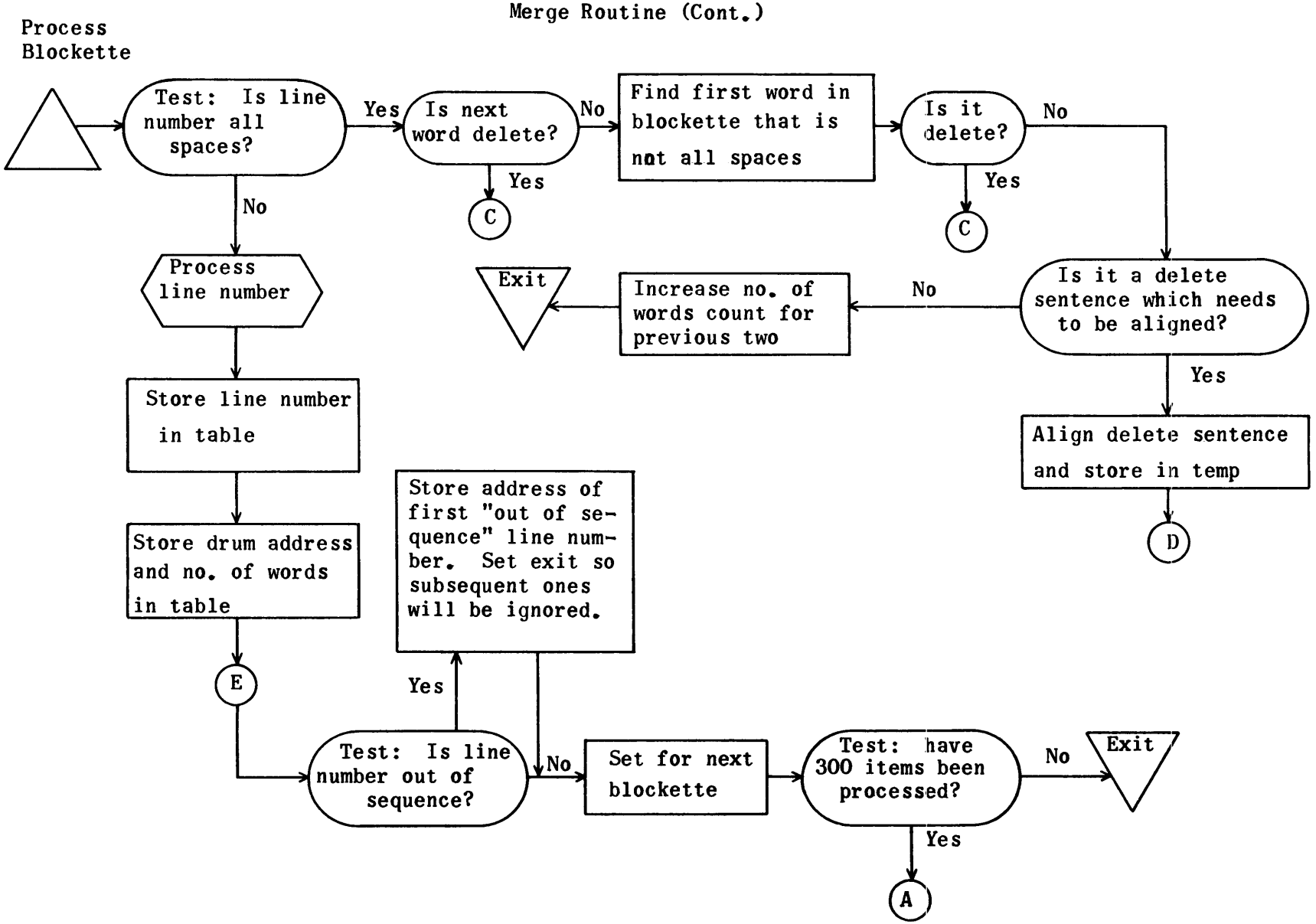


Merge Routine (Cont.)

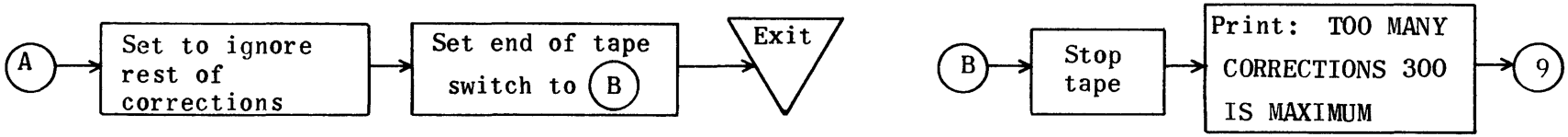


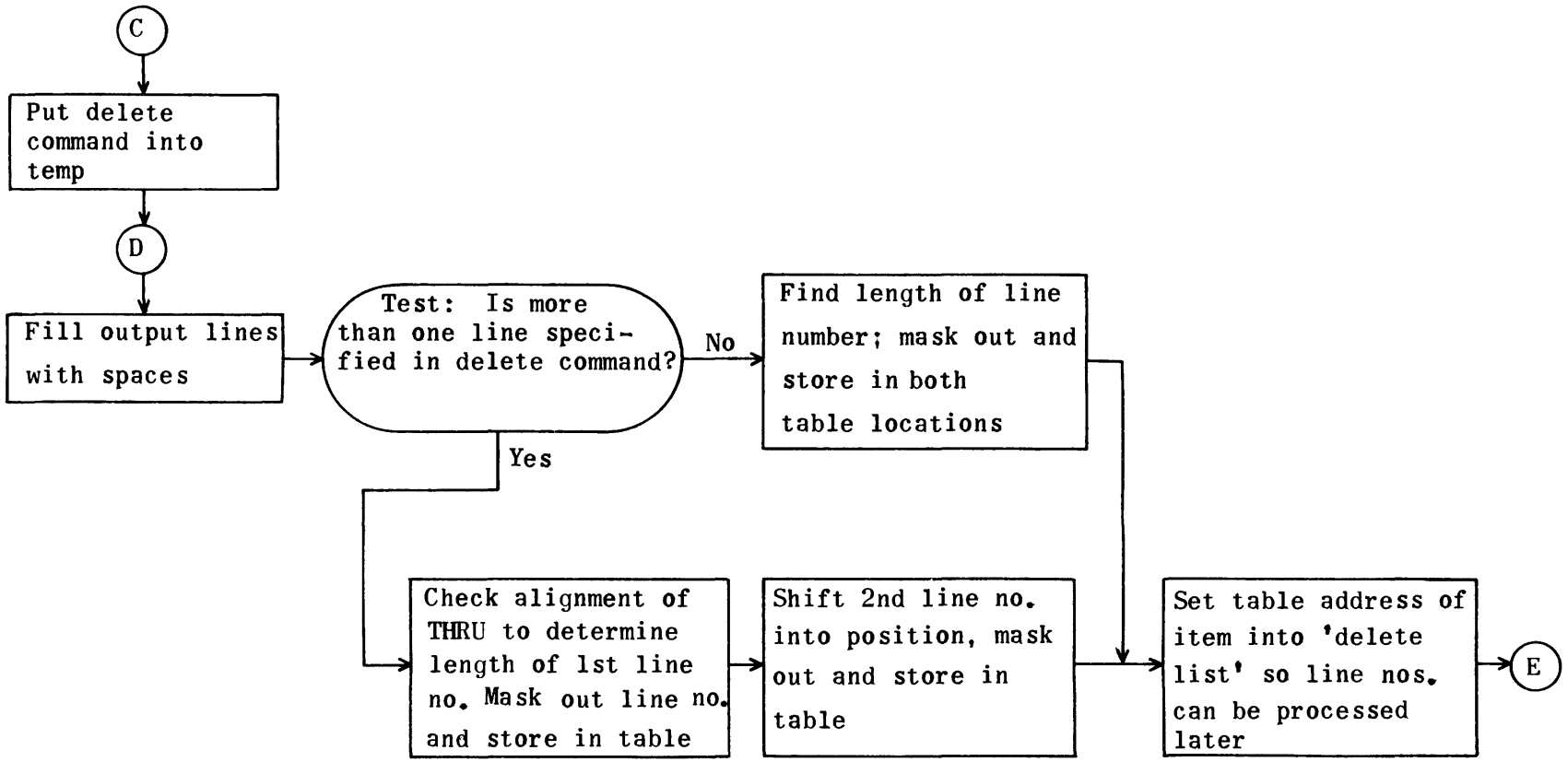
Merge Routine (Cont.)



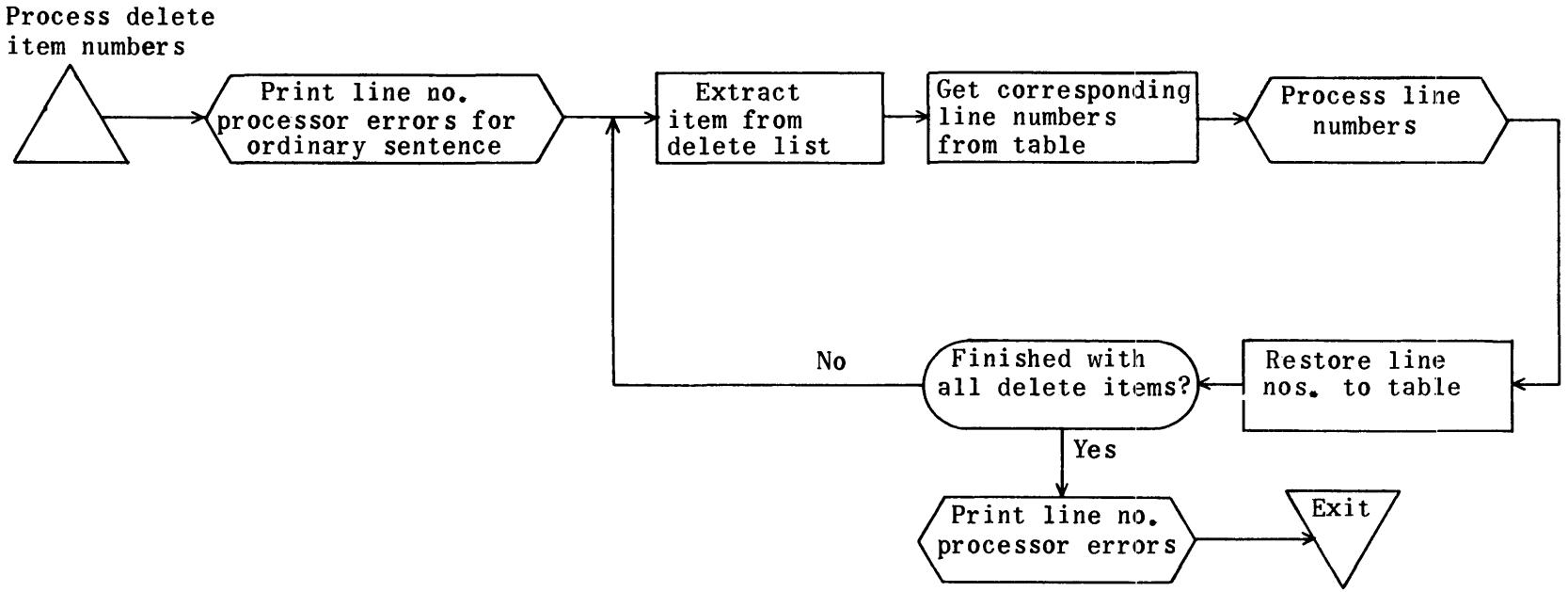

Process Blockette
Test: Is line number all spaces?
Yes
Is next word delete?
No
Find first word in blockette that is not all spaces
Is it delete?
No
Yes
C
Yes
C
No
Process line number
Exit
Increase no. of words count for previous two
No
Is it a delete sentence which needs to be aligned?
Yes
Align delete sentence and store in temp
D
Store line number in table
Store drum address and no. of words in table
E
Store address of first "out of sequence" line number. Set exit so subsequent ones will be ignored.
Yes
Test: Is line number out of sequence?
No
Set for next blockette
Test: have 300 items been processed?
No
Exit
Yes
A

Merge Routine (Cont.)

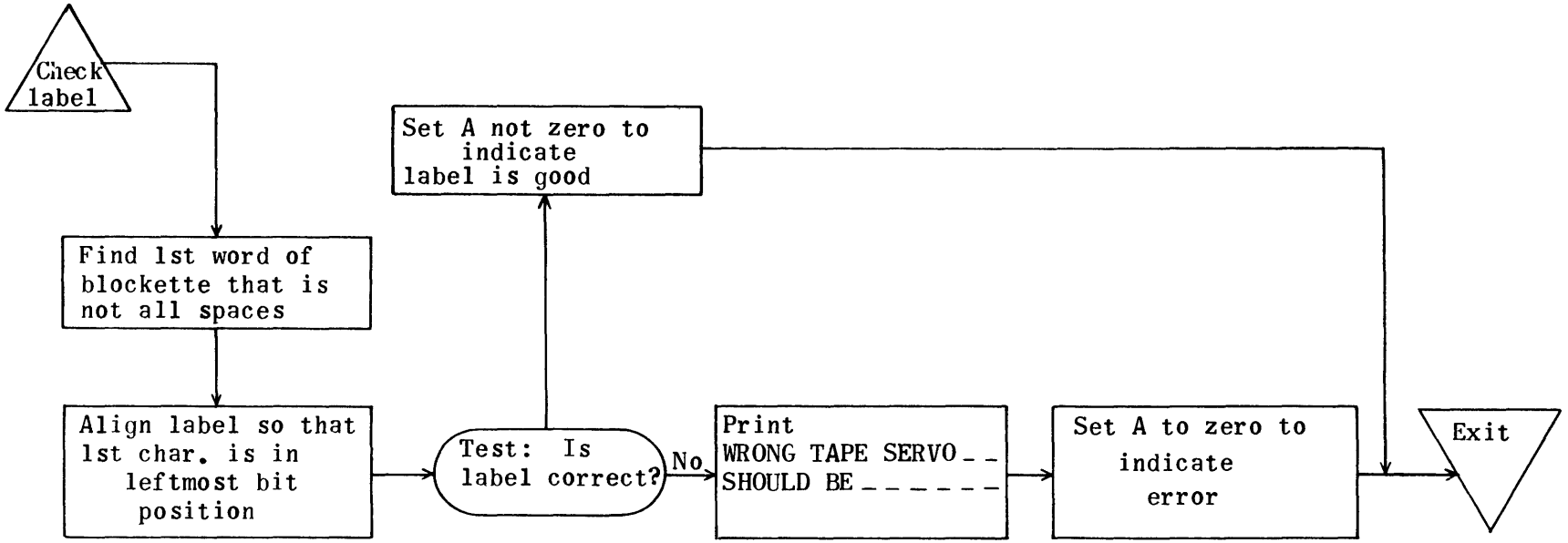




Merge Routine (Cont.)

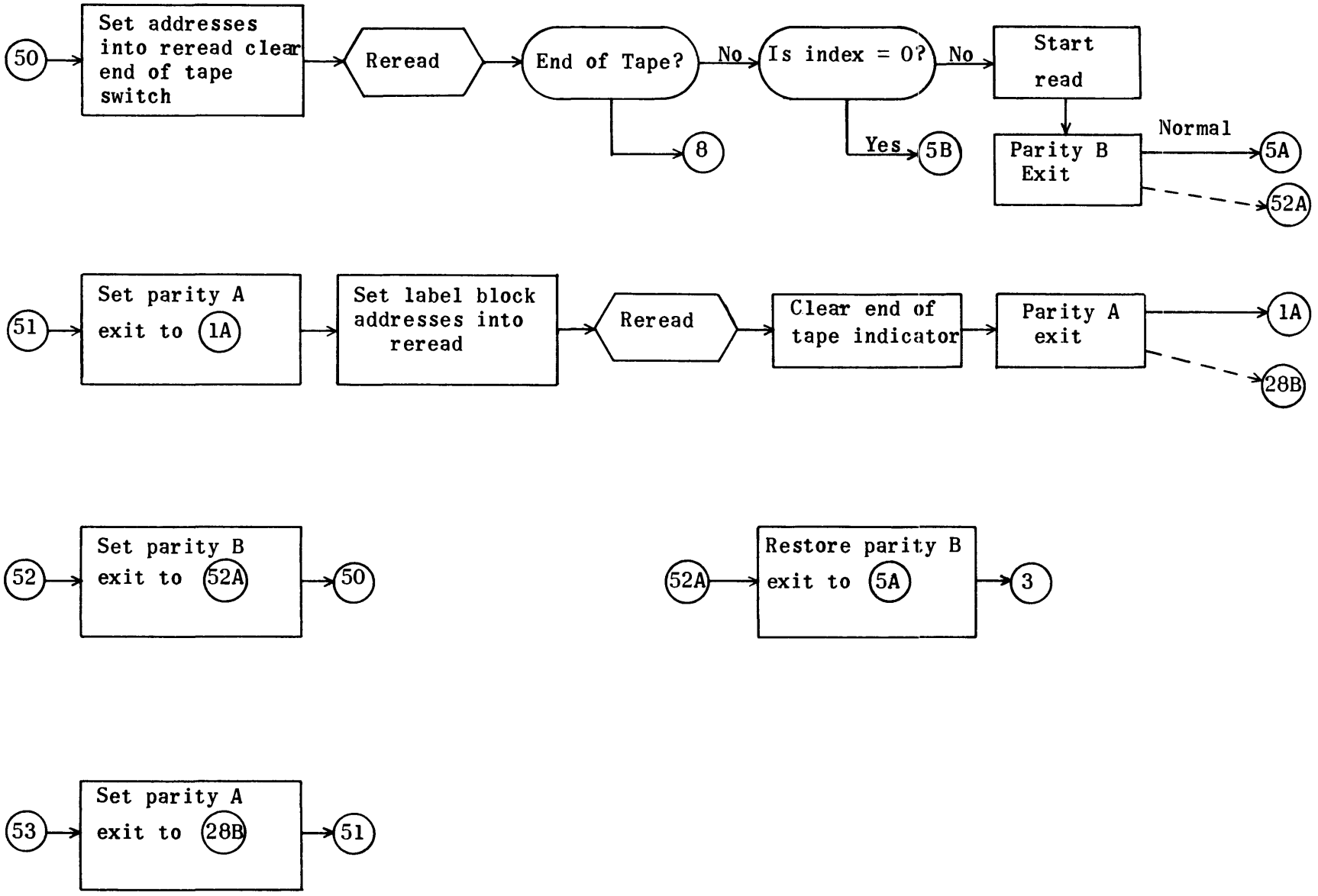


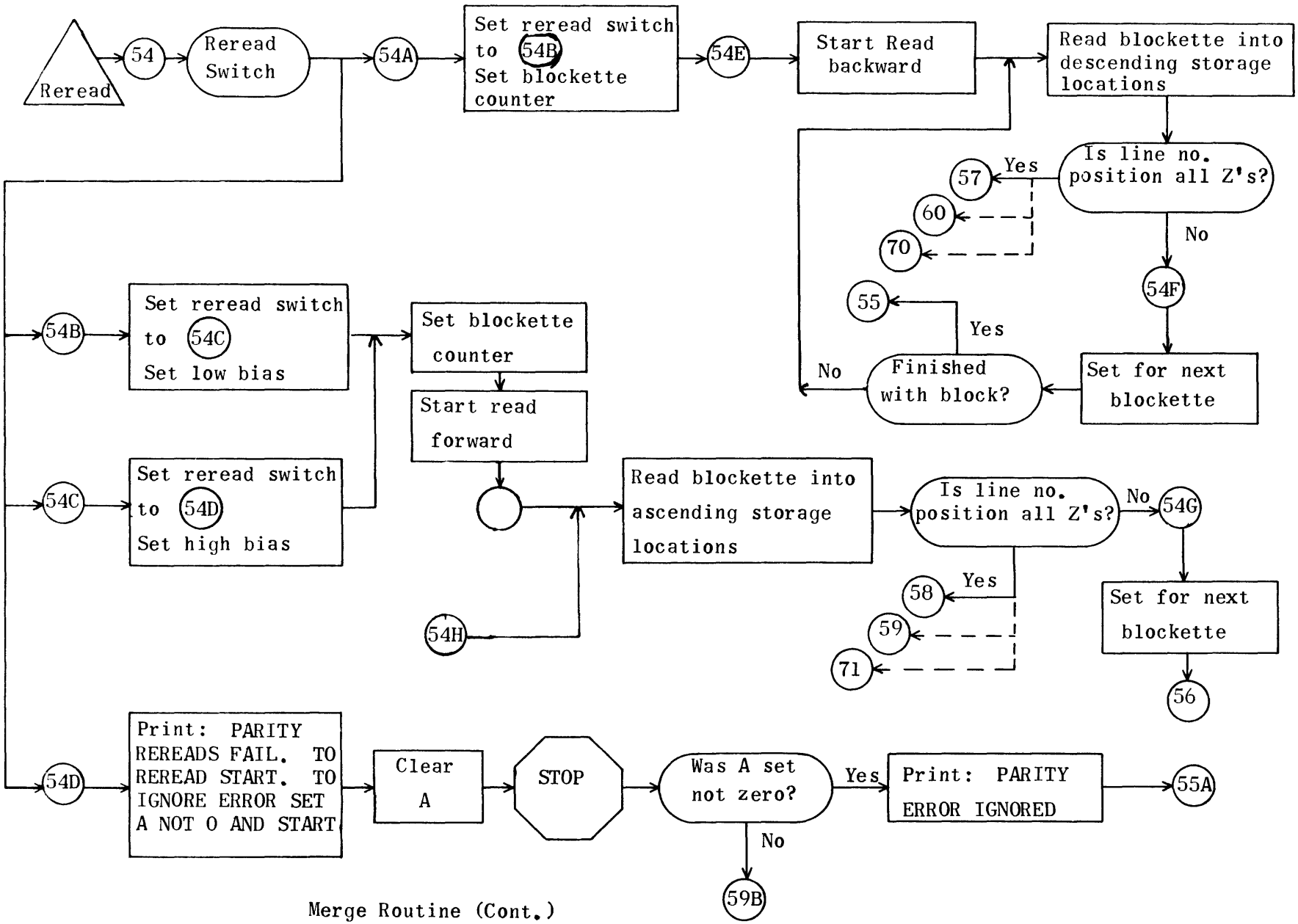
Merge Routine (Cont.)



Parity Rereads

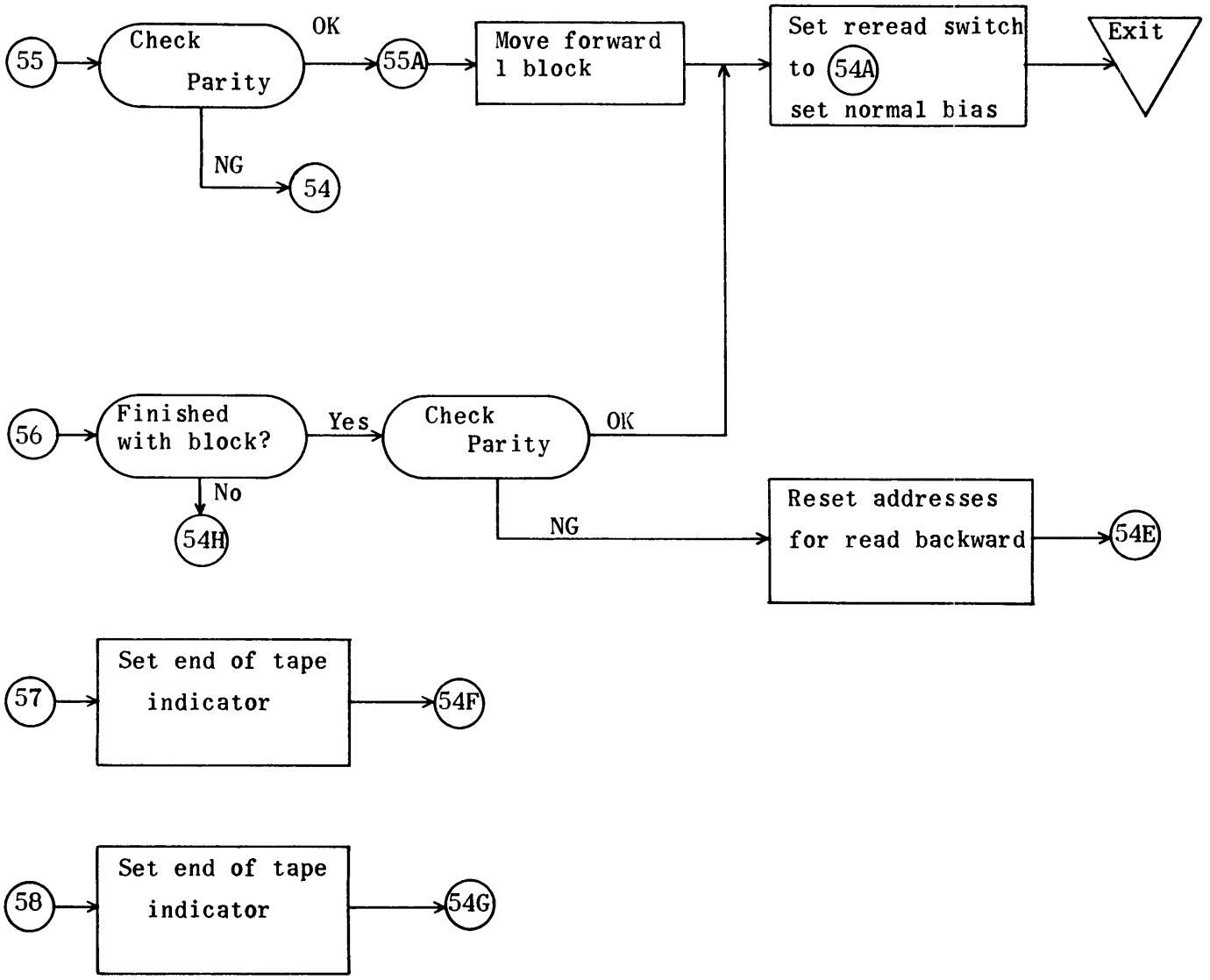
Merge Routine (Cont.)



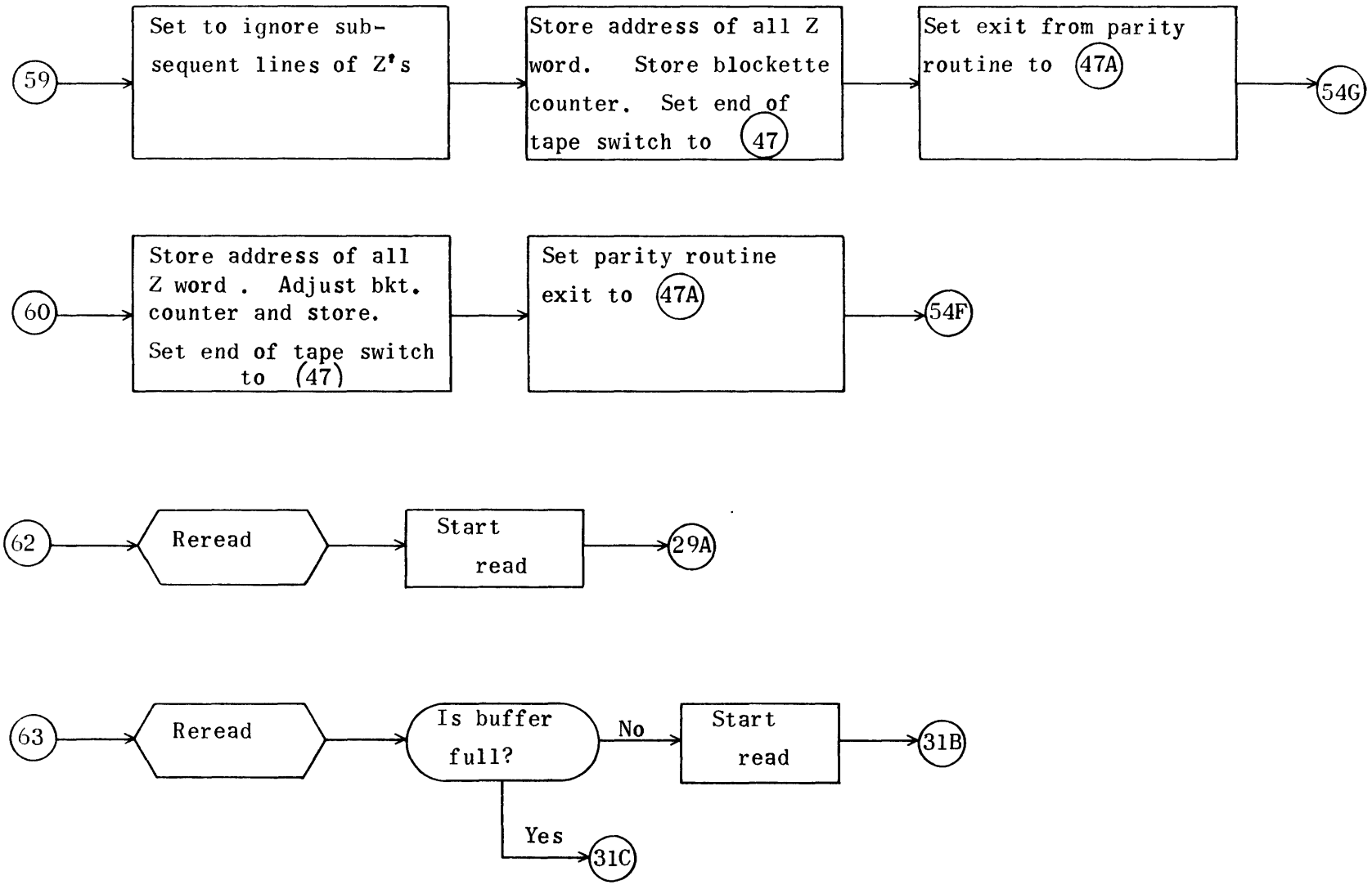


Merge Routine (Cont.)

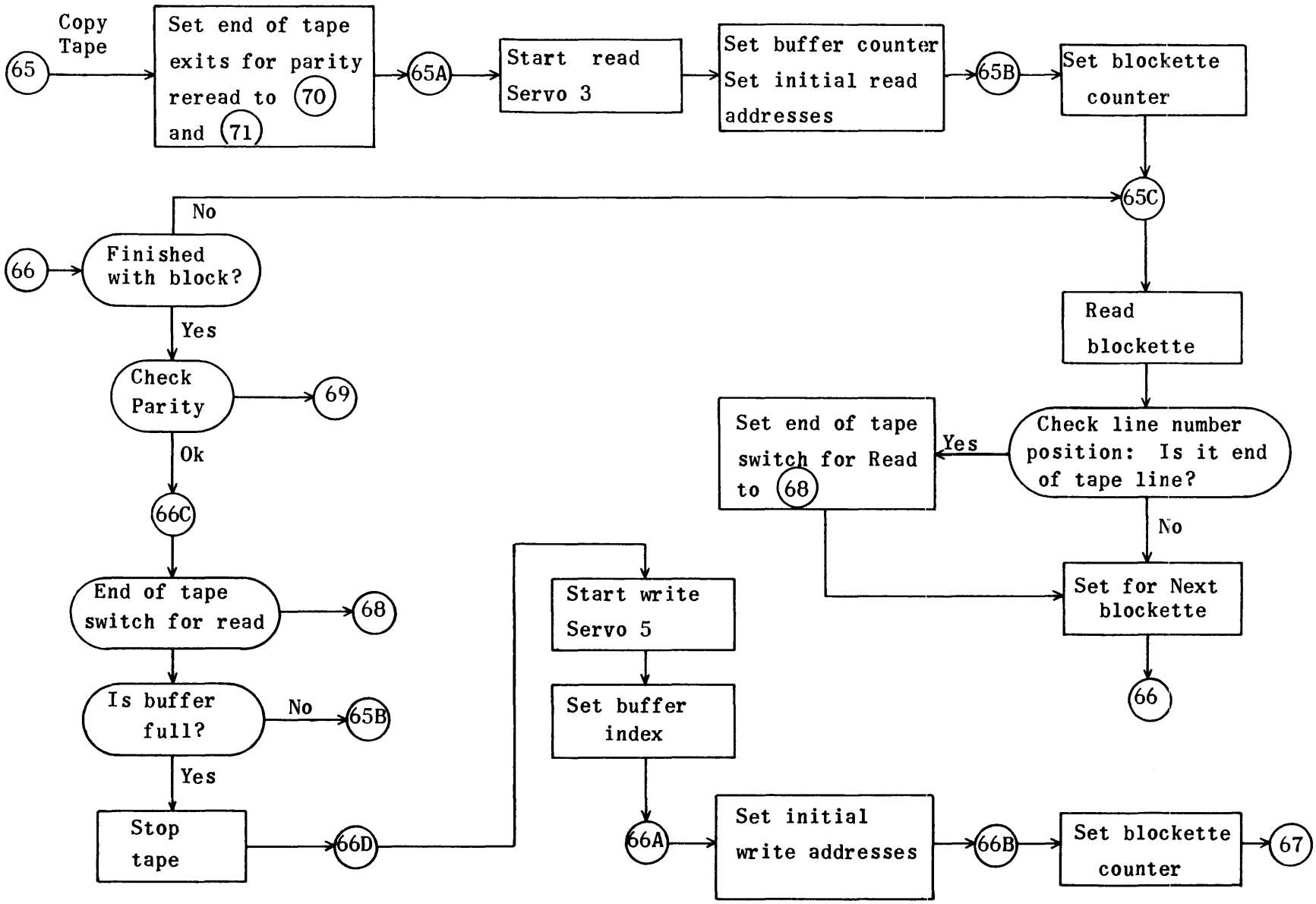
Merge Routine (Cont.)

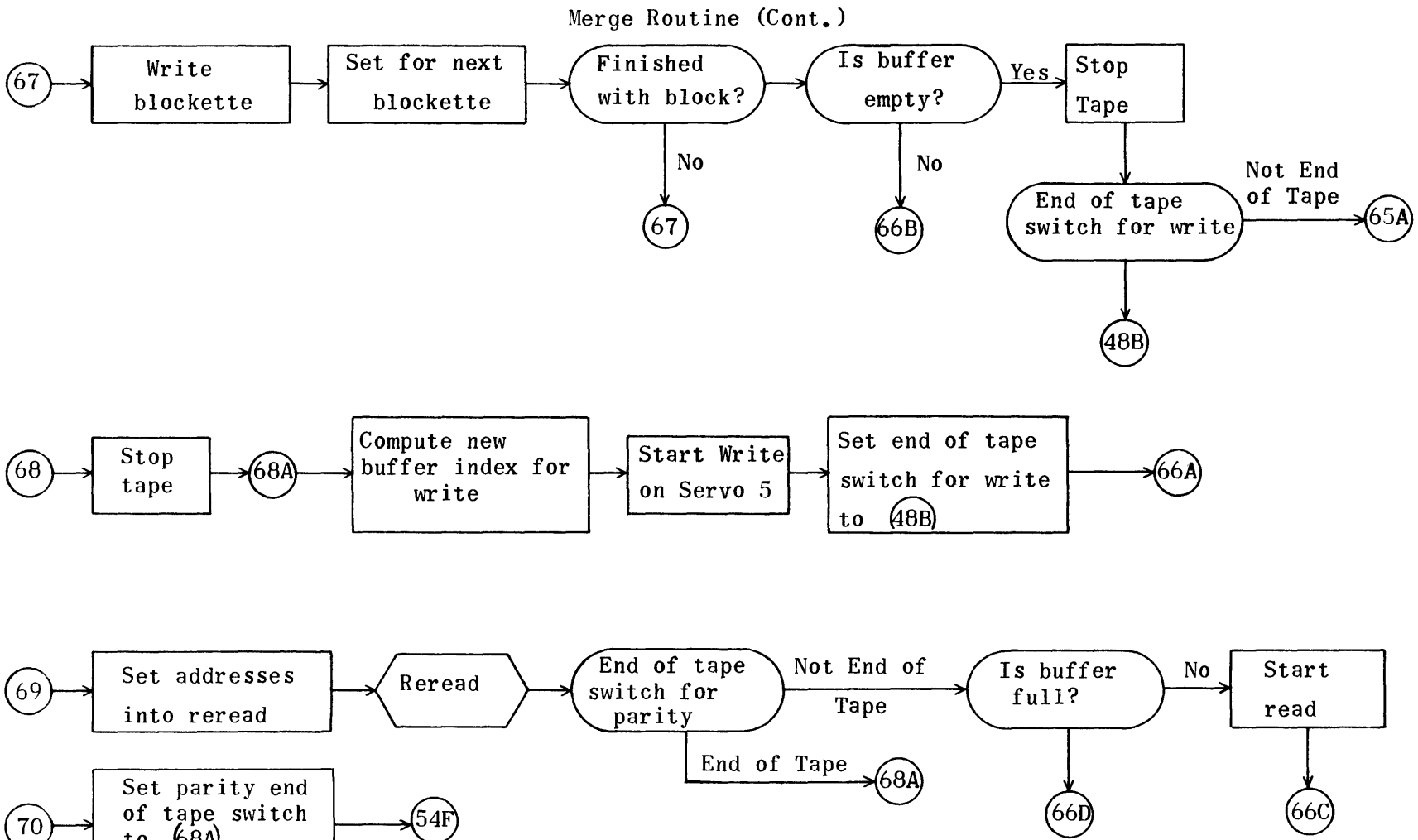


Merge Routine (Cont.)



Merge Routine (Cont.)




Merge Routine (Cont.)
67
Write blockette
Set for next blockette
Finished with block?
No
67
Is buffer empty?
No
66B
Yes
Stop Tape
End of tape switch for write
Not End of Tape
65A
48B
68
Stop tape
68A
Compute new buffer index for write
Start Write on Servo 5
Set end of tape switch for write to 48B
66A
69
Set addresses into reread
Reread
End of tape switch for parity
Not End of Tape
End of Tape
68A
Is buffer full?
No
66D
Start read
66C
70
Set parity end of tape switch to 68A
54F
71
Set parity end of tape switch to 68A
54G

TAPE MERGE REGIONS

```
RE GQ761
RE PZ3071
RE ZA77000
RE UC2507
RE RQ35
RE RR52
RE RS110
RE LP140
RE CN147
RE CX232
RE XY273
RE JL375
RE XX433
RE VF525
RE MG543
RE MN604
RE NW711
RE MP770
RE KW1003
RE BJ1042
RE UN1077
RE NR1164
RE ET1213
RE SV1310
RE LC1374
RE SR1453
RE XZ1553
RE WT1655
RE NZ1736
RE ZR1766
RE WX2005
RE FT2045
RE PB2105
RE DL2205
RE RB2246
RE LN2340
RE DR2353
RE MR2375
RE RC2471
RE WS2534
RE PA2540
RE CD2575
RE UP2653
RE UQ2675
RE US2705
RE UW2745
RE FC2771
RE DT40002
RE BR7635
RE TM5
RE BU4225
RE CB6125
RE GZ3073
RE GR5735
RE TB3073
```

TAPE MERGE ROUTINE

```
     IA    RQ
 0   RJ    PZ2       PZ          MJ 0 RQ2 on 1103A tape
 1   EF    0         VF14        Set bypass
 2   MJ    30000     SV          If MJ3 is set go to SV from MERGE.  If
                                 MJ3 not set take next instruction and copy
                                 tape
 3   RJ    ET42      SR60        Set up tape commands and parity reread
 4   RJ    MG14      MG          Read label, check and write
 5   RJ    XY11      RQ10        Set end of tape exit for parity reread
 6   TV    RQ13      RS14
 7   MJ    0         XY12

10   RJ    XY35      RR          Set end of tape exit for parity reread
11   TV    RQ13      RS14        Set new exit for parity reread
12   MJ    0         XY36        Go back to finish reread
13   0     0         RS2
14   0     0         24
     CA    RQ15


     IA    PZ
 0   MJ    20000     RQ1         Test: Is TCU2 to be used?  If so go to RQ1
 1   TP    PZ5       VF14   }
 2   RP    20014     30000  }    If not, reset tape commands for TCU1
 3   CC    VF        PZ4    }
 4   03    0         0
 5   00    10000     4000
     CA    PZ6


     IA    RR
 0   EF    0         VF          Start read
 1   TP    RQ14      TM2         Set buffer index
 2   TU    RR35      RR7    }
 3   TV    RR35      RR6    }    Set initial read addresses
 4   TP    CN4       TM4         Set blockette index
 5   RP    10024     RR7    }
 6   ER    10000     30000  }    Read blockette
 7   TP    30000     A      }
                              }  Test:  Is it end of tape blockette?  If so
10   EJ    CX6       RS     }  go to RS
11   RA    RR6       CN5    }
12   RA    RR7       CN12   }    Set for next blockette
13   IJ    TM4       RR5         Test:  Finished with block?  If not go
                                 back to RR5
14   ER    0         A      }
15   ZJ    RS11      RR16   }    Check parity
16   RJ    RR16      RR17
17   IJ    TM2       RR4         Is buffer full?  If not go back to RR4
```

```
20   EF   0       VF10        If so, stop tape
21   EF   0       VF12        Start write on Servo 5
22   TP   RQ14    TM2         Set buffer index
23   TV   RR35    RR26        Set initial write address
24   TP   CN4     TM4         Set blockette index
25   RP   10024   RR27   }
26   EW   10000   30000  }    Write blockette
27   RA   RR26    CN5         Set for next blockette

30   IJ   TM4     RR25        Finished with block?
31   IJ   TM2     RR24        Is buffer empty?
32   EF   0       VF10        If so stop tape
33   RJ   RR33    RR34
34   MJ   0       RR          Go back to read more items
35   0    GZ      GZ
     CA   RR36



     IA   RS
 0   RJ   RR16    RR11
 1   EF   0       VF10        Stop tape
 2   TP   RQ14    A      }
 3   ST   TM2     TM2    }    Set buffer index for write
 4   EF   0       VF12        Start write
 5   RJ   RR33    RR23        S
 6   EF   0       VF11        Rewind tape 5
 7   EF   0       VF2         Rewind tape 3
10   MJ   0       RS21        Go to RS21
11   TV   RR6     XY5    }
12   TU   RR7     XY10   }    Set addresses into parity reread
13   RJ   XY20    JL26        Reread
14   RJ   RS14    RS15        Switch set by parity routine if end of tape
                              occurred in block
15   TP   TM2     A      }
16   ZJ   RS17    RR21   }    Are more blocks to be read?  If not, go to
                              RR21 to write
17   EF   0       VF     }
20   MJ   0       RR16   }    If so, start read and go to RR16
21   TP   CX31    UP3    }
22   RJ   UP2     UP     }    Print:  PROGRAM NOW ON TAPE 5
23   MJ   10000   RS27        If MJ1 is set, go to exit. If not, take
                              next instruction
24   TP   XX71    UP3    }
25   RJ   UP2     UP     }    Print: PUT 1500 FT. TAPES ON SERVO 3 AND 4
26   MS   0       RS27        Stop, set to re-enter after tapes are
                              changed
27   MJ   0       ZA10        Exit
     CA   RS30
```

```
     IA   SV
 0   TP   CX40     UP3        }
 1   RJ   UP2      UP         }   Print:  MERGE
 2   TP   CX33     UP3        }
 3   RJ   UP2      UP         }   Print correction tape
 4   EF   0        VF14           Set bypass if 1105  MJ 0 SV5 on 1103A copy
 5   MJ   0        SV40           Out to rest of setup
 6   TV   SV47     TM ---->   |
 7   MJ   0        SV15       |
10   TV   SV50     TM ---->   |--> Set address of error entry
11   MJ   0        SV15       |
12   TV   SV51     TM ---->   |
13   MJ   0        SV15       |
14   TV   SV52     TM ---->   |
15   TP   LN4      (30000)        Store illegal line number
16   TV   TM       (30000)        Store entry for error print
17   RA   SV15     CN1        }
20   RA   SV16     CN1        }   Set for next entry
21   TP   CN52     LN3            Set LN3 to zero
22   MJ   0        LN2            Exit
23   TP   SV15     TM         }
24   RS   TM       SV55       }   Store no. of illegal line no's. in index
25   LT   43       TM         }
26   TU   SV54     SV32       }
27   TU   SV53     SV33       }   Set up initial addresses
30   IJ   TM       SV32           Finished with all error prints
31   MJ   0        SV43           If so ---> SV43
32   TV   (30000)  SV34           Set entry to LNP no. 3
33   TP   (30000)  LN4            Send line number to LNP no. 3
34   RJ   LN2      (30000)        Go to LNP for error print
35   RA   SV32     CN14       }
36   RA   SV33     CN14       }   Set for next print
37   MJ   0        SV30           Go back for remaining items
40   TP   CX36     XX10       }
41   RP   30003    SV43       }   Set print for label check
42   TP   CX20     XX13       }
43   TV   SV55     SV15       }
44   TV   SV54     SV16       }   Set initial error entries
45   RJ   SV45     SV46
46   MJ   0        WX
47   00   0        PA11       }
50   00   0        PA16       }   Line number error print entry addresses
51   00   0        PA23       }
52   00   0        PA30       }
53   00   BR       0
54   00   BR1      BR1
55   TP   LN4      BR         }
56   TP   CX35     UP3        }   Print:  SENTENCE - - - (DELETE)
57   RJ   UP2      UP         }
60   MJ   0        (30000)    }
61   TP   XX24     UP3        }
62   RJ   UP2      UP         }   Print:  TOO MANY CORRECTIONS  300 IS
63   MJ   0        ET7        }     MAXIMUM
     CA   SV64                }
```

```
     IA   WX
 0   EF   0        VF3
 1   RP   10170    WX3      Read label
 2   ER   10000    BU
 3   ER   0        A        Check parity
 4   ZJ   JL12     WX5
 5   TP   CX25     LC3      Check label
 6   TP   CX30     LC4
 7   RJ   LC2      LC

10   ZJ   WX13     WX11     Was label in error?
11   EF   0        VF2      If so, rewind tape & stop set to re-enter
12   MS   0        SV2
13   EF   0        VF       If not start read
14   TV   RB53     PB7
15   TV   CN22     PB4
16   TV   CN23     PB5
17   TP   CN7      TM5

20   TU   CN46     PB6
21   TU   CN46     PB7
22   TV   CN20     FT20
23   TU   CN20     CN17
24   TV   CN25     FT5      Set initial buffer storage address
25   TP   CN4      TM4      Set block index
26   TV   CN24     FT2      Set initial drum address
27   RP   10170    WX31     Read 1 block
30   ER   10000    BU
31   ER   0        A        Check parity
32   ZJ   JL21     WX33
33   TV   WX35     MP2
34   RJ   MP5      MP       Check:  end of tape
35   TV   MN25     MP5      If so --> ET1
36   EJ   CX6      ET1
37   MJ   0        FT23     If not, to FT23
     CA   WX40
```

```
     IA   FT
 0   RJ   PB35     PB       Process blockette
 1   ER   10000    A        Read 1st word
 2   TP   A        30000    Store 1st word
 3   EJ   CX6      ET       End of Tape?  If so --> ET
 4   RP   10023    FT6      Read rest of blockette
 5   ER   10000    30000
 6   RA   FT2      CN5      Increment addresses
 7   RA   FT5      CN5
10   IJ   TM4      FT       Finished with block?
11   ER   0        A        Check parity
12   ZJ   JL       FT13
```

```
13   RJ   FT13     FT14          End of tape switch
14   RJ   FT14     FT15          Last block of group switch
15   IJ   TM5      FT24          Test:  Is buffer full?  If not go to FT24
16   EF   0        VF10          If so, stop tape
17   RP   33220    FT21   }
20   TP   BU       30000  }      Transfer corrections to drum
21   RJ   ET20     ET11          Out to print errors and process delete
                                  item no's.
22   RJ   FT14     FT26          Set last block of group switch to exit to
                                  FT23 and go to FT26
23   TU   FT20     PB            Reset first buffer address
24   TP   CN4      TM4           Set block index
25   MJ   0        FT            Go back to FT
26   EF   0        VF            Start read
27   TV   WX2      FT2    }
30   SP   FT2      0      }
31   SA   CN       0      }      Reset initial buffer address
32   TV   A        FT5    }
33   TP   CN10     TM5           Set buffer index
34   TP   CN4      TM4           Set block index
35   RA   FT20     CN11          Set for next drum transfer
36   TV   CN41     RB50
37   MJ   0        FT
     CA   FT40


     IA   PB
 0   TP   BU       A             Sentence number to A
 1   EJ   CX7      PB22          Is it all spaces?  If so, go to PB22
 2   TP   A        LN4    }
 3   RJ   LN2      LN1    }      Process line number
 4   TP   LN3      30000         Store line no. in table
 5   TP   CN17     30000         Store directory word in table
 6   TP   30000    A      }
 7   TJ   30000    PB74   }      Test:  Is item out of sequence?  If so,
                                  go to PB74
10   TU   PB6      PB7    }
11   RA   PB6      CN14   }
12   RA   PB4      CN1    }      Set for next table entry
13   RA   PB5      CN1    }
14   IJ   CX26     PB33          Have 300 corrections been processed?
                                  If not, go to PB33
15   TV   PB35     FT     }
16   RA   FT       CN     }      If so, set to ignore remaining corrections
17   RJ   FT13     PB35          Back to read, remainder of block

20   EF   0        VF10          Stop tape
21   MJ   0        SV61          Go to SV61 to indicate error
22   TP   PB       A      }
23   AT   CN13     PB24   }      Set up test
24   TP   PB1      A      }
25   EJ   RB62     DL     }      Test:  Is the first word of sentence
                                  'DELETE'? If so, go to DL
```

```
26   MJ   0       PB36        If not, go to PB36
27   SP   PB5     17
30   SS   CN14    0           Continuation of last sentence. Increase
31   TU   A       PB32        word count in directory word
32   RA   30000   CN5
33   RA   PB      CN12        Set for next blockette
34   RA   CN17    CN12        Set for next table entry
35   MJ   0       30000       Exit
36   TU   PB24    PB41
37   TP   CX7     A
     CA   PB40

     IA   PB40
40   RP   20020   PB27        Find 1st word that is not all spaces
41   TJ   30000   PB42
42   TP   RB63    A
43   ST   Q       TM          Set address of 1st word that is not all
44   LA   A       17          spaces into PB24 and PB47
45   AT   PB24    PB24
46   TU   A       PB47
47   TP   30000   A           If word is delete, go to DL
50   EJ   RB62    DL
51   RP   20005   PB27        If word is not delete, test alignment of word
52   EJ   RB55    PB53
53   TP   RB64    A
54   ST   Q       TM
55   LA   A       1           Set number of shifts needed to put 1st
56   SA   TM      1           significant character at 1st character
57   TV   A       PB65        position

60   TU   PB47    PB64
61   TP   CN37    TM
62   TV   DL2     PB66
63   MJ   0       PB70
64   SP   30000   44
65   SA   30000   30000       Align sentence and store
66   TP   A       30000
67   RA   PB66    CN
70   TU   PB64    PB65
71   RA   PB64    CN13
72   IJ   TM      PB64
73   MJ   0       RB67        Go to RB67
74   TV   PB77    PB7         Set to bypass out of sequence test
75   TV   PB4     CN30        Set address of 1st out of sequence item
                              into CN30
76   TV   PB77    RB53
77   MJ   0       PB10
     CA   PB100
```

```
     IA   DL
 0   TU   PB24    DL2
 1   RP   30006   DL3     } Put aligned delete command into TM6-TM13
 2   TP   30000   TM6     }
 3   TV   PB4     DL5     }
 4   RP   10002   DL6     } Fill output lines in table with spaces
 5   TP   CX7     30000   }
 6   TP   CN15    Q       }
 7   QT   TM10    A       }
                          } Test: Is more than one sentence to be
10   EJ   RB54    DL31    }       deleted?
11   SP   TM11    44      }
12   SA   TM10    30      } Position sentence number
13   TP   A       TM12    }
14   TV   DL5     DL27    }
15   TV   DL5     RB47    } Set up storage address for line number
16   TV   PB5     RB52    }
17   RJ   RB44    RB40      Find length of number
20   LQ   CN15    Q22 ---->
21   MJ   0       DL27
22   TP   DL40    Q  ---->
23   MJ   0       DL27
24   TP   DL36    Q  ----> Set mask into Q
25   MJ   0       DL27
26   TP   DL37    Q  ---->
27   QS   TM12    30000     Set line number into table and into A

30   MJ   0       RB65      Go to RB65
31   TV   PB4     RB5     }
32   TV   PB4     RB15    } Set storage address
33   TV   PB4     RB36    }
34   LQ   CN15    Q22       Set mask into Q
35   MJ   0       RB        Go to RB
36   77   77770   0
37   77   77777   70000
40   77   77000   0
     CA   DL41
```

```
     IA   RB
 0   QT   TM12    A       } Mask out character
 1   TP   A       A       }
 2   RP   20005   RB4     } Check alignment of THRU to determine
 3   EJ   CX      RB10    } length of 1st number
 4   TP   CN6     Q       } Character was not T,H,R,U or space.  Put
 5   QS   TM10    30000   } last character of TM10 into table
 6   TV   PB5     DL5       Set new output storage
 7   MJ   0       DL14

10   TP   RB64    A
11   ST   Q       TM        Was first character of TM12 a T?
12   TJ   CN3     RB25
```

```
13   SP   TM11    44        }
14   SA   TM10    36        }   If so, align TM10 and TM11
15   TP   A       30000         Store 1st line number in table
16   LA   TM      A1        )
17   SA   TM      1         }
                            }   Set no. of shifts to align second line no.
20   TV   A       RB22      )
21   SP   TM13    44        )
22   SA   TM12    30000     }   Align second line no. and put into TM12
23   TP   A       TM12      )
24   MJ   0       RB6           Go to RB6
25   LA   A       1         )
26   SA   TM      1         }   Set no. of shifts to align first line no.
27   TV   A       RB32      )

30   TP   CN3     A         }
31   ST   TM      TM        }   Adjust index in TM
32   SP   CN50    30000     }   Put mask into Q
33   TP   A       Q         }
34   SP   TM11    44        )
35   SA   TM10    36        }   Store 1st line no. in table
36   QS   A       30000     )
37   MJ   0       RB16          Go to RB16
     CA   RB40

     IA   RB40
40   TP   CN2     TM            Set space test index to 3
41   LQ   TM12    Q6        )
42   LQ   Q       6         }   Mask out character and check for space
43   QT   CN6     A         }   If space is found, exit
44   EJ   CN      30000     )
45   RA   RB44    CN1       }   If no space is found, set to check next
46   IJ   TM      RB42      }   character
47   TP   TM12    30000         If no more characters are to be tested,
                                store all of TM12 in line no. position.
50   TV   PB4     30000     }   Set table address of delete item into
                            }   delete list
51   RA   RB50    CN            Set for next delete item
52   TP   TM12    30000         Store 2nd line number in table
53   MJ   0       PB74          Exit to PB74
54   00   6       50000     )
55   01   27304   63066     )
56   01   01273   04630     )
57   01   01012   73046     )
60   01   01010   12730     }   Constants
61   01   01010   10127     )
62   27   30463   06630     )
63   00   0       20017     )
64   00   0       20005     )
65   TP   A       TM12          Store line number in TM12 and go to RB50
66   MJ   0       RB50
67   TP   TM6     A         )
70   EJ   RB62    DL3       }   Test: Is first word of aligned sentence
71   MJ   0       PB27      }   'Delete'? If so, go to DL3. If not, go to
     CA   RB72                  PB27
```

```
     IA    ET
 0   RJ    FT13     FT4        Set end of tape switch to exit to ET1 and
                                go to FT4
 1   EF    0        VF10       Stop tape
 2   TU    PB       ET3        Set next line number address
 3   TP    30000    A          }
 4   EJ    CX6      ET7        } End of tape? If so, go to ET7
 5   RJ    PB35     PB1        If not, process line number
 6   MJ    0        ET2        Go back for next item
 7   EF    0        VF2        Rewind tape 4

10   RJ    FT17     FT17       Transfer last group of corrections to drum
11   RJ    SV45     SV23       Print Line Number Processor Errors for
                                ordinary sentences
12   TP    RB50     TM3        )
13   RS    TM3      CN41       } Set to process delete command item numbers
14   SP    CN41     17         }
15   TU    A        ET22       )
16   IJ    TM3      ET22       Finished with all delete command item
                                numbers? If not go to ET22
17   RJ    SV45     SV23       Print Line Number Processor errors

20   RJ    ET20     ET21       Exit when used as subroutine
21   MJ    0        ET70       Exit after end of tape has been found
22   TP    30000    A          )
23   TV    A        ET31       }
24   LA    A        17         } Set address of item number to be processed
25   TU    A        ET27       )
26   TP    CN       TM1
27   TP    30000    LN4

30   RJ    LN2      LN         }
31   TP    LN3      30000      } Process line number and store
32   RA    ET27     CN13       }
33   RA    ET31     CN         } Set to process next item no. of delete
                                command
34   IJ    TM1      ET27       Finished with both line numbers? If not go
                                to ET27
35   RA    ET22     CN13       } If so, set for next delete item and go back
36   MJ    0        ET16       } to ET16
37   TP    CN52     TM17       Set out of sequence item counter to zero

40   TP    CN52     TM15
41   TP    CX37     XX10       )
42   RP    30003    ET44       } Reset tape label error print for servo 3
43   TP    CX15     XX13       ) and program tape
44   TP    CX34     UP3        }
45   RJ    UP2      UP         } Print: PROGRAM TAPE
46   TP    CN4      TM10       }
47   TP    CN4      TM5        } Set up write fill counter and write fill
                                temp.
```

```
50   TU   CN46    ET52
51   TP   CN43    TM1
52   TP   30000   A
53   ZJ   ET56    ET54      Find address of first non zero table entry
54   RA   ET52    CN14
55   IJ   TM1     ET52
56   SP   ET52    0
57   SS   CN46    70

60   AT   CN      TM
61   RS   CN43    TM
62   TU   ET52    MN42      Set up address of 1st non zero table entry
63   TU   ET52    MN63
64   RA   ET52    CN13
65   TU   A       MN54
66   RP   30003   MG        Set up printout for out of sequence items
67   TP   CN55    XX21      and go to MG

70   TP   CN30    A
71   EJ   CN22    GQ        If 1st table item was a delete adjust number
72   TJ   CN51    SR        of in sequence items
73   TV   PB4     CN30
74   MJ   0       SR        Go to SR
     CA   ET75

     IA   GQ
 0   RA   CN30    CN1       Set new number of in sequence items
 1   MJ   0       SR
     CA   GQ2

     IA   XX                UP parameters & XS3 codes for printouts
 0   01   01010   10101
 1   47   30543   23022     ΔΔΔΔΔΔ Merge.
 2   26   51545   43026
 3   66   34515   00166     Correction Tape
 4   24   52300   12277
 5   71   54515   03201
 6   66   24523   00165     Wrong Tape Servo _ _ _ Should be _ _ _
 7   30   54705   10177

10   01   01010   10101
11   65   33516   74627
12   01   25300   17777
13   01   01010   10101
14   01   01010   10101
15   01   01010   10101
16   65   30506   63050
17   26   30010   10101

20   17   27304   63066     Sentence -----(Delete)
21   30   43010   10101
22   52   54513   25424
23   47   01662   45230     Program Tape
```

```
24   00   XX25    6
25   66   51510   14724
26   50   73012   65154      Too many corrections 300 is maximum
27   54   30266   63451
30   50   65010   60303
31   01   34650   14724
32   72   34476   74701
33   52   24543   46673
34   01   54305   43024      Parity reread fails:  to reread start.
35   27   01312   43446      To ignore error set A not = 0 and start
36   65   22016   65101
37   54   30543   02427

40   01   65662   45466
41   22   01665   10134
42   32   50515   43001
43   30   54545   15401
44   65   30660   12401
45   50   51660   17601
46   03   01245   02701
47   65   66245   46622

50   52   24543   46673      Parity Error ignored
51   01   30545   45154
52   01   34325   05154
53   30   27227   77777
54   00   XX50    4
55   01   01010   10101
56   52   54513   25424    ΔΔΔΔΔΔProgram now on tape 5.
57   47   01505   17101

60   51   50016   62452
61   30   01102   27777
62   01   01010   10101   ΔΔΔΔΔΔput 1500 ft. tapes on s. 3 and 4
63   52   67660   10410
64   03   03013   16622
65   01   66245   23065
66   01   51500   16522
67   01   06012   45027

70   01   07227   77777
71   00   XX62    7
     CA   XX72

     IA   SR
0    TV   PB4     CN43
1    RS   CN43    CN22       Set total number of items
2    LT   43      CN43
3    TV   CN30    CN44
4    RS   CN44    CN22       Set number of in sequence items
5    LT   43      CN45
6    SP   CN45    20
7    AT   CN33    ZR10       Set up transfer command
```

```
10   TU   CN46    XZ
11   SP   CN45    20
12   AT   XZ      A
13   TU   A       XZ1
14   TP   XZ1     TM
15   TP   CN37    TM1
16   TV   XZ20    SR21
17   TV   XZ15    SR23

20   RA   TM      CN14
21   TU   A       30000
22   RA   TM      CN14
23   TU   A       30000      Set addresses into compares
24   RA   SR21    CN35
25   RA   SR23    CN35
26   IJ   TM1     SR20
27   TV   SR70    XZ4

30   TP   XZ4     A
31   AT   CN61    XZ31
32   AT   CN61    XZ51       Set initial exits of switches
33   AT   CN61    XZ71
34   TV   SR71    XZ21
35   TP   XZ21    A
36   AT   CN1     XZ41
37   AT   CN1     XZ61
40   AT   CN1     XZ101
41   TV   SR72    WT10
42   TP   WT10    A
43   AT   CN61    WT32
44   TV   SR73    WT21       Set switch J to WT44
45   TV   SR74    WT43       Set switch L to WT46
46   TV   SR75    NZ10       Set switch M to NZ26
47   TP   CN45    A
                             Set no. compares index.
50   ST   CN      TM
51   RA   CN45    CN47       Form new no. of in sequence items
52   TJ   CN43    SR64       Test: Is total no. of items > than new no. of
                             in sequence items?
53   TP   CN43    TM1
54   TV   PB4     SR56
55   RP   10036   SR57       Fill remainder of table with largest number
56   TP   CN50    30000
57   RJ   ZR10    SR66
60   TV   SR77    XY11       Set parity reread exits for program tape
61   TV   SR76    XY35
62   RP   20005   ET37       Reset tape read codes for servo 3
63   RS   VF      VF15
64   TP   CN45    TM1
65   RS   TM1     CN
66   TV   CN54    ZR5
67   MJ   0       XZ
70   0    0       WT
71   0    0       WT51
```

```
72    0     0        NZ
73    0     0        WT44
74    0     0        WT46
75    0     0        NZ26
76    0     0        XY65
77    0     0        XY73
      CA    SR100

      IA    XZ
 0    TP    30000    A        }
 1    TJ    30000    XZ5      }     Compare A
 2    TU    XZ1      WT1
 3    TU    WT50     XZ1
 4    MJ    0        WT             Switch A
 5    TU    XZ       WT1
 6    RA    XZ       CN14
 7    IJ    TM       XZ4
10    TU    WT50     XZ
11    MJ    0        XZ4
12    TP    30000    A        }
13    TJ    30000    XZ17     }     Compare B
14    TU    XZ13     WT2
15    TU    WT50     XZ13
16    MJ    0        XZ21
17    TU    XZ12     WT2
20    TU    WT50     XZ12
21    MJ    0        WT51           Switch B
22    TP    30000    A        }
23    TJ    30000    XZ27     }     Compare C
24    TU    XZ23     WT12
25    TU    WT50     XZ23
26    MJ    0        XZ31
27    TU    XZ22     WT12
30    TU    WT50     XZ22
31    MJ    0        WT11           Switch C
32    TP    30000    A        }
33    TJ    30000    XZ37     }     Compare D
34    TU    XZ33     WT13
35    TU    WT50     XZ33
36    MJ    0        XZ41
37    TU    XZ32     WT13
      CA    XZ40

      IA    XZ40
40    TU    WT50     XZ32
41    MJ    0        WT53           Switch D
42    TP    30000    A        }
43    TJ    30000    XZ47     }     Compare E
44    TU    XZ43     WT23
45    TU    WT50     XZ43
46    MJ    0        XZ51
47    TU    XZ42     WT23
```

```
 50   TU   WT50    XZ42
 51   MJ   0       WT22        Switch E
 52   TP   30000   A       }
 53   TJ   30000   XZ57    }   Compare F
 54   TU   XZ53    WT24
 55   TU   WT50    XZ53
 56   MJ   0       XZ61
 57   TU   XZ52    WT24

 60   TU   WT50    XZ52
 61   MJ   0       WT55        Switch F
 62   TP   30000   A       }
 63   TJ   30000   XZ67    }   Compare G
 64   TU   XZ63    WT34
 65   TU   WT50    XZ63
 66   MJ   0       XZ71
 67   TU   XZ62    WT34

 70   TU   WT50    XZ62
 71   MJ   0       WT33        Switch G
 72   TP   30000   A       }
 73   TJ   30000   XZ77    }   Compare H
 74   TU   XZ73    WT35
 75   TU   WT50    XZ73
 76   MJ   0       XZ101
 77   TU   XZ72    WT35
100   TU   WT50    XZ72
101   MJ   0       WT57        Switch H
      CA   XZ102

      IA   WT
  0   RJ   XZ4     XZ12        Set switch A to exit to WT1 and go to XZ12
  1   TP   30000   A       }
  2   TJ   30000   WT6     }   Compare I
  3   TU   WT2     NZ1
  4   TV   XZ20    NZ22
  5   MJ   0       WT10
  6   TU   WT1     NZ1
  7   TV   XZ10    NZ22
 10   MJ   0       NZ          Switch I
 11   RJ   XZ31    XZ32        Set Switch C to exit to WT12 and go to XZ32
 12   TP   30000   A       }
 13   TJ   30000   WT17    }   Compare J
 14   TU   WT13    NZ2
 15   TV   XZ40    NZ23
 16   MJ   0       WT21
 17   TU   WT12    NZ2
```

```
20   TV   XZ30     NZ23
21   MJ   0        WT44        Switch J
22   RJ   XZ51     XZ52        Set Switch E to exit to WT23 and go to XZ52
23   TP   30000    A       }
24   TJ   30000    WT30    }   Compare K
25   TU   WT24     NZ12
26   TV   XZ60     NZ24
27   MJ   0        WT32

30   TU   WT23     NZ12
31   TV   XZ50     NZ24
32   MJ   0        NZ11        Switch K
33   RJ   XZ71     XZ72        Set Switch G to exit to WT34 and go to XZ72
34   TP   30000    A       }
35   TJ   30000    WT41    }   Compare L
36   TU   WT35     NZ13
37   TV   XZ100    NZ25

40   MJ   0        WT43
41   TU   WT34     NZ13
42   TV   XZ70     NZ25
43   MJ   0        WT46        Switch L
44   TV   WT10     WT21    }   Set Switch J to exit to NZ1 and go to
45   MJ   0        WT23    }   compare K
46   TV   WT32     WT43    }   Set Switch L to exit to NZ11 and go to
47   MJ   0        NZ1     }   compare M

50   00   CN50     0
51   TV   XZ4      XZ21
52   MJ   0        XZ22
53   TV   XZ31     XZ41
54   MJ   0        XZ42
55   TV   XZ51     XZ61
56   MJ   0        XZ62
57   TV   XZ71     XZ101
60   MJ   0        WT1
     CA   WT61

     IA   NZ
 0   RJ   WT10     WT12
 1   TP   30000    A       }
 2   TJ   30000    NZ6     }   Compare M
 3   TU   NZ2      ZR
 4   TV   NZ23     ZR15
 5   MJ   0        NZ10
 6   TU   NZ1      ZR
 7   TV   NZ22     ZR15
```

```
10   MJ   0       NZ26
11   RJ   WT32    WT34
12   TP   30000   A       }
13   TJ   30000   NZ17    }   Compare N
14   TU   NZ13    ZR1
15   TV   NZ25    ZR16
16   MJ   0       ZR
17   TU   NZ12    ZR1

20   TV   NZ24    ZR16
21   MJ   0       ZR
22   0    0       30000   )
23   0    0       30000   }   Storage for return address { Compare I
24   0    0       30000   }                               { Compare J
25   0    0       30000   )                               { Compare K
                                                           { Compare L
26   TV   NZ21    NZ10
27   MJ   0       NZ12        Set Switch M to ZR and go to compare N
     CA   NZ30

     IA   ZR
 0   TP   30000   A       }
 1   TJ   30000   ZR12    }   Compare P
 2   TU   ZR1     ZR5
 3   TV   ZR16    ZR7
 4   RP   30002   ZR6     }
 5   TP   30000   30000   }   Transfer table item to buffer
 6   RA   ZR5     CN1         Set next buffer address
 7   IJ   TM1     30000       Test:  finished with this group of items?
10   RP   30000   30000   }
11   TP   BU      TB      }   Yes:   Transfer sorted string back to table
12   TU   ZR      ZR5
13   TV   ZR15    ZR7
14   MJ   0       ZR4
15   00   0       30000
16   00   0       30000
     CA   ZR17

     IA   MG
 0   EF   0       VF3     )
 1   RP   10170   MG3     }   Read label block
 2   ER   10000   BU      )
 3   ER   0       A       }
 4   ZJ   JL24    MG5     }   Check parity
 5   TP   CX24    LC3     )
 6   TP   CX30    LC4     }   Check label
 7   RJ   LC2     LC      )
10   ZJ   MG13    MG11        Was label correct?
11   EF   0       VF2     }
12   MS   0       MG      }   If not, rewind tape and stop set to re-enter
13   EF   0       VF13    )
14   RP   10170   MG16    }   If so, write label block on tape 5
15   EW   10000   BU      )
```

```
16   EF   0       VF
17   SP   MN1     17
20   TU   A       MN5
21   SP   CN24    17
22   TU   A       MN13
23   TV   MN1     MN36
24   TV   CN24    MN12
25   TP   CN4     TM4

26   TP   CN4     TM2
27   TP   CN62    TM1
30   TP   CN62    TM3
31   TU   MN5     NW23
32   TV   MN77    MN64
33   TU   MN5     KW3
34   TV   CN26    MN63
35   TV   NW56    MN65
36   TU   CN26    KW5
37   TU   CN26    KW6
40   TP   CX7     LN3
     CA   MG41

     IA   MN
 0   RP   10170   MN2
 1   ER   10000   BU
 2   ER   0       A
 3   ZJ   XY52    MP
 4   TP   CN4     TM4
 5   TP   BU      A
 6   TP   CN      TM14
 7   RJ   NR25    MN33
10   MJ   0       MN11
11   RP   10024   MN13
12   ER   10000   BU
13   TP   BU170   A
14   EJ   CX6     NR
15   RA   MN5     CN12
16   MJ   0       MN17
17   RA   MN12    CN5
20   RA   MN13    CN12
21   IJ   TM4     MN5
22   ER   0       A
23   ZJ   XY55    MN24
24   RJ   MN24    MN25
25   IJ   TM2     MN4
26   EF   0       VF10
27   TP   CN4     TM14
30   TU   MN5     MN32
31   TV   CN42    NR25
32   TP   30000   A
33   EJ   CX7     MN36
34   TP   A       LN4
```

Start read

Set up initial buffer addresses

Set block index

Set buffer index

Set No. of blockettes into TM1 and TM3

Set initial correction buffer address into KW5 and KW6

MN must equal MG41

Read block

Check Parity

Set block index

Read blockette

Was line no. all Z's? If so, go to NR

Set for next blockette

Finished with block?

If so, check parity

End of tape switch

Finished filling buffer?

If so, stop tape

Set to process line numbers of last block

Test: Is line number all spaces? If so, go to MN36

```
35    RJ    LN2      LN1          Process line number
36    TP    LN3      30000        Put processed line number back in buffer
37    TP    LN3      A
      CA    MN40

      IA    MN40
40    ZJ    NR17     NR23         Test: was line number illegal?  If so, go
                                  to NR23.  If not, go to NR17
41    TP    TM15     LN3          Put last legal line number into LN3
42    TP    TB       A            Correction item number to A
43    TJ    LN3      MN54   }     Test: does correction refer to this group
44    EJ    LN3      MN54   }     of program items?
45    TV    MN63     MN46
46    TP    CN50     30000        If not, fill next correction buffer loca-
                                  tion with largest number
47    MJ    0        NW46         Go to NW46

50    TV    MN52     MN64         Set new exit for drum transfer command
51    TV    MN67     MN43   }     Set new exits for tests at MN43 and MN44
52    RJ    MN44     MN66   }     and go to MN66
53    EJ    30000    NW
54    TP    TB1      A      }     Test:  Is this a delete sentence?
55    SJ    MN56     NW21   }
56    RJ    MN56     MN57         Equal Switch for drum items
57    TU    A        MN65         Set drum address into MN65
60    TV    A        TM14   )
61    SA    CN51     17     }     Set number of words into MN64
62    TU    A        MN64   )
63    TP    TB       30000        Item number to correction buffer
64    RP    30000    MN50   }     Remainder of item from drum to correction
65    TP    30000    30000  }     buffer
66    SP    MN63     17     }     Set address of last item transferred
67    TU    A        MN53   }
70    RA    MN63     TM14   }     Set next correction buffer location
71    AT    CN       MN65   }
72    RA    MN42     CN14   )
73    TU    A        MN63   }     Set for next table entry
74    AT    CN13     MN54   )
75    IJ    CN43     MN42         Have corrections been exhausted?
76    MJ    0        LP           Out to block next correction phase
77    00    0        MN50
100   TV    MN101    MN14   }     Set to ignore additional lines of Z's
101   MJ    0        MN15   }
102   RJ    NW44     MN54
103   RJ    MN56     MN56
104   MJ    0        NW3
      CA    MN105
```

```
     IA    NW
 0   RJ    MN56     MN102      Set equal switch for drum items
 1   RJ    NW44     NW44
 2   MJ    0        MN72
 3   RS    MN63     TM14   }
 4   AT    CN       MN65   }  Reset transfer command
 5   TU    MN54     NW7    }
 6   TU    MN42     NW10   }  Set up test
 7   TP    TB1      A      }  Test: Is item to be deleted? If not, go
10   TJ    TB       NW14   }  to NW14
11   RA    NW10     CN14       If so, set to test next item
12   IJ    CN43     NW7        Test: Are there any more correction items
                                to be tested? If so, go to NW7
13   MJ    0        LP         Out to set end of corrections
14   TU    NW10     MN42   )
15   TU    NW10     MN63   }
16   TU    NW10     MN54   }  Reset table addresses
17   RA    MN54     CN13   )
20   MJ    0        MN42       Go to MN42

21   TU    MN42     NW24
22   TU    MN54     NW34
23   TP    BU       A      }  Test: Is program item greater than 1st
24   TJ    30000    NW26   }  delete address? If not, go to NW26
25   MJ    0        NW31       If so, go to NW31
26   RA    NW23     CN12       Set to test next item
27   IJ    TM1      NW23       Have all items been tested? If not, go
                                back to NW23
30   MJ    0        MN45       If so, go to MN45
31   LQ    NW23     Q25    )
32   TV    Q        NW36   }  Set address of item from above into NW35
33   TU    NW23     NW35   )
34   TP    TB1      A          Last address of delete item to A
35   TJ    30000    NW44       Test: Is this program item to be deleted?
36   TP    CN52     30000      Yes, store zero in line number position so
                                that item will be dropped
37   RA    NW35     CN12   )
                             }  Set for next program item
40   RA    NW36     CN5    }
41   RA    NW23     CN12   )
42   IJ    TM1      NW34       Test: Finished with current group of pro-
                                gram items? If not, go back to test next item
43   MJ    0        NW52       If so, go to NW52
44   RJ    NW44     NW45
45   MJ    0        NW5
46   RJ    SV45     SV23       Print line number processor errors
47   MJ    0        KW
     CA    NW50
```

```
     IA   NW52
52   LQ   NW24    Q25
53   TV   Q       NW54     Set zero as 1st address of delete command
54   TP   CN52    30000
55   MJ   0       MN45
56   00   0       CB1
     CA   NW57

     IA   KW
 0   TP   TM10    A        Set up write fill index
 1   AT   CN      TM7
 2   EF   0       VF12     Start write
 3   TP   30000   A        Current program item no. to A
 4   ZJ   KW5     KW34     Is it to be deleted?
 5   TJ   30000   KW23     Is current correction item > than current
                           program item?
 6   EJ   30000   KW14     Is current correction item = to current prog. item
 7   TU   KW5     TM       Set correction item address into TM.
10   RA   KW5     CN12     Set for next correction item
11   TU   A       KW6
12   RJ   BJ34    BJ       Out to set up and write blockette
13   MJ   0       KW3      Back for next item
14   RA   KW3     CN12     Set for next program item
15   TP   A       KW20
16   IJ   TM3     KW20     Finished with all program items
17   MJ   0       UN32     If so, go to UN32 to write remainder of
                           block
20   00   30000   30000    Test: Is next program blockette a contin-
21   EJ   CX7     KW14     uation of last sentence?
22   MJ   0       KW7      If not, go to KW7 to write correction item
23   TU   KW3     TM
24   RA   KW3     CN12
25   TP   A       KW31
26   IJ   TM3     KW30     Test: finished with all program items?  If
                           not, go to KW30
27   MJ   0       UN       If so, go to UN
30   RJ   BJ34    BJ       Set up and write next blockette
31   00   30000   30000
32   EJ   CX7     KW23
33   MJ   0       KW3
34   RA   KW3     CN12
35   IJ   TM3     KW3      Finished with all program items?  If not,
                           go to KW3
36   MJ   0       UN1      If so, go to UN1
     CA   KW37
```

```
     IA   BJ
 0   IJ   TM7     BJ26          Has whole block been set up?  If not, go
                                to BJ26
 1   RJ   BJ1     BJ11
 2   RJ   BJ1     BJ13
 3   RJ   BJ1     BJ15
 4   RJ   BJ1     BJ17
 5   RJ   BJ1     BJ21
 6   RJ   BJ1     BJ23
 7   TV   CN53    BJ27          Reset store address of 1st write
10   RJ   BJ1     BJ34          Reset switch to write 1st blockette
11   RP   10024   BJ25    }     Write 1st blockette and go to BJ25 to set
12   EW   10000   30000  }     up new '1st Blockette'
13   RP   10024   BJ25    }     Write 2nd blockette and go to BJ25 to set
14   EW   10000   30000  }     up new '2nd Blockette'
15   RP   10024   BJ25    }     Write 3rd blockette and go to BJ25 to set
16   EW   10000   30000  }     up new '3rd Blockette'
17   RP   10024   BJ25    }     Write 4th blockette and go to BJ25 to set
20   EW   10000   30000  }     up new '4th Blockette'
21   RP   10024   BJ25    }     Write 5th blockette and go to BJ25 to set
22   EW   10000   30000  }     up new '5th Blockette'
23   RP   10024   BJ25    }     Write 6th blockette and go to BJ25 to set
24   EW   10000   30000  }     up new '6th Blockette'
25   RJ   BJ25    BJ26
26   LQ   TM      Q25    }
27   TV   Q       BJ12   }     Set address from TM into write command
30   RA   BJ27    CN1           Set for next write command
31   IJ   TM5     BJ34          Has full block been written?
32   TP   CN4     TM5           Reset index
33   MJ   0       BJ7           Back to BJ7
34   MJ   0       30000         Exit
     CA   BJ35

     IA   UN
 0   RJ   BJ34    BJ            Set up and write last block
 1   TV   BJ27    TM11   }     Store index and address of next write
 2   TP   TM5     TM10   }     command to be set up
 3   RJ   BJ25    BJ     }
 4   IJ   TM5     UN3    }     Write remainder of block
 5   EF   0       VF10          Stop tape
 6   RJ   UN56    UN34          Transfer unwritten blockettes to temp buffer
 7   MJ   0       MG16          Go back to MG16 for next group of program
                                tape items

10   TP   CN52    TM14   }
11   TP   CN50    LN3    }     Put largest number into LN3
12   TP   CN50    30000         Put largest number into end of tape line
13   RJ   UN33    MN42          Out to bring in remaining corrections
14   TP   CX6     30000         Reset all Z's into end of tape line
15   LA   CN30    17     }
16   RA   UN64    TM5    }     Set index for remaining block(s)
17   TU   CN30    TM            Set address of end of tape line into TM
```

```
20   RJ   BJ34    BJ      }
21   IJ   UN64    UN20    }   Write remaining block(s)
22   EF   0       VF10        Stop tape
23   IJ   TM17    UN25        Are there more out of sequence items?
24   MJ   0       RS6         If not, go to RS6
25   TP   CX27    UP3     )
26   TP   TM20    XX20    }   If so, fill in sentence number and print:
27   RJ   UP2     UP      )   SENTENCE - - - OUT OF SEQUENCE

30   RA   UN26    CN13        Set for next item
31   MJ   0       UN23        Back to test
32   TU   KW5     TM      }   Set correction item address into TM and go
33   MJ   0       UN      }   to UN
34   TU   CN42    UN37
35   TP   CN4     TM
36   TV   CN60    UN42        Set temp buffer address into transfer command
37   SP   30000   17      )
                           }   Set address of item into transfer command
40   TU   A       UN42    )
41   RP   30024   UN43    }   Move blockette to temp buffer so it will
42   TP   30000   30000   }   not be overwritten by next read in
43   RA   UN37    CN14    }
44   RA   UN42    CN5     }   Set for next blockette
45   IJ   TM      UN37        Have all blockettes been transferred?  If
                               not, go to UN37
46   TP   CN60    A       )
47   TP   A       BJ12    |
                           |
50   AT   CN5     BJ14    |
51   AT   CN5     BJ16    }   Reset addresses of write commands
52   AT   CN5     BJ20    |
53   AT   CN5     BJ22    |
54   AT   CN5     BJ24    )
55   TP   TM10    TM5
56   MJ   0       30000       Exit
57   TP   A       TM20        Store sentence no. of out of sequence item

60   TP   CN52    LN3         Set a zero into LN3 so item will be deleted
61   RA   UN57    CN      }   Set for next out of sequence item
62   RA   TM17    CN      }
63   MJ   0       MN36
64   0    0       14
     CA   UN65

     IA   NR
 0   TV   MN12    CN30        Set address of end of tape line into CN30
 1   TP   TM4     TM16        Store block index
 2   RJ   MN24    MN100       Set End of Tape Switch to exit to NR3 and
                               go to MN100
 3   EF   0       VF10        Stop tape
 4   TP   CN30    A       )
 5   SS   CN54    0       }   Compute no. of blockettes read in before
 6   DV   CN5     TM3     }   end of tape and set into TM3 and TM1
 7   TP   Q       TM1     )
```

```
10  TP  CN3     A       }  Compute no. of line number left to be
11  ST  TM16    TM14    }  processed
12  TV  CN30    UN12    }  Set address of end of tape line
13  TV  CN30    UN14    }
14  SJ  UN10    NR15       If no line numbers are left to be processed,
                           go to UN10
15  RJ  NR26    MN30       Out to process line numbers
16  MJ  0       UN10       Go to UN10
17  EJ  CX7     NR23       Was line number all spaces?  If so, go to
                           NR23

20  TJ  TM15    UN57       Was previous line number greater than this
                           line number?  If so, go to UN57
21  TP  LN3     TM15       If not, set current line number into TM15
22  TP  CX7     LN3        Put all spaces into LN3
23  RA  MN36    CN5     }  Set for next line number
24  RA  MN32    CN12    }
25  IJ  TM14    [MN32]     Finished with all line numbers
26  MJ  0       MN41       Exit to MN41
    CA  NR27

    IA  MP
 0  TU  MN5     MP1        Set up line number address
 1  TP  30000   A       }  Is line number all Z's?  If so, go to MP6
 2  EJ  CX6     MP6     }
 3  RA  MP1     CN12       If not, set for next line number
 4  IJ  TM4     MP1        Finished with all line numbers?  If not, go
                           to MP1
 5  RJ  MP2     [MN4]      Exit, end of tape not found

 6  EF  0       VF10       Stop tape
 7  LQ  MP1     Q25     }  Set address of end of tape line into CN30
10  TV  Q       CN30    }
11  TP  TM4     TM16       Store block index
12  MJ  0       NR4        Go to NR4
    CA  MP13

    IA  LP
 0  TV  LP2     NR26    }  Set to bypass correction phase
 1  TV  LP2     UN13    }
 2  MJ  0       MN45
 3  ZJ  LP4     JL30       Is parity error to be ignored? If not, go
                           to JL30
 4  TP  XX54    UP3        If so, print: PARITY ERROR IGNORED
 5  RJ  UP2     UP
 6  MJ  0       XY15       Go to XY15 to reposition tape
    CA  LP7
```

Parity Error Routine

```
    IA   JL
 0  TV   FT2     XY5
 1  SP   FT2     17        Set addresses into reread
 2  TU   A       XY10
 3  RJ   FT13    FT13      Clear end of tape indicator
 4  RJ   XY20    JL26      Reread
 5  RJ   JL5     JL6       End of tape?
 6  TP   TM5     A         Buffer index = 0
 7  ZJ   JL10    FT17
10  EF   0       VF        Start read
11  MJ   0       FT13      Exit
12  TV   WX4     JL20      Set exit
13  TV   CN24    XY5
14  SP   CN24    17        Set addresses into reread
15  TU   A       XY10
16  RJ   XY20    JL26      Reread
17  RJ   JL5     JL5       Clear end of tape indicator
20  MJ   0       [30000]   Exit
21  RJ   JL11    JL        Out for set up
22  TV   FT12    JL11
23  MJ   0       FT23      Exit
24  RJ   JL20    JL13      Out for set up
25  MJ   0       MG5       Exit
26  RJ   JL26    JL27
27  RJ   JL26    XY1       Distributor for reread cycles
30  RJ   JL26    XY23
31  RJ   JL26    XY45
32  TP   CX32    UP3       Print error indication
33  RJ   UP2     UP
34  TP   CN52    A         Set A to 0
35  MS   0       LP3       Stop set to re-enter
    CA   JL36
```

Reread Cycles for Parity Error

```
     IA    XY
 0   00    0        XY1
 1   TP    CN4      TM4          Set block index
 2   EF    0        VF1          Start read backward 1 block
 3   TP    CN21     TM6          Set blockette index
 4   RS    XY5      CN           Reset read command
 5   ER    10000    [30000]      Read word
 6   IJ    TM6      XY4          Finished with blockette
 7   RS    XY10     CN12         Set for blockette
10   TP    [30000]  A        }   End of corrections? (i.e., line number
                              }   all Z's)
11   EJ    CX6      XY21     }
12   IJ    TM4      XY3          Finished with block?
13   ER    0        A        }   Check parity
14   ZJ    JL26     XY15     }
15   EF    0        VF4          Move forward 1 block
16   RJ    JL26     JL26         Reset distributor
17   EF    0        VF5          Set to normal bias
20   MJ    0        30000        Exit
21   TV    XY51     JL5
22   MJ    0        XY12
23   EF    0        VF6          Set to low bias
24   TP    CN4      TM4          Set block index
25   TV    XY5      XY31     }   Set initial addresses
26   TU    XY10     XY34     }
27   EF    0        VF3          Start read
30   TP    CN21     TM6          Set blockette index
31   ER    10000    [30000]      Read word
32   RA    XY31     CN           Set for next word

33   IJ    TM6      XY31         Finished with blockette
34   TP    [30000]  A        }   End of corrections
35   EJ    CX6      XY47     }
36   RA    XY34     CN12         Set for next blockette
37   IJ    TM4      XY30         Finished with block
40   ER    0        A        }   Check parity
41   ZJ    XY42     XY16     }
42   TV    XY31     XY5          Parity error:  Reset addresses
43   TU    XY34     XY10           and go back to reread
44   MJ    0        XY1
45   EF    0        VF7          Set to high bias
46   MJ    0        XY24         Go to reread
47   TV    XY51     JL5
50   MJ    0        XY36
51   00    XY51     ET2
```

```
 52   RJ   XY60     XY55          Set XY go to exit to NI and go to XY55
 53   EF   0        VF            Start read
 54   MJ   0        MN4           Go to MN4
 55   TV   MN12     XY51    }     Set address into end of tape test
 56   TU   MN13     XY10    }
 57   RJ   XY20     JL26          Reread
 60   RJ   XY60     XY61          Exit
 61   TP   TM2      A       }     Test:  Is buffer full?  If so, go to MN27
 62   ZJ   XY63     MN27    }     If not, take next instruction
 63   EF   0        VF            Start read
 64   MJ   0        MN25
 65   LQ   XY34     Q25     )
 66   TV   Q        CN30    |
 67   RJ   NR2      NR1     }
 70   TV   MN24     XY60    |
 71   TV   XY72     XY35    |
 72   MJ   0        XY36    )
 73   TV   XY5      CN30
 74   TP   CN4      A
 75   ST   TM4      TM16
 76   RJ   NR2      NR2
 77   TV   MN24     XY60
100   RA   XY60     CN
101   MJ   0        XY22
      CA   XY102

      IA   LC
  0   MJ   0        LC5
  1   RS   TM       TM            Set A = 0 to indicate error
  2   MJ   0        [30000]
  3   00   30000    30000   }     Storage for correct label
  4   00   30000    30000   }
  5   TP   CX5      TM3
  6   TU   LC3      LC11
  7   SP   LC3      17
 10   TU   A        LC47
 11   TP   30000    A       }     Find 1st word that is not all spaces
 12   EJ   CX7      LC15    }
 13   RP   20006    LC17    }     Determine alignment of 1st word that is
 14   EJ   CX10     LC22    }     not all spaces
 15   RA   LC11     CN13    }
 16   IJ   TM3      LC11    }     Find 1st word that is not all spaces.
 17   TP   LC4      UP3     )
 20   RJ   UP2      UP      }     Error - out to print
 21   MJ   0        LC1     )
 22   TP   CX23     A       )
 23   ST   Q        TM      |
 24   EJ   CN37     LC55    }     Set up number of shifts needed to align
 25   LA   A        1       |     sentence
 26   SA   TM       1       |
 27   TV   A        LC36    )
```

```
30   TU   LC11    LC36
31   RA   LC11    CN13      Set up addresses and index
32   TU   A       LC35
33   TV   LC23    LC37
34   TP   CN1     TM3
35   SP   30000   44        Align label so that 1st significant
36   SA   30000   30000     character is to left
37   TP   A       30000
40   TU   LC35    LC36
41   RA   LC35    CN13
42   RA   LC37    CN
43   IJ   TM3     LC35
44   TU   LC26    LC46
45   TP   CN      TM3
46   TP   30000   A         Check a word of label.
47   EJ   30000   LC51
50   MJ   0       LC17      Word is incorrect; go to LC17
51   RA   LC46    CN13
52   RA   LC47    CN13      Set to check next word
53   IJ   TM3     LC46
54   MJ   0       LC2       All words have been checked and label is OK
55   TU   LC11    LC46
56   MJ   0       LC45
     CA   LC57
```

Tape Codes

```
     IA   VF
 0   02   00002   [4]0000   Read forward
 1   02   00612   [4]0000   Read backward (stop included)
 2   02   00200   [4]0000   Rewind
 3   02   00602   [4]0000   Read forward (stop included)
 4   02   00004   [4]0001   Move forward 1 block

 5   02   00001   50000     Set normal bias
 6   02   00001   60000     Set low bias
 7   02   00001   70000     Set high bias
10   02   00600   00000     Stop tape
11   02   200     50000     Rewind servo 5
12   02   00146   50000     Write 1" bkt, 2.4" bk, 128/in-servo 5
13   02   00746   50000     Write 1" bkt, (stop included)
14   00   20000   04000     Bypass buffer
15   00   0       10000     Increment to reset tape codes
     CA   VF16
```

Excess Three Characters and Words

	IA	CX		
0	54	00000	00000	R
1	67	00000	00000	U
2	01	00000	00000	Δ
3	33	00000	00000	H
4	66	00000	00000	T
5	00	00000	00022	Period
6	74	74747	47474	All Z's
7	01	01010	10101	All spaces
10	01	67503	42651	Δ U N I C O
11	01	01675	03426	Δ Δ U N I C
12	01	01016	75034	Δ Δ Δ U N I
13	01	01010	16750	Δ Δ Δ Δ U N
14	01	01010	10167	Δ Δ Δ Δ Δ U
15	67	50342	65127	U N I C O D } Tape labels
16	30	01525	45132	E Δ P R O G } Tape labels
17	54	24470	12201	R A M Δ . Δ } Tape labels
20	67	50342	65127	U N I C O D } Tape labels
21	30	01265	15454	E Δ C O R R } Tape labels
22	30	26663	45150	E C T I O N } Tape labels
23	00	0	20006	
24	00	BU	CX15	
25	00	BU	CX20	
26	00	0	454	Index for no. corrections
27	00	XX16	6	To set index
30	00	XX5	11	Constants for error print - correction tape
31	00	XX55	5	Constants for error print - program tape
32	00	XX33	15	Constants for error print - parity error
33	00	XX2	3	Constants for heading print - correction tape
34	00	XX22	2	Constants for heading print - program tape
35	00	XX16	4	Constant for delete print
36	77	77777	70701	ɣ ɣ ɣ ɣ 4 Δ
37	77	77777	70601	ɣ ɣ ɣ ɣ 3 Δ
40	0	XX	2	
	CA	CX41		

	IA	CN		
0	00	0	1	Constants
1	00	0	2	
2	00	0	3	
3	00	0	4	
4	00	0	5	
5	00	0	24	
6	77	0	0	
7	00	0	14	To set buffer index 1st group only
10	00	0	15	To set buffer index all other groups
11	00	0	3220	Increment for drum transfer
12	00	24	0	
13	00	1	0	
14	00	2	0	
15	00	7	70000	
16	00	1	1	
17	77	[30000]	00024	
20	00	DT1	DT	
21	00	0	23	
22	00	0	TB	
23	00	0	TB1	
24	00	0	BU170	
25	00	0	BU171	
26	00	CB	CB	
27	00	2	2	
30	00	0	30000	Sequence indicator
31	00	4	4	
32	00	36	0	
33	75	30036	SR6	
34	00	4	0	
35	00	0	10	
36	00	0	7	
37	00	0	6	
40	00	0	FT21	
41	TV	PB4	TM14	
42	00	BJ12	MN32	
43	00	0	[30000]	Total no. of items
44	00	0	[30000]	Number of storage locations
45	00	0	[30000]	Number of in sequence items
46	TP	TB	A	
47	00	0	17	
50	37	77777	77777	
51	00	0	30000	
52	00	0	0	
53	00	0	BU12	
54	00	0	BU	

55	01	51676	60151	XS3 codes for
56	31	01653	05367	OUT OF SEQUENCE
57	30	50263	02277	
60	EW	10000	GR	
61	00	0	11	
62	00	0	51	
	CA	CN63		

```
     IA    LN
 0   MJ    0        LN5
 1   MJ    0        LN7
 2   MJ    0        30000
 3   00    30000    30000
 4   00    30000    30000
 5   TV    RC14     PA1
 6   MJ    0        DR0
 7   TV    RC15     PA1
10   MJ    0        DR
11   TP    LN4      LN3
12   MJ    0        LN2
     CA    LN13

     IA    DR
 0   TV    RC6      MR27
 1   TV    DR       DR15
 2   RJ    MR12     MR12
 3   RJ    MR43     MR43
 4   TP    UC23     LN3
 5   TP    UC4      WS3
 6   TP    LN4      WS
 7   TV    RC10     MR3
10   TV    RC13     MR4
11   TV    RC12     MR5
12   TV    RC11     MR7
13   TV    RC1      MR16
14   IJ    WS3      MR
15   RJ    DR15     30000
16   TP    LN3      A
17   EJ    UC23     SV14
20   EJ    UC24     SV14
21   MJ    0        LN2
     CA    DR22

     IA    MR
 0   LQ    WS       6
 1   QT    UC20     A
 2   RP    20004    MR4
 3   EJ    UC14     30000
 4   EJ    UC13     30000
 5   EJ    UC1      30000
 6   RP    20011    SV6
 7   EJ    UC2      30000
10   TV    RC       MR16
11   TV    RC2      MR3
```

The remaining coding is a modified version of the Line-number Processor used in the translation phase. Instructions which have been changed are noted below. Explanation and annotation for non-changed instructions are to be found in Section III, 3, A - Translation Subroutines.

Exit to store line number and type of error for later print out

```
12    RJ    MR12     MR13
13    RJ    MR12     MR22
14    RJ    MR12     MR22
15    MJ    0        MR24
16    MJ    0        30000
17    SA    LN3      6
20    TP    A        LN3
21    MJ    0        30000
22    RJ    MR21     MR16
23    MJ    0        DR14
24    TV    RC3      MR5
25    TV    RC3      MR7
26    RJ    MR21     MR16
27    RJ    MR27     30000
30    MJ    0        DR14
31    SP    LN3      0
32    SA    UC13     6
33    SA    UC14     6
34    AT    UC14     LN3
35    MJ    0        DR14
36    TV    RC2      MR3
37    TV    RC5      MR4
40    TV    RC7      MR5
41    TV    RC7      MR7
42    MJ    0        MR27
43    RJ    MR43     MR44
44    RJ    MR43     MR46
45    MJ    0        MR60
46    EJ    UC1      MR55
47    LA    A        6
50    TP    UC21     Q
51    QS    A        LN3
52    TP    UC       WS1
53    TP    UC20     WS2
54    MJ    0        DR14
55    LT    10006    WS1
56    TP    UC22     WS2
57    MJ    0        DR14
60    TV    RC4      MR5
61    TV    RC4      MR7
62    EJ    UC1      DR14
63    AT    WS1      WS1
64    TP    WS2      Q
65    QS    WS1      LN3
66    MJ    0        DR14
67    TV    RC10     MR3
70    TV    RC5      MR4
71    TV    RC5      MR5
72    TV    RC5      MR7
73    MJ    0        DR14
      CA    MR74
```

```
     IA   RC
 0   0    0       MR17
 1   0    0       MR21
 2   0    0       MR67
 3   0    0       SV10 —  ]
 4   0    0       SV12 —  }   Used to set error exits so that line number
 5   0    0       SV14 —  ]   and type of error will be stored for later
 6   0    0       MR31        print out
 7   0    0       MR43
10   0    0       DR14
11   0    0       MR10
12   0    0       MR11
13   0    0       MR36
14   0    0       PA3
15   0    0       PA5
     CA   RC16

     IA   UC
 0   0    0       0
 1   0    0       3
 2   0    0       4
 3   0    0       5
 4   0    0       6
 5   0    0       7
 6   0    0       10
 7   0    0       11
10   0    0       12
11   0    0       13
12   0    0       14
13   0    0       22
14   0    0       1
15   0    0       77
16   0    0       17
17   0    0       43
20   0    0       77
21   0    0       7700
22   0    0       7777
23   1    1010    10100
24   1    1012    20101
     CA   UC25

     IA   WS
 0   0    30000   30000
 1   0    30000   30000
 2   0    30000   30000
 3   0    30000   30000
     CA   WS4
```

```
     IA   PA
 0   MJ   0        PA1
 1   MJ   0        0
 2   MJ   0        0
 3   RJ   SV60     SV56        Out to print sentence type
 4   MJ   0        PA2
 5   TP   LN4      CD3
 6   TP   CD       UP3
 7   RJ   UP2      UP
10   MJ   0        PA2
11   RJ   PA2      PA
12   TP   LN4      CD15
13   TP   CD5      UP3
14   RJ   UP2      UP
15   MJ   0        LN11
16   RJ   PA2      PA
17   TP   LN4      CD27
20   TP   CD17     UP3
21   RJ   UP2      UP
22   MJ   0        LN11
23   RJ   PA2      PA
24   TP   LN4      CD42
25   TP   CD31     UP3
26   RJ   UP2      UP
27   MJ   0        LN11
30   RJ   PA2      PA
31   TP   LN4      CD54
32   TP   CD44     UP3
33   RJ   UP2      UP
34   MJ   0        LN11
     CA   PA35

     IA   CD
 0   00   CD1      4
 1   65   30506    63050
 2   26   30010    17777
 3   01   01010    10101
 4   01   01017    77777
 5   40   CD6      11
 6   34   46463    03224
 7   46   01263    32454
10   24   26663    05401
11   34   50016    53050
12   66   30502    63001
13   50   67472    53054
14   01   77777    77717
15   01   01010    10101
16   43   77777    77777
17   40   CD20     11
20   30   72665    42401
21   34   50663    03254
22   24   46012    73432
23   34   66650    13450
```

24	01	65305	06630
25	50	26300	15067
26	47	25305	40117
27	01	01010	10101
30	43	77777	77777
31	40	CD32	12
32	30	72665	42401
33	31	54242	66634
34	51	50244	60127
35	34	32346	66501
36	34	50016	53050
37	66	30502	63001
40	50	67472	53054
41	01	77777	77717
42	01	01010	10101
43	43	77777	77777
44	40	CD45	11
45	34	46463	03224
46	46	01653	05367
47	30	50263	00134
50	50	01653	05066
51	30	50263	00150
52	67	47253	05401
53	17	77777	77777
54	01	01010	10101
55	43	77777	77777
	CA	CD56	

3. TRANSLATION PHASE

3. Translation Phase

a. Translation Subroutines

(1) Abbreviations Used

WL - Translation List (output of translators)

FXX - (where XX is a decimal number) An error number

CB - Combination List

BF - Read in Buffer

CW - Call word

DP - List of Pseudo Operation Dummies

VL - Vary List

VF - Vary File

FC - Flex Codes

IZ - List of Referenced Line Numbers

CL - Constant Pool

VB - }
VD - } Variable or Temporary Regions
TF - }

XS3 - Excess Three Code

SS - Subscripts or Subscripted

Line - Sentence

String Out - Translation or Translation List

S.O. - Translation

(2) General Description

This phase of UNICODE reads the UNICODE Program sentences from tape 5, assigns call words to symbols, and detects errors in the program. The output is a list made up in region WL for each sentence. When the space period (Δ.) symbol is encountered, the list in region WL is written on tape 6 (3) to become the input to the generation phase for the sentence translated. This process continues until END OF TAPE is encountered and translated.

The translation subroutines remain in the core and a translator for the type of sentence encountered is transferred from the drum and referenced by translation control (CT) which acts as a switch. Most translators are transferred into core addresses 4400 to 4400 + N_i where N_i is the length of the translator. The equation and LIST translators are transferred into 4000-4000 + N_i, the IF into 4566 to 4566 + N_i, and the JUMP into 4700 to 4700 + N_i.

Each translator uses as many of the translation subroutines as necessary to make up the translation list (WL). There is no standard format for the translation list with the exception of the heading (WL-WL3). The heading format is described in the format of lists section.

(3) Core During Translation (Not all regions mentioned)

Name	Region	Addresses
1. Indicates 5 or 7 servos 5 servos - TN = 0 7 servos - TN = 0 3 0	TN	20
2. Tape Handler	TH GT	21-420
3. Print Text	UP	421-536
4. Print Error Heading	WA	537-562
5. Build Symbol	WB BS	563-602 603-627
6. Translation Control	CT	714-751
7. Delete Spaces	DS	1001-1007
8. Send Call Word to Translation List	EW	1010-1031
9. Fill Symbol (with 77 codes)	FS	1032-1060
10. Switch List (used by control)	FW	1061-1117
11. Convert Constant to Floating Point	GG	1134-1323
12. Get Next Character	GN	1324-1376
13. Get Next Sentence	GS	1377-1456
14. Assign Constant Call Word	GW	1457-1474
15. Send Referenced Sentence Number to List	IX	1552-1621
16. Check and Standardize Sentence Number	LN	2037-2145
17. Set TN for 5 or 7 servos	OT	2146-2155
18. Send Sentence Call Word to Reference List	RA	2156-2171
19. Check Floating Point Constant	RB	2172-2220
20. Check Fixed Point Constant	RD	2237-2264
21. Check Variable Type Symbol	RH	2265-2315
22. Get Rest of Lower Symbol	RL	2316-2347
23. Decimal to Octal Conversion	RS	2350-2424
24. Get Rest of Superscript Symbol	RU	2425-2443
25. Rewind all Tapes	RW	2444-2465
26. Get Next Symbol	SY	2466-2545
27. Get File From Combination List	TA	2546-2653
28. Send File Back to CB List	TD	2654-2662
29. Add File to CB List	TE	2663-2741
30. Increase 66, 65, 64 Call Word Counter	TK	2742-2761
31. Get C.W. from Dummy Pseudo Op. List	TS	2762-3004
32. Setup Translation Tape	UB	3013-3032
33. Error Routine	UZ	3065-3110
34. Close Vary File	VE	3122-3202
35. Send Translation List to Tape	SS WT	3207-3224
36. Increase 26, 27, 22 Sentence Number Call Word Counter	XJ	3225-3245
37. Read Buffer	BF	3317-3506
38. Translation List	WL VN	3507-3757
39. Operating Area for Translators	OR	4000-7777

(4) Drum During Translation

Name		Region	Addresses
1. Flex Codes		FC	40001-40100
2. Combination List: Dimension List and List of Library Routines, Pseudo operations and Variables		CB	40101-46100
3. Constant Pool		CL	46101-47100
4. Vary File		VF	47101-47245
5. Referenced Line Numbers		IZ	47246-47721
6. Pseudo Op Sentence Call Words of 2nd Sentences		JN	47722-50022
7. Rewind List of Referenced Tape Numbers		WR	50023-50045
8. List of Pseudo Operation Dummies		DP	50046-50177
9. Vary List: Variables which cannot be altered in range of VARY.		VL	50200-50332
10. Error Texts (Translation Subroutines)		FA	50333-50406
		FB	50407-50446
		FD	50447-50512
		FE	50513-50544
		FF	50545-50574
		FG	50575-50623
		FH	50624-50672
		FI	50673-50721
		NO	50722-50751
		PA	50752-51006
		CD	51007-51074
Translators	**Locations Transferred**		
11. COMPUTE	2253_8	CP	51075-53347
12. READ	223_8	RE	53350-53572
13. TYPE	1220_8	TL	53573-55012
14. LIST	3753_8	LM	55013-60765
15. PRINT	250_8	PS	60766-61235
16. IF	2423_8	KP	61236-63660
17. VARY	2444_8	VY	63661-66324
18. RESUME	67_8	RV	66325-66413
19. JUMP	144_8	SJ	66414-66557
20. STOP	30_8	SP	66560-66607
21. END OF TAPE	225_8	EU	66610-67034
22. EXIT	52_8	EZ	67035-67106
23. START	45_8	ST	67107-67153
24. Equation	3663_8	YA	67154-74527
25. Pseudo Operation Heading	1050_8	HE	74530-75600
26. Remainder of drum is service routines			

(5) Error Texts of Translation Subroutines

F1 Sentence - - - - before START, begins with a key word (VARY, LIST, etc.). Rest of this sentence not checked.

F2 Sentence - - - - First symbol, - - - -, does not indicate legal UNICODE sentence. Rest of this sentence not checked.

F3 More than five errors. Rest of this sentence not checked.

F4 More than 25 errors this program. Reread specifications. Program not checked beyond sentence - - - -.

F5 More than 512 lines in this program. 513th sentence number is - - - -.

F6 Number of unsubscripted variables plus functions is greater than 512. 513th symbol is - - - -. Working on sentence - - - -.

F7 More than 12 characters in floating point constant. (Const.)

F8 More than one decimal point in constant. (Const.)

F9 Assumed constant contains a letter. (Const.)

F10 More than six characters in fixed point constant. (Const.)

F11 A decimal point in a fixed point constant. (Const.)

F12 More than six characters in variable type symbol. (sym.)

F13 Variable type symbol contains a point. (sym.)

F14 No sentence number on sentence following - - - - -.

F15 List of variables, Library routines, functions and pseudo operations has become too long. Working on sentence _ _ _ _ _ _ _ _ _.

F16 Too many symbols.

F17 No end of sentence symbol.

F18 Incorrect symbol sequence. - - - -ΔΔ- - - -

F19 Key word, - - - - - , used as variable.

(6) List Formats

(a) Combination List (CB) 40101-46100

This list consists of an item for each variable, library routine, and pseudo operation. The order in which these items appear in the list is as follows:

1) Subscripted Variables, if any (Dimension List)
2) Library Routines
3) Unsubscripted Variables (Floating & Fixed Point) Functions and Pseudo Operations in the order in which they appear in the program.

Address CB = $\begin{matrix} Op \\ 0 \end{matrix}$ $\begin{matrix} u \\ 2(N) \end{matrix}$ $\begin{matrix} v \\ 0 \end{matrix}$ where N = the number of words in the CB list not including CB. N is increased each time an item is added to the CB list.

The items have somewhat different formats so each will be described. Parentheses indicate the extent of the word used.

1. Subscripted Variable

Op	u	v
00	(D)	0 0 0 0 0
(	S	)
00	0 0 0 0 0	(CW)
0(Z)	M	N
0 0	(m_2)	(m_1)
0 0	0	(m_3)

D = The first drum address of this variable in the object program.

S = The XS3 code for the symbol.

CW = The call word (77xxx)

Z = An octal digit with binary representation $Z_1Z_2Z_3$

Z_1 = 1 the variable is defined by an equation before START

Z_2 = 1 the variable is defined by an equation

Z_3 = 1 the variable appears in a READ sentence

M = The modulus (the total number of words of storage needed by this variable in the object program).

N = The number of subscripts.

m_1, m_2, m_3 - multipliers used in the manipulation of subscripts

m_1 = product of rightmost n-1 dimensions

m_2 = product of rightmost n-2 dimensions

m_3 = rightmost dimension

2. Pseudo Operation

Op	u	v
(	S	)
0 0	0 0 0 0 0	(CW)
0(N)	0 0 0 0 0	(F_1)
0(Z)	N 0 0 0 0	(F_2)
0(N)	0 0 0 0 0	(etc.)

function

S = the XS3 code for the symbol

CW = the call word (4XXYY where YY = the number of operands; XX = pseudo operation number.)

F_1, F_2, etc. = Operand formats which read from right to left with the following codes:

1 - Floating point operand

2 - Fixed point operand

3 - Function

4 - Subscripted variables

For example, the function F (X, Y(I)) would have format: 413.

The subscripted variable U(I, J, K, L) would have format: 4

The variable R would have format: 1.

The constant 3.49 would have format: 1.

The constant 75 would have format: 2. (a constant with a decimal point is floating point, one without is fixed point.)

N = number of commas for a function.

N = number of subscripts for a subscripted variable.

N = 0 for other operands.

Z is the same as Z described under the subscripted variable.

3. Function

Op	u	v
(	S	)
0 0	0 0 0 0 0	(CW)
0 Z	N 0 0 0 0	(F)

S = XS3 code of symbol

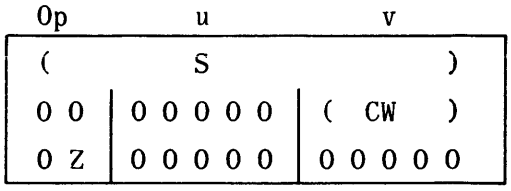
CW = Call word (66XXX)

F = Format as described under Pseudo Operation item.

Z = Same as Z described under subscripted variable.

N = The number of commas.

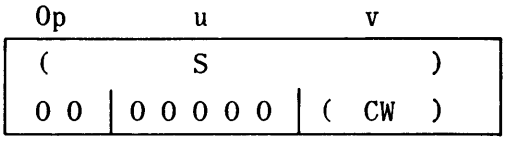
4. Floating Point or Fixed Point Variable



S = XS3 code of symbol

CW = Call word (65XXX for floating point, 64XXX for fixed)
Z is described under subscripted variable item.

5. Library Routine



S = XS3 code of symbol

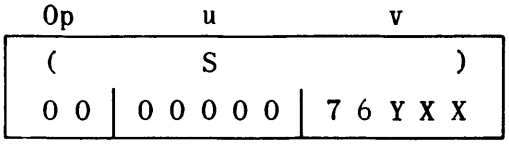
CW = Call word (5XXXY where Y = number of operands,
XXX = routine number.)

(b) Pseudo Operation Dummy List (DP)

50046-50177

This list is made up by the pseudo operation heading translator. The entire list is searched for a symbol even if the list is not full since it is short (132_8). The list is cleared before adding the first symbol. The items in the list are as follows:

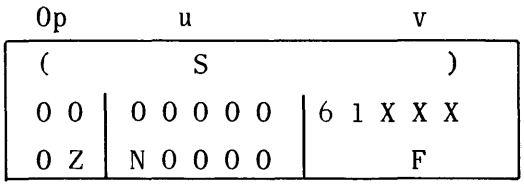
1. Subscripted Variable



S = XS3 code of dummy symbol

76YXX = dummy call word (Y = number of subscripts,
XX = variable number.)

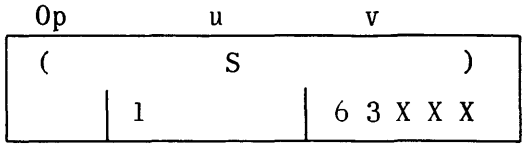
2. Function



S, Z, N, F = same as described under Pseudo operation in combination list.

6 1 X X X = Call word.

3. Floating Point or Fixed Point Variable



S = XS3 of dummy symbol

63XXX = Call word (the same for both floating or fixed point)

(c) Translation List (WL)

WL = Number of words (includes word WL) in translation list $\leq 250_8$.

WL1 = Standardized sentence number. This is of the form:

$$X_1 X_2 X_3 \cdot X_4 X_5$$

where X_i is the XS3 code for a decimal digit or XS3 for a space (01). The XS3 for the point (22) always appears. Spaces will not separate digits or a digit and the point.

WL2 = the sentence type (READ, LIST, DIMENS, etc.) in XS3 code. Only the first six codes appear for words of more than six characters and for words of less than six characters. WL2 is filled on the right with 77 codes. There are two special cases, the equation and the pseudo operation heading. For the equation:

WL2 = EQUATN (XS3) - Before START

WL2 = EQUATI (XS3) - After START

For the pseudo operation heading:

WL2 = The XS3 codes for the symbol which represents the pseudo operation.

WL3 = Sentence call word in v address.

2 2 X X X - Sentence within pseudo operation

2 3 0 0 0 - END OF TAPE sentence

2 4 X X X - Equation for S.S. variable (before START)

2 5 X X X - Equation for other variables (before START)

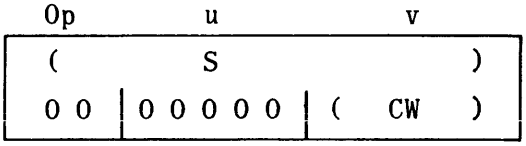
2 6 X X X - VARY sentence (not in pseudo op.)

2 7 X X X - Sentence (other than VARY) START or after.

where X is an octal digit.

The rest of the translation list is not standard but depends on the particular translator.

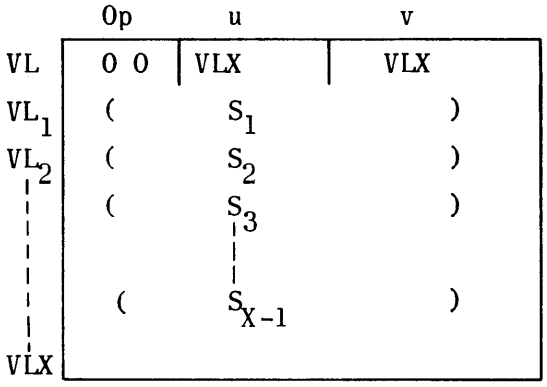
(d) Referenced Sentence Numbers (IZ) 47246-47721



S = Standardized Sentence Number

CW = Sentence call word (26XXX, 27XXX or 22XXX)

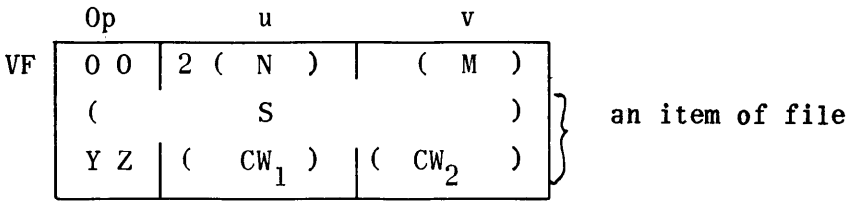
(e) VARY (Variable) List (VL) 50200-50332



VLX = Next available address in list

S_1, S_2, etc. = XS3 code for symbols which cannot be altered within range of a VARY sentence.

(f) VARY File (VF) 47101-47245



M = Number of words down to first 22XXX VARY sentence, if any.

N = Number of words in file not including VF

S = Standardized sentence number of last sentence in range of VARY.

CW_1 = Call word of sentence jumped to or resumed at end of VARY loop (27XXX, 26XXX or 22XXX)

CW_2 = Call word of the VARY sentence (26XXX or 22XXX)

Z (4 bits) = Count of number of "with" words.

Y (2 bits) = Y_1, Y_2 where:

Y_1 = 1 Indicates closed out or completed item.

Y_1 = 0 Item not closed out or completed.

Y_2 = 0 Normal jump

Y_2 = 1 Resume jump (either stated or implied)

(g) Rewind List of Referenced Tape Numbers (WR)

50023-50045

	Op	u	v
WR	0 0	2(N)	0 (N)
WR1	0 0	CW_1	0 0 0 0 0
etc.	0 0	CW_2	0 0 0 0 0

N = Number of words in list not including WR, CW_1, CW_2, etc. = Call words of tape numbers (67XXX or 64XXX)

This list is used by the STOP object program so the machine operator may rewind input and output tapes from the console.

(h) List of Second Sentences of Pseudo Operations (JN) 47722-50022

	Op	u	v
JN	0 0	2 (N)	0 0 0 0 0
	0 0	(CW_1)	0 0 0 0 0
	0 0	(CW_2)	0 0 0 0 0

N = Number of words in list not including JN.

CW_1, CW_2, etc. = Call words of Sentence numbers of second sentences of pseudo operations.

This list is used in generation to check for illegal jumps from one pseudo operation to another.

(i) Constant Pool for Object Program (CL) 46101-47100

This is a list of the constants both floating point and fixed point, one constant per word.

address 10 =00 2 (N) 0 (N)

N = Number of constants in list CL.

The constants are assigned call words according to their position in the list, i.e., the constant in CL has call word 67000, in CL1 67001 and so forth.

(j) Key Words of UNICODE (Region KB - 1664-1722)

The following words cannot be used as variables in UNICODE. (See region KB of coding.)

DIMENS	JUMP
START	STOP
VARY	END
COMPUT	EXIT
READ	POW
LIST	NOT
TYPE	TAPE
PRINT	WITH
IF	THEN
RESUME	AND

(7) Descriptions of Translation Subroutines

(a) Translation Control (CT)

This routine gets the next sentence (GS), gets the first symbol of the sentence (SY), sets up the translation list (WL) heading, transfers the proper translator to the core (depending on the first symbol of the sentence), and jumps to this translator.

After the translator has completed building list WL and found the space period (Δ.) of the sentence it writes WL on tape by referencing routine SS. Then it jumps back to translation control (CT).

Sometimes a translator will find a sentence so full of errors that it is a waste of time to continue checking it. Then the translator will jump to control (CT) before finding the Δ. symbol. Translation control will pick up symbols until a Δ. appears in SY2.

This routine is not written as a subroutine and is referenced by:

```
MJ   0   CT
```

(b) Get Next Sentence (GS)

This routine sets the Get Next Character (GN) routine to pick up the first character of the next sentence, standardizes the sentence number, reads tape if necessary, and sets up the first two words of the translation heading. It is referenced by the instruction.

```
RJ     GS     GS1
```

If the sentence number word is all Z's it sets up the entire translation list for END Δ OF Δ TAPE , and goes directly to the end of tape translator. If the sentence number word is all spaces it checks the entire sentence and if all spaces proceed to the next sentence. If the entire sentence is not all spaces it prints NO SENTENCE NUMBER ON SENTENCE FOLLOWING - - - - -. (Error F14).

It also clears the indicator of previous print-out in routine EW.

(c) Get Next Character (GN)

This routine picks up the next character from the raw input, read into BF, of the corrected problem tape and places the character in the rightmost six bits of A and GN4. It is referenced by the return jump.

```
RJ     GN     GN1
```

The first time the routine is used, the address where the character is located must be put in both the u and v addresses of GN2, and a shift count in GN3. To explain the shift count, consider the characters of a word to be numbered from 1 to 6 left to right. To obtain the first character at the address in GN2, set the shift equal to zero and perform RJ GN0 GN1. But to obtain the second character, first left shift word in address at GN2 by 6, and set shift count equal to 6, then perform the return jump. The following table gives all cases:

Character Number	No. of Places to shift word at address put in GN2	Set shift count in GN3 equal to:
1	0	0
2	6	6
3	14_8	14_8
4	22_8	22_8
5	30_8	30_8
6	36_8	36_8

References after the first are by the return jump only since the routine automatically sets itself for successive characters.

(d) Get Next Symbol (SY)

To pick up the next symbol from the raw data, the instruction is used.

```
RJ      SY      SY1
```

In picking up the symbol, the routine ignores spaces until a significant character is encountered, then characters are assembled into a symbol until a symbol separator character is encountered. The following characters are separators:

1. Δ
2. ,
3. ;
4. (
5.)
6. +
7. -
8. *
9. Any lower case character when building a superscript symbol.
10. Any superscript character when building a lower case symbol.
11. /
12. superscript -
13. superscript /
14. =
15. |
16. >
17. <

All of these characters, except 1, 9, and 10 are symbols by themselves. Following is a list of all symbols:

1. Δ.
2. ,
3. ;
4. (
5.)
6. +
7. -
8. *

9.	/	14.	>
10.	superscript -	15.	<
11.	superscript /	16.	Variable Type Symbol
12.	=	17.	Constant
13.	|	18.	Superscript Constant

A symbol of type 16, 17, or 18 may contain any number of characters, but the routine will save only the first 18 of these in SY2 - SY4. All symbols are stored with characters packed to the left, and unused spaces in SY2 - SY4 are filled with 77 codes. For example, if the symbol were XYZ, SY2 - SY4 would be

	Op	u	v
SY2	X	Y Z 7	77777
SY3	77	77 77 7	77777
SY4	77	77 77 7	77777

SY2 is sent to A before exiting. The rest of the outputs are:

SY5 = Number of characters

SY6 = Number of decimal points

SY7 = Indicator; 1st character a letter

SY10 = Indicator; 1st character I, J, K, L or M

SY11 = Indicator; 1st character decimal point or digit

SY12 = Indicator; symbol contains a letter

SY13 = Indicator; symbol is superscript

SY14 = Indicator; space was separator

where the indicator is 40 in operation portion of word if condition on right of semicolon is satisfied.

This routine uses the Get Next Character, Build Symbol, Fill Symbol, Get Rest of Lower Symbol, Get Rest of Upper Symbol, and Delete Spaces routines as subroutines.

(e) Get Rest of Lower Symbol (RL)

This routine builds the symbol in SY2-SY4 until a separator for a lower case symbol is encountered. Reference:

```
RJ  RL  RL1
```

(f) Get Rest of Superscript Symbol (RU)

This routine builds the symbol in SY2-SY4 until a separator for a superscript symbol is encountered. Reference:

```
RJ   RU   RU1
```

(g) Build Symbol (BS)

This routine adds the last character picked up by the Get Next Character (GN) routine to the symbol in SY2-SY4. Reference:

```
RJ  BS  BS1
```

(h) Fill Symbol (FS)

This routine fills the symbol in SY2-SY4 on the right with 77 codes. Reference:

```
RJ    FS    FS1
```

(i) Delete Spaces (DS)

This routine picks up characters until a character other than a space is encountered. It sets the indicator in SY14 if a space is encountered so the Get Next Symbol routine (SY) will recognize the $\Delta.$ combination. Reference:

```
RJ    DS     DS1
```

(j) Send File Back to Combination List (TD)

This routine returns the file of a variable to the CB list when it has been picked out of the CB list by the Get File from CB List (TA) routine. The file is located at TA2 up to TA31. The exact number of words in the file is in the u address of TA47. The routine is necessary since a file may be altered after adding it to the list.

(k) Add file to CB List (TE)

The file of a variable is built in TF up to TF27 with the exact number of words in both addresses of TF. When the file is complete this routine is referenced and the file is added to the CB list.

(l) Get File from CB List (TA)

The file of the variable in SY2 is picked up from the CB list and placed in TA2-TA31 by this routine if the file is in the CB list. If RJ TA TA1 is at address Y, then the exit is to Y + 1 if the file is not in the list. If the file is in the list, the exit is to Y + 2.

(m) Get Call Word from Pseudo Operation Dummy List (TS)

The pseudo operation heading translator makes up a list of dummy variables in region DP. The list consists of the XS3 and the dummy call word of each variable which appears in the heading. This routine searches the list for the variable in SY2 and if it is in the list the output is as follows:

TS2 = XS3 of symbol

TS3 = Call word in v address

If RJ TS TS1 is at address Y, then the exit is as follows:

Y + 1, if variable is not in list.

Y + 2, if variable is in list.

(n) Send Call Word to Translation List (EW)

This routine transfers the contents of EW2 to the next available location in the translation list (region WL). The word to go into the translation list is sent to EW2; then the instruction RJ EW EW1 is executed.

(o) Increase 66XXX, 65XXX, 64XXX Call Word Counter (TK)

This routine is referenced before assigning a variable a 66, 65 or 64 type call word. The counter is in VB1 and is less than or equal to 777_8. The counter is left in A and VB1 so that 66000, 65000, or 64000 may be added to it to obtain the proper call word. Reference the routine with RJ TK TK1.

(p) Increase Sentence Call Word Counter for 26XXX, 27XXX, or 22XXX Type Call Words (XJ)

This routine is exactly like TK (above) except that the counter is in VB4 and A. Reference is by RJ XJ XJ1.

(q) Print Error Heading (WA)

This routine prints the following on the typewriter:

SENTENCE ΔΔ X ΔΔΔ(Y)ΔΔ

where: X = sentence number from WL1 (1-6 characters),

Y = type of sentence (IF, COMPUTE, etc.). First six characters from WL2 if more than six, otherwise all characters from WL2.

Reference the routine as follows:

RJ WA WA1 If it is desired that the error routine (UZ) be referenced.

RJ WA WA2 If the error routine (UZ) is not to be referenced.

(r) Error Routine (UZ)

This routine should be referenced before each error print out. It will add up the number of errors and when a maximum of 25 has been exceeded, it will print F4, rewind all tapes, and stop the computer.

After five errors in a single sentence F3 is printed, the rest of the sentence is skipped and error checking on following sentences begins.

The Error Routine is referenced by the following instruction:

RJ UZ UZ1

The sum of the errors per program is kept in UZ2, the number of errors per sentence in UZ3.

(s) Check Floating Point Constant (RB)

This routine uses the indicators in SY5-SY13 to check the constant for errors F8, F9 and F7.

(t) Check Fixed Point Constant (RD)

This routine uses the indicators in SY5-SY13 to check the constant for errors F9, F10, and F11.

(u) Print Text (UP)

Send the parameter to UP3 and perform the instruction:

```
RJ    UP2    UP
```

The parameter is of the form:

Op	u	v
OP	X	N

where X = address of the first word of six XS3 characters and N is the number of words in octal to be printed. If OP = 00, the following will precede printing:

1) Set to print 80_{10} characters this line.
2) Set to print spaces.
3) Print 4 carriage returns.

If OP = 40, printing will continue where left off.

When 80_{10} characters have been printed and N is not exhausted, Print Text will suppress spaces until a character other than space is encountered. If this character is a 77 code, this routine will pick up characters until a code other than 77 is encountered. If N is not exhausted by 77 codes or suppressed spaces, it will:

1) Print two carriage returns.
2) Print 19_{10} spaces.
3) Set to print 61_{10} more characters this line.
4) Print the character.

If one wishes more indentation than 19 spaces, place a 77 code before the desired number of spaces to be printed in excess of 19.

If N = 0 and OP = 00, four carriage returns will be printed before the exit.

X can not be A or Q. If X is an illegal address, an SCC fault will occur.

The character count is in UW21. It is counted down.

The routine assumes a manual jump instruction is at address 0.

The routine leaves the Flexowriter in the upper case position.

The XS3 character > is printed "gtr" and < is printed "lsr" but the Print Text routine only counts each as one character, when actually three are printed. Hence, if too many of these symbols appear in one line of printing, the character count of 80 will not be reached before printing runs off the right side of the paper. The alternative is to make the user of Print Text count three characters instead of one for > and < .

(v) Put Call Word in List of Referenced Sentence Numbers (RA)

This routine searches the list (IZ) for the sentence number in WL1 and if it is in the list, the call word of the sentence which is in WL3 is put in the list following the sentence number.

The sentence numbers are put in the list when encountered by the VARY, JUMP, IF, READ and RESUME translators.

(w) Translation Set-Up Subroutines (OT and UB)

When seven Uniservos are to be used for UNICODE, the MJ1 switch should be set. When only five Uniservos are used, this switch should not be set.

Routine OT puts 0 3 0 in TN if the MJ1 switch is set. Otherwise TN is cleared. Parameters for the generalized tape handler are prepared initially for the five-Uniservo layout. If the same Uniservo is used by a routine, whether there are five or seven Uniservos available, no alteration is made on the parameter. If a different Uniservo is used in these two situations, addition of TN to the parameter will ensure selection of the proper Uniservo. This is true because when a different Uniservo is used with seven Uniservos available, there is a constant difference of 3 between the logical numbers of the two Uniservos involved.

Routine UB puts the proper parameter word in WT, the Tape Write routine, depending upon the assigned value to TN. Also this routine writes the title block of the string-out tape: 20 words of Z's; the title, String-Outs; and the balance of the block in Z's.

RJ OT OT1 and RJ UB UB1 are the instructions needed for these set-up subroutines.

(x) Write Translation List on Tape (WT or SS)

Instruction RJ WT WT1 will cause a completed string-out to be written on tape if no errors have been recorded in UZ2 previously. A quick exit with no tape action results when the Error routine has been referenced.

The routine divides the number of lines by 170_8 to determine the number of blocks to be written and sets up the proper parameter for use in referencing the generalized tape handler.

(y) Put Referenced Sentence Number In List IZ (IX)

Region IZ, the list of line numbers and call words, may expand to 454_8 addresses. Each line number in IZ is followed either by its call word in the v position of the next address or by zero. In fixed location 11 is a counter which keeps track of the size of IZ. At the beginning of string-out 00 20000 00000 is put into 11. Each new line number added to the list increases 11 by 00 00002 00002.

To use routine IX, a referenced line number is first put into proper form by sending it to the Line Number routine. The line number in proper form is then put into A and the routine is set into operation by RJ IX IX1. First list IZ is checked to see if the line number is already in the list. If it is there, no further action is taken. If it is not in the list, it is added at the end and the number in the subsequent address is cleared to zero.

As 11 is increased with each new number, it is checked to see if it exceeds 00 20454 00454. If this happens, the following error print-out occurs: - REFERENCED LINES EXCEED 150. At the same time a return jump is made to the UZ Error routine and all subsequent referenced line numbers, if not found in the list, are used merely to increase a count of excess referenced line referrals contained in IX47. A later subroutine EE of the End of Tape instruction prints the number in IX47 if it is other than zero.

(z) Excess-Three Decimal to Octal Routine (RS)

Input to this routine is assumed to be one line of six characters packed starting at the left. Numbers are positive integers varying from 0 to 999999. Decimal fractions are not converted. A non-digit character in the leftmost

position of the input line causes an early exit from the routine with the output line cleared to zero.

RS4 is the input line. RS3 is the output line. The instruction of entry to the routine is RJ RS2 RS.

One reco region, RS, contains all instructions, constants, and working-storage locations comprising the routine. This region takes up 55_8 addresses.

Each character of the line of input is masked out and converted from excess three. Before storing in a common line with other digits, it is checked to see if it has a value below 0 or above 9. Such a value causes a termination of further assembly of characters and, if the count of digits is not zero, the immediate beginning of octal conversion.

Before starting conversion, the number of digits is subtracted from six and this difference multiplied by six to determine an initial shift of the line of digits.

To obtain the index for the loop used in conversion, one is subtracted from the number of digits.

The conversion-to-octal loop consists of a left Q-shift of the digit line six places, an SP u 2 command, an SA u 1, and a Q-controlled-add masking operation setting up the u of the preceding steps for the next loop.

(aa) Excess-Three Decimal to Floating Point (GG)

The two lines of excess-three input are GG4 and GG5. Numbers are packed, starting from the leftmost position. Positive numbers from .00000000001 to 999999999999 are permissible input to the routine. Negative numbers may be converted to floating point by complementing the floating point output. Each number starts in GG4. The output goes to GG3. The instruction to use the routine is RJ GG2 GG.

Three reco regions contain the routine. GG takes up 170_8 locations; CF occupies 17_8 places; and CC, temporary storage, is in 27_8 addresses. Thus 236_8 addresses are needed for its operation.

Each line of input contains six excess-three characters. A 6-bit binary number represents each character. If a character number is less than three or greater than 12, it acts as an end-of-digits notice to the routine. The one exception to this notice is the number representing a first period (or decimal point) occurring in the input lines.

All characters following an end-of-digits character are ignored by the routine. If no significant figures have preceded an end-of-digits number,

the output line is cleared to zero. The routine converts to floating point only the figures preceding an end-of-digits representation.

Zeros in front of a decimal point when not preceded by a figure greater than zero are not counted as significant figures. Significant figures, whether preceding or following a decimal point, are stored after subtracting three in a set of consecutive registers and converted in a multipying loop to an octal number stored in two address locations.

This number is divided by 10 to a power equal to the number of original decimal figures following the decimal point. Before this division the dividend is shifted left by the number of binary places needed to give a quotient of 10 significant octal figures. Special provision is made in the case of 11 figures after the decimal point when the divisor 10^{11} overflows a 36-bit register. If no decimal point was among the original decimal figures, or if no figures followed the decimal point, this division is bypassed.

In either case, the scale factor command is used in the routine to determine the size of the result and location of the first significant bit. The number is rounded off to 9 octal places and shifted till it occupies the positions $i_{26}.....i_0$ in the output. It becomes the mantissa of the floating point number with i_{26} as its first significant bit.

The k of the scale factor command is used to compute the biased characteristic (b) as follows: (numbers are in octal)

If k = 45, both the biased characteristic and mantissa equal zero.

If k > 45, b = k - 110 + 243 + (CC 6) - (CC 5).

If k < 45, b = k + 243 + (CC 6) - (CC 5).

(CC 6) = 1 or 0 depending upon whether a round off caused or didn't cause a carry-over from the first significant bit. (CC 5) = shift of dividend before division.

(ab) Assign Constant Call Word (GW)

To use this routine, put the input constant in A and give instruction RJ GW GW1. Call-word output goes to A_u and Q_v.

If n of 0 2, n n in absolute address 10 exceeds 1000_8, the Error routine UZ is referenced and the alarm print-out occurs: TOO MANY CONSTANTS. The computer does not stop. A count is kept in FX13 of the number of excess referrals to the routine beyond the maximum. Later an End of Tape subroutine EA prints out this number if it is other than zero. Call-word number 67777 is assigned to all constants beyond the maximum.

The counter for number of constants, 10, is set at 00 20000 00000 at the start of translation.

(ac) Tape Handlers (TH or GT)

There are two Tape Handlers, one for the 1105 and one for the 1103A. They are essentially the same. The 1105 version operates in the Bypass mode only but on either tape control unit as follows:

```
MJ2     OFF     TCU1
MJ2     ON      TCU2
```

The 1103A version ignores MJ2.

When the routine is entered, a parameter must be present in TH3. This parameter is not destroyed so that consecutive references to accomplish identical operations do not require that a parameter be sent to the routine for each reference. However, the general calling sequence is

```
TP      PM      TH3
RJ      TH2     TH
```

where PM is the address of the parameter.

The parameter has the following form:

R M T N N S S V V V V V

each letter representing one octal digit.

R specifies the operation desired:

- R = 1 Rewind
- R = 2 Rewind with Interlock
- R = 3 Move forward
- R = 4 Move backward
- R = 5 Read forward
- R = 6 Read backward
- R = 7 Write
- R = 0 Write (special case, explained later)

M specifies blockette spacing (for write only)

- M = 1 0.0 inches
- M = 2 0.1 inches
- M = 4 1.2 inches

T specifies block spacing and density (for write only)

T = 0 1.2 inches - lower density

T = 1 1.2 inches - higher density

T = 2 2.4 inches - lower density

T = 3 2.4 inches - higher density

NN = number of blocks to be written.

MTNN = number of blocks to be read or moved.

SS = Uniservo number

VVVVV = High speed storage address for read and write. It specifies the first location to or from which the first word on tape is read or written. Thus, successive words on tape correspond to ascending storage addresses for Read Forward and Write operations. In Read Backward operations successive words on tape are read into descending storage locations.

If the number of blocks to be moved, read, or written is zero, the Tape Handler will not cause a move, read or write, except in the case of R = 0 when one block is written. Hence, it is suggested that R = 0 never be used in a parameter. There is a print-out and stop after six reads of a block in which a parity or sprocket error occurs.

(ad) Line Number Processor (LN)

A Line or Sentence number may be any positive, rational number which requires at most three decimal digits to the left of the decimal point and at most two to the right. Every Sentence of a UNICODE Program must be numbered and each Sentence number must be greater than the Sentence number corresponding to the preceding Sentence.

According to the above definition, a Line or Sentence number need consist of at most six characters (five digits and a decimal point). Hence, if six characters are allowed for each number and no further restrictions are made, representation of those Line numbers which have less than five digits will not be unique. To clarify this statement, consider the following possible representations of Line Number 1.3:

```
Δ Δ Δ 1 . 3
( Δ 1 . 3 )
1 . 3 Δ Δ Δ
0 0 1 . 3 0
( 1 . 3 0 )
Δ Δ 1 . 3 Δ
```

The purpose of this routine is to process all input Line numbers so that each processed number can be represented in one and only one way. This is done in such a way that equality and threshold jumps can be used to make direct comparisons of the numbers. The excess-three representation of decimal digits makes this possible. Because six Unityper codes occupy one 36-bit word, these comparisons can be single precision comparisons.

Definition of Legal Input to Line Number Processor

Input consists of the six Unityper characters which represent the Line number. Let these characters be numbered 1 through 6 from left to right. To be legal, an input Line number must obey the following rules:

1. No Unityper codes other than those for ( ) Δ . ∤ 0 1 2 3 4 5 6 7 8 9 may appear, (here ∤ denotes the "Nonexisting character" whose code is 77).
2. At most one "." code may appear (none is needed if the Line number is an integer).
3. At least one non-zero digit code must appear.
4. An input Line number can not have more than three digits to the left of the "." or more than two digits to the right of the "." regardless of whether or not these digits actually contribute something to the Line number (see examples below).
5. The four characters ( ) Δ . are all considered equivalent and, hence, are all treated in the same way. They are ignored (or discarded) except that none of them can appear between the first and last digit of the Line number.

Description of Output Line Numbers

The output Line numbers also consist of six Unityper characters. The output or processed Line number will, of course, satisfy the five conditions given above. In addition, it will satisfy the following requirements:

1. Character 4 will always be the decimal point ".".
2. In the integral part, only non-zero digits and zero digits which follow at least one non-zero digit will appear.
3. In the fractional part only non-zero digits and zero digits which are followed by at least one non-zero digit will appear.
4. The integral digits appear in character positions 1, 2, 3 and the fractional digits in character positions 5, 6.

5. All character positions which are not filled with digits or a "." are filled with space (Δ) codes.

Hence, the effect of this routine operating on a Line number is to shift the Line number until the decimal point is in position 4 (insert a decimal point if there is none), delete all unnecessary digits, and put Δ codes in all unused positions.

Examples:

Legal Input Line numbers	Corresponding Output Line numbers
123.45	123.45
023.40	Δ23.4Δ
003.00	ΔΔ3.ΔΔ
(Δ1ΔΔ)	ΔΔ1.ΔΔ
000.01	ΔΔΔ.01
999.99	999.99

Illegal Input Line Numbers

A12.3Δ	0127.9	1.23.Δ
0000.2	.004ΔΔ	93Δ2.6
4321ΔΔ	.12000	ΔΔΔ0ΔΔ

Two classes of Line numbers are taken care of by the Line number processor; those occurring within a UNICODE instruction and those appearing in the left-hand margin in the first six character positions of a Sentence.

For both classes of numbers, the input number is put into LN4 and the out-out number is obtained from LN3.

Operation of the routine for those numbers appearing within a UNICODE instruction is called Case I and is done by instruction RJ LN2 LN.

Case II operation of the routine takes care of Line numbers in the left-hand margin. Instruction RJ LN2 LN1 is used for this type of number.

In either case, if there is an error print-out, the output cell, LN3, will contain the input Line number and the routine sets the error bit by instruction RJ UZ UZ1 before the print-out.

Error print-outs, Case I: In this case the Print Error Heading routine,

WA, is used to print the number and type of the sentence in which the illegal Line number occurred. This is then followed by one of the four possible alarm comments and the input Line number which gave rise to the alarm.

Examples of the four different alarm print-outs for Case I follow:

```
SENTENCE 123.45  (VARY)  ILLEGAL CHARACTER IN SENTENCE NUMBER (ABCDEF)
SENTENCE 123.45  (VARY)  EXTRA INTEGRAL DIGITS IN SENTENCE NUMBER (1234.0)
SENTENCE 123.45  (VARY)  EXTRA FRACTIONAL DIGITS IN SENTENCE NUMBER (12.345)
SENTENCE 123.45  (VARY)  ILLEGAL SEQUENCE IN SENTENCE NUMBER (1.2.3.)
```

Note that the length of the longest comment is 74 characters plus the number required by the sentence name.

Error print-outs, Case II: Because the Line numbers in the left-hand margin are examined before the type of sentence has been determined, it is not possible to use the WA routine to give the type of sentence. The illegal sentence numbers are given twice as shown in the four examples below:

```
SENTENCE ABCDEF  ILLEGAL CHARACTER IN SENTENCE NUMBER (ABCDEF)
SENTENCE 1234.0  EXTRA INTEGRAL DIGITS IN SENTENCE NUMBER (1234.0)
SENTENCE 12.345  EXTRA FRACTIONAL DIGITS IN SENTENCE NUMBER (12.345)
SENTENCE 1.2.3.  ILLEGAL SEQUENCE IN SENTENCE NUMBER (1.2.3.)
```

(8) Region Assignments of Translation Subroutines

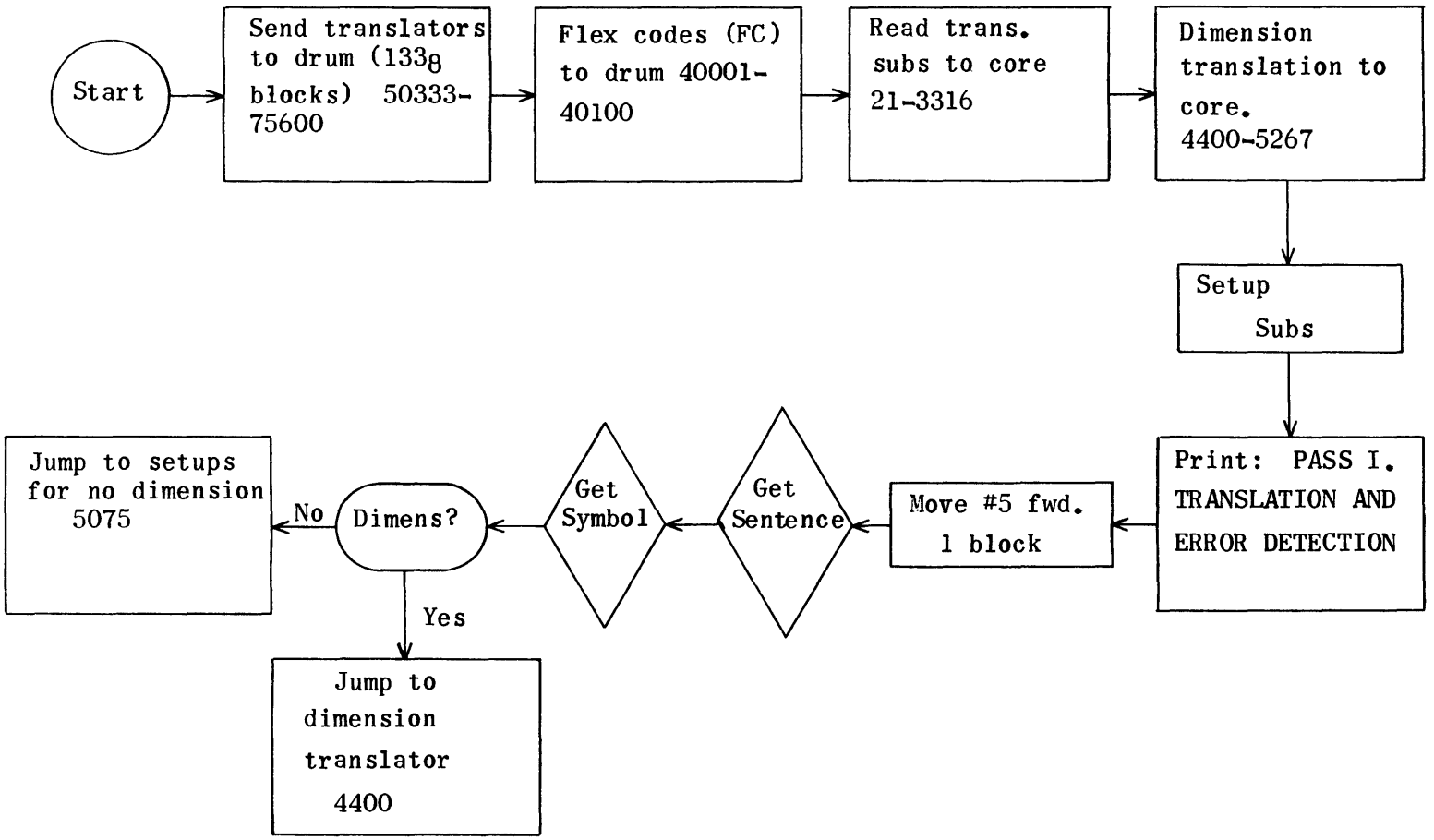
```
RE   TN20                Indicates 5 or 7 servos
RE   GT21      }
RE   TH21      }         Tape Handler
RE   UP421     )
RE   UQ443     )
RE   US453     )         Print Text
RE   UW513     )
RE   WA537     }         Print Error Heading
RE   WB563     }
RE   BS603               Build Symbol
RE   CC630     )
RE   CF657     )         Constants
RE   CJ676     )
RE   CT714               Translation Control
RE   DG752               Part of Get Symbol
RE   DR757
RE   DS1001              Delete Spaces
RE   EW1010              Send Call Word to Translation list
RE   FS1032              Fill Symbol
RE   FW1061              Switch List
RE   FX1120              Part of Constant C.W. routine
RE   GG1134              XS3 constant to floating point
RE   GN1324              Get Next Character
RE   GR1377    )
RE   GS1406    )         Get Next Sentence
RE   GU1444    )
RE   GW1457              Assign constant C.W
RE   GX1475              Part of Get Sentence
RE   HA1500    }         Part of Trans. Control
RE   HB1541    }
RE   IX1552              Referenced line no. to list
RE   KA1622    )
RE   KB1664    )         Constants
RE   KK1723    )
RE   LL2030              Part of Get Symbol
RE   LN2037    }         Line no. Processor
RE   MR2052    }
RE   OT2146              Set for 5 or 7 servos
RE   RA2156              Sentence C.W. to Reference List
RE   RB2172              Check Floating Point Constant
RE   RC2221              Part of line no. processor
RE   RD2237              Check Fixed Point Constant
RE   RH2265              Check Variable Type Symbol
RE   RL2316              Part of Get Symbol
RE   RS2350              Decimal to Octal Conversion
RE   RU2425              Part of Get Symbol
RE   RW2444              Rewind All Tapes
RE   SY2466    }         Get next symbol
RE   SZ2530    }
```

```
RE   TA2546   }
RE   TB2637   }   Get File from CB list
RE   TD2654       Send File back to CB list
RE   TE2663   ]
RE   TF2702   }   Add File to CB list
RE   TG2732   ]
RE   TK2742       Increase 66, 65, 64 C.W. counter
RE   TS2762       Get C.W. from dummy pseudo op. List
RE   UA3005       Part of Trans. Control
RE   UB3013       Setup Translation Tape
RE   UC3033       Part of Line no. processor
RE   UD3060       Part of Get Symbol
RE   UM3065       Part of Get Symbol
RE   UZ3067       Error Routine
RE   VB3111   }   Variables
RE   VD3116   }
RE   VE3122   ]
RE   VG3142   }   Close Vary File
RE   VH3171   }
RE   VI3176   ]
RE   WS3203       Part of line no. processor
RE   WT3207   }   Send Translation List to Tape
RE   SS3207   }
RE   XJ3225       Increase 26, 27, 22 C.W. counter
RE   XM3246       Part of Get Symbol
RE   Z03263       Part of Trans. Control
RE   ZS3267       Part of constant call word routine
RE   BF3317       Read buffer
RE   VN3507   }   Translation List
RE   WL3507   }
RE   FC40001      Flex Codes
RE   CB40101      Combination List
RE   CL46101      Constant Pool
RE   VF47101      Vary File
RE   IZ47246      List of Referenced Line Numbers
RE   JN47722      2nd lines of Pseudo operations
RE   WR50023      Rewind List of Referenced tape nos.
RE   DP50046      List of Pseudo op. Dummies
RE   VL50200      Vary List
RE   FA50333   ]
RE   FB50407   }
RE   FD50447   }
RE   FE50513   }  Error Texts
RE   FF50545   }
RE   FG50575   }
RE   FH50624   }
RE   FI50673   }
RE   NO50722   }
RE   PA50752   }
RE   CD51007   ]
```

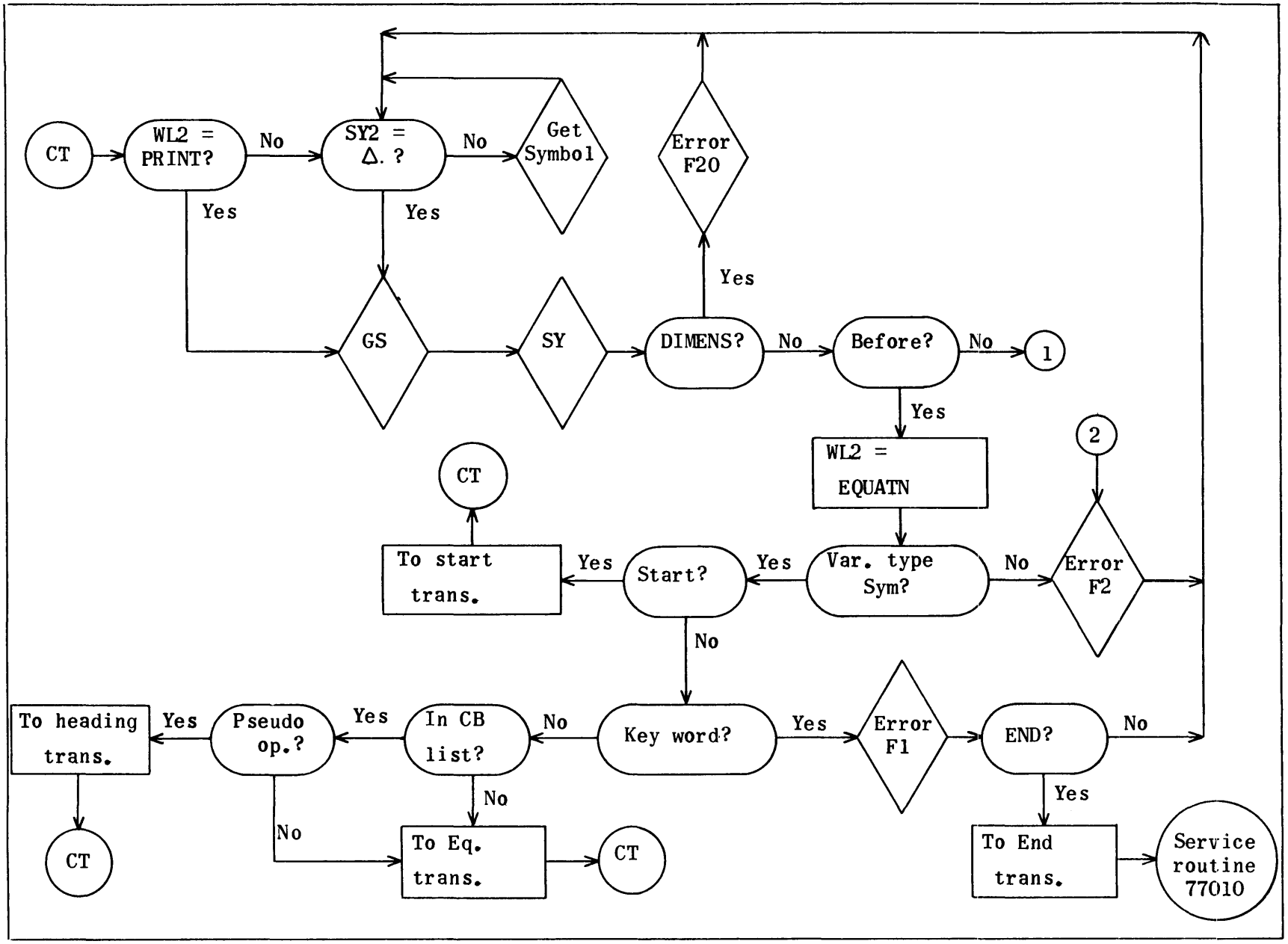
RE	0A32253	Compute	3 (n) Where n = number of words to transfer from drum.
RE	0C30223	Read	
RE	0D31220	Type	
RE	0E33753	List	
RE	0F30250	Print	
RE	0G32423	If	
RE	0H32444	Vary	
RE	0I30067	Resume	
RE	0J30144	Jump	
RE	0K30030	Stop	
RE	0L30225	End of Tape	
RE	0M30052	Exit	
RE	0N30045	Start	
RE	0033663	Equation	
RE	0P31050	Pseudo Op. Heading	
RE	0Q4400	Most Translators	Starting addresses in core
RE	0R4000	Eq and List	
RE	0S4566	If	
RE	0U4700	Jump	
RE	CP51075	Compute	1st addresses of Translators on drum
RE	RE53350	Read	
RE	TL53573	Type	
RE	LM55013	List	
RE	PS60766	Print	
RE	KP61236	If	
RE	VY63661	Vary	
RE	RV66325	Resume	
RE	SJ66414	Jump	
RE	SP66560	Stop	
RE	EU66610	End of Tape	
RE	EZ67035	Exit	
RE	ST67107	Start	
RE	YA67154	Equation	
RE	HE74530	Pseudo Op. Heading	

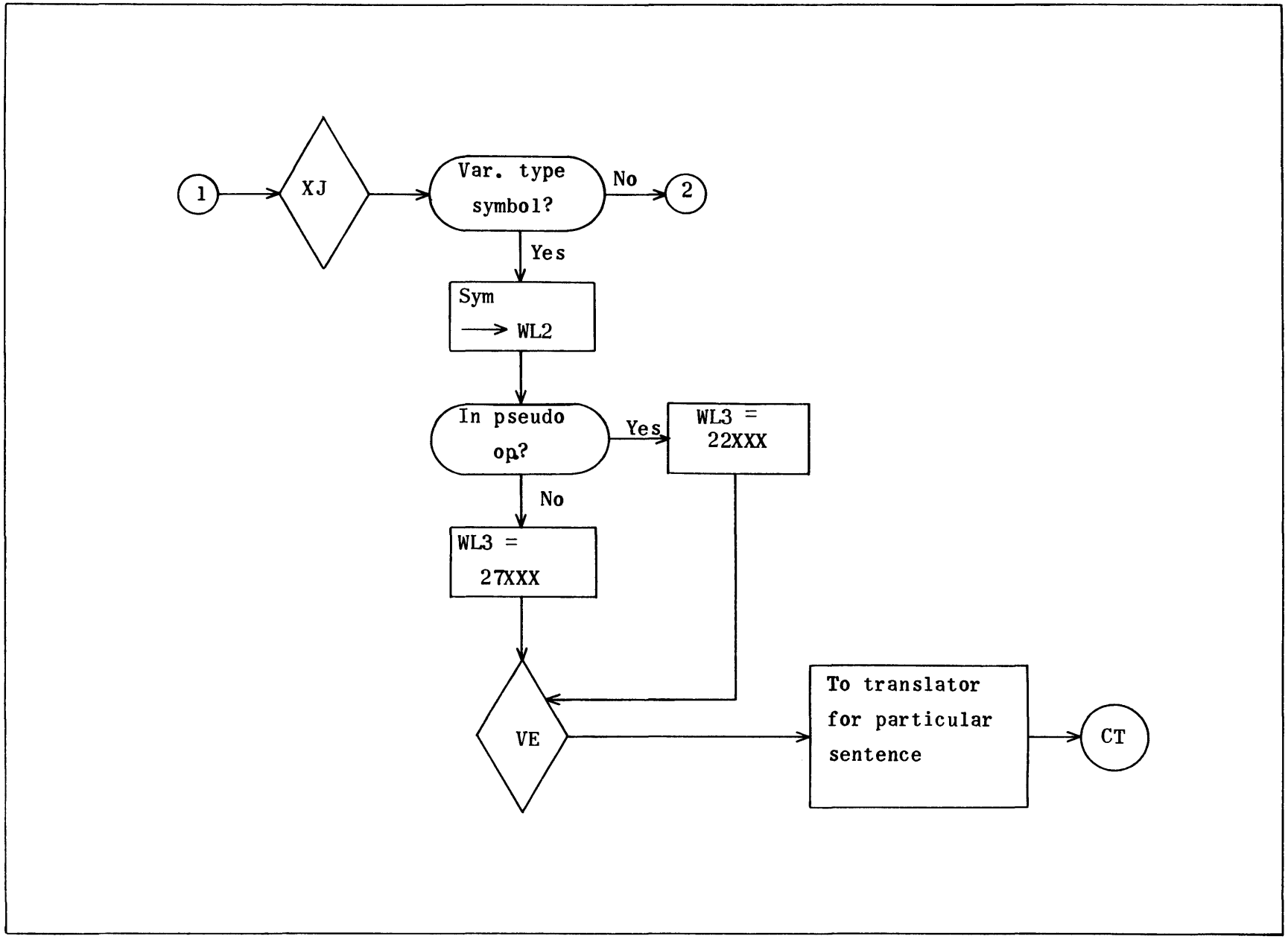
(a) Flow Charts of Translation Subroutines

(a) Setup Translation Phase

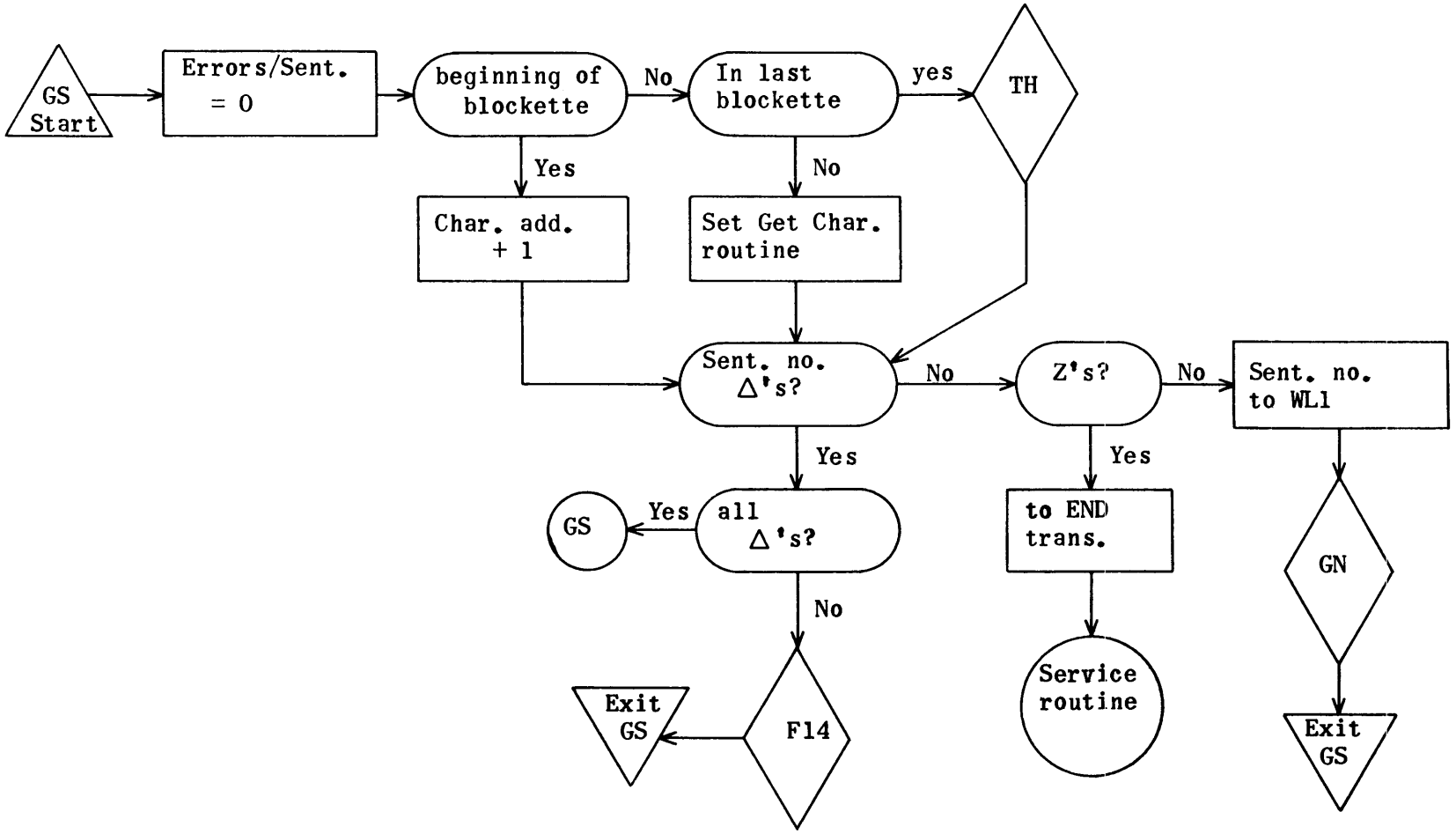


(b) Translation Control (CT)

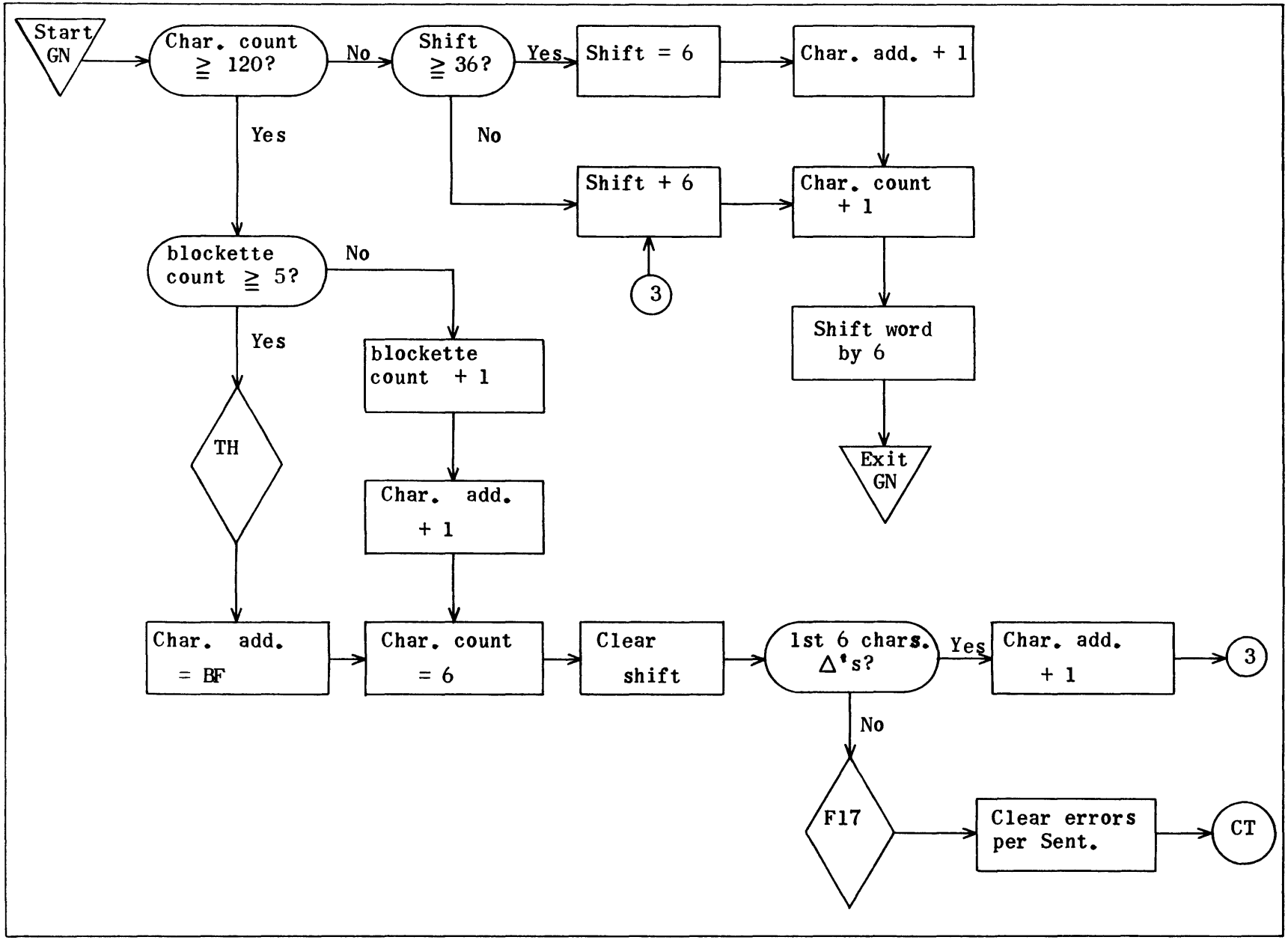


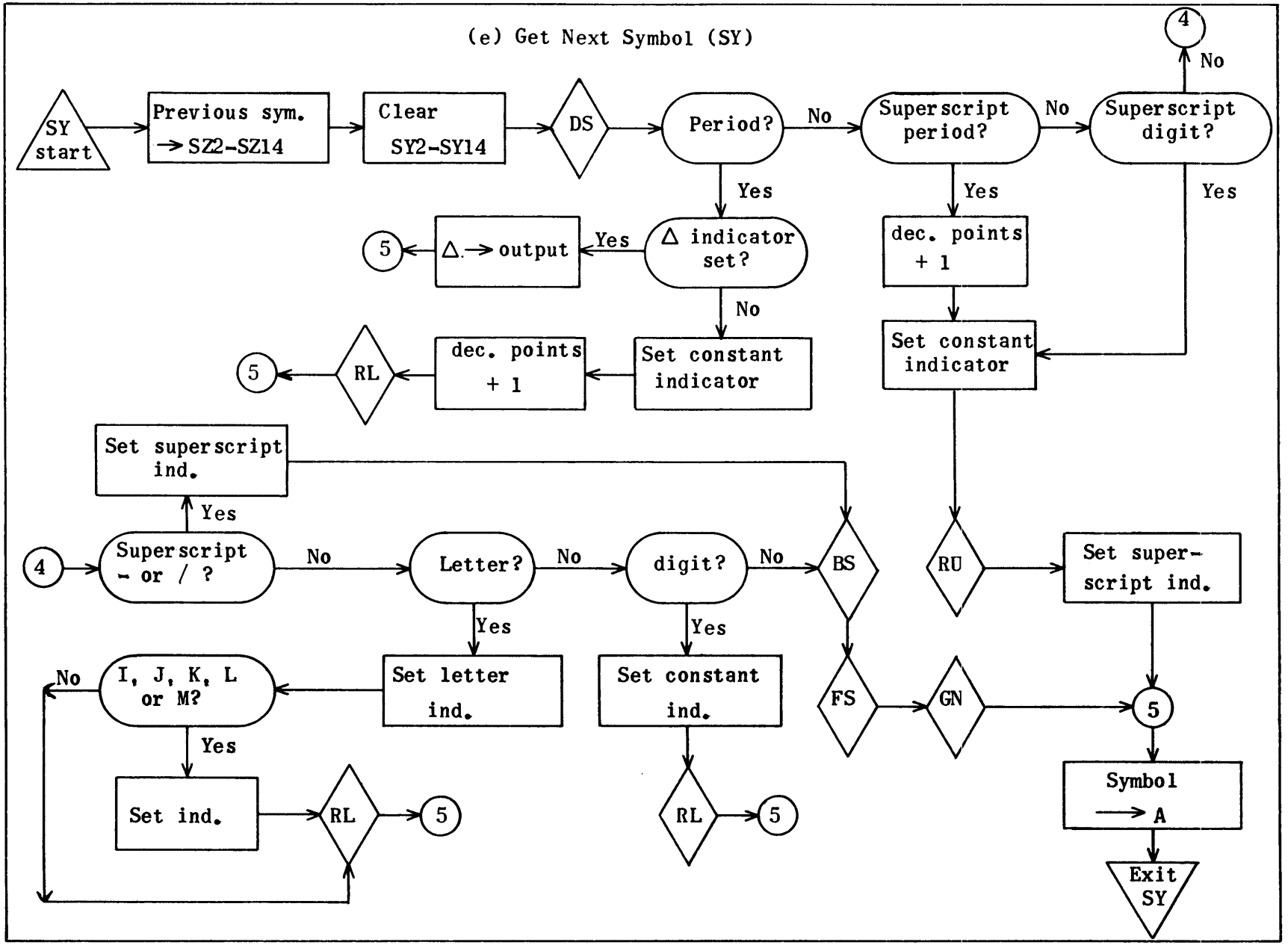

1
XJ
Var. type symbol?
No
2
Yes
Sym → WL2
In pseudo op?
Yes
WL3 = 22XXX
No
WL3 = 27XXX
VE
To translator for particular sentence
CT

(c) Get Next Sentence (GS)

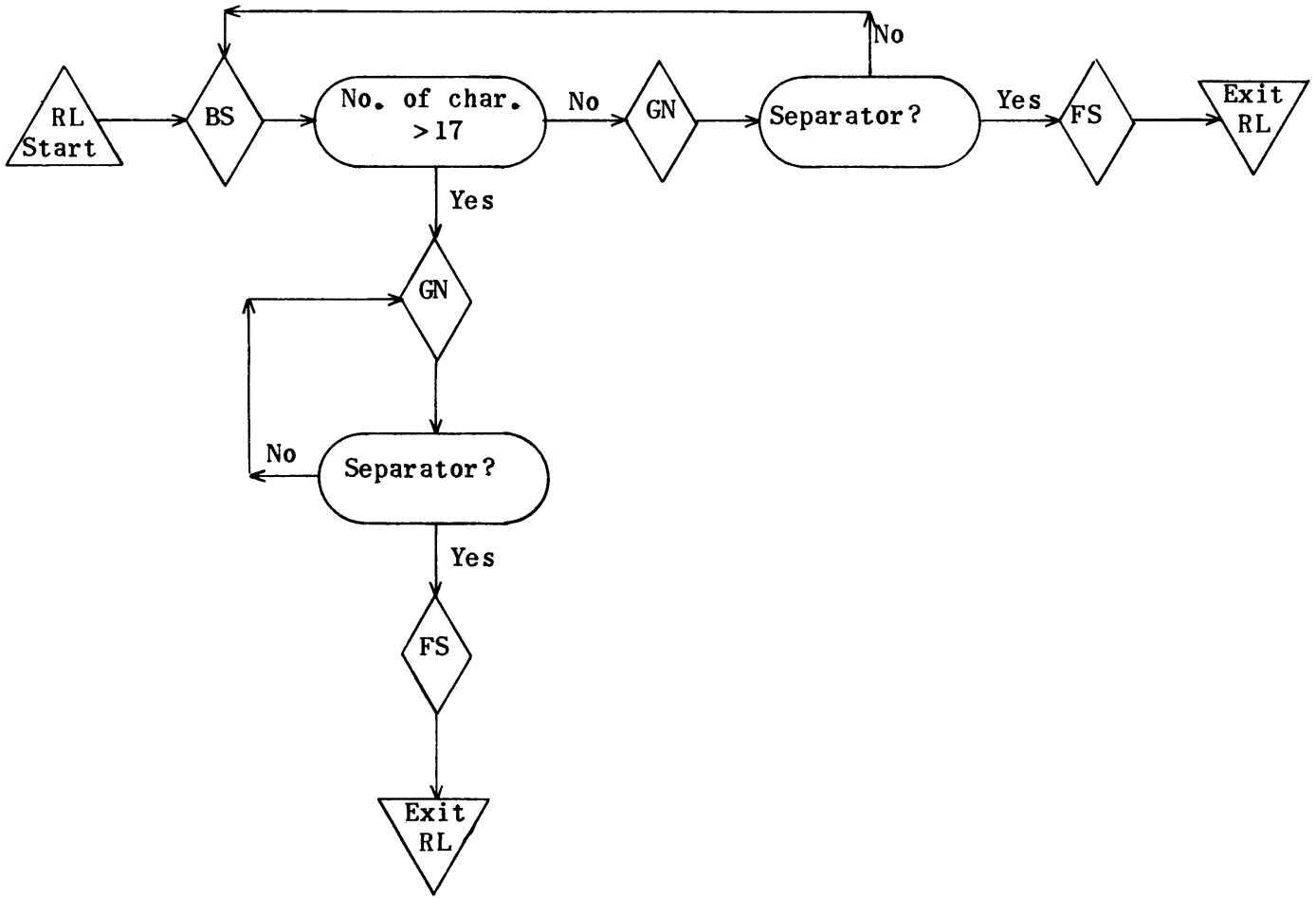


(d) Get Next Character (GN)



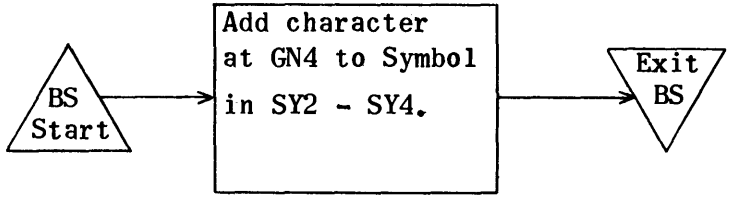

(e) Get Next Symbol (SY)
SY start
Previous sym. → SZ2-SZ14
Clear SY2-SY14
DS
Period?
No
Superscript period?
No
Superscript digit?
No
4
Yes
Yes
Yes
Δ indicator set?
Yes
Δ → output
5
No
dec. points + 1
Set constant indicator
Set constant indicator
dec. points + 1
RL
5
Set superscript ind.
Yes
4
Superscript - or / ?
No
Letter?
No
digit?
No
BS
RU
Set super-script ind.
Yes
Yes
Yes
No
I, J, K, L or M?
Set letter ind.
Set constant ind.
FS
GN
5
Yes
Set ind.
RL
5
RL
5
Symbol → A
Exit SY

(f) Get Rest of Lower Symbol (RL)

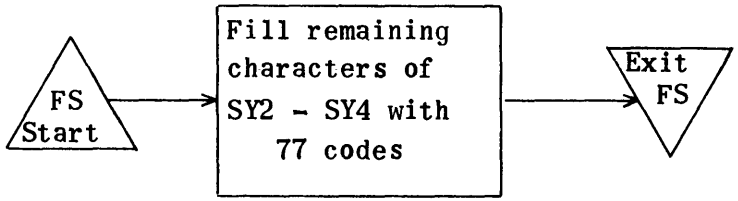


(g) The Get Rest of Superscript Symbol (RU) is the same as above except that the characters which are separators are somewhat different.

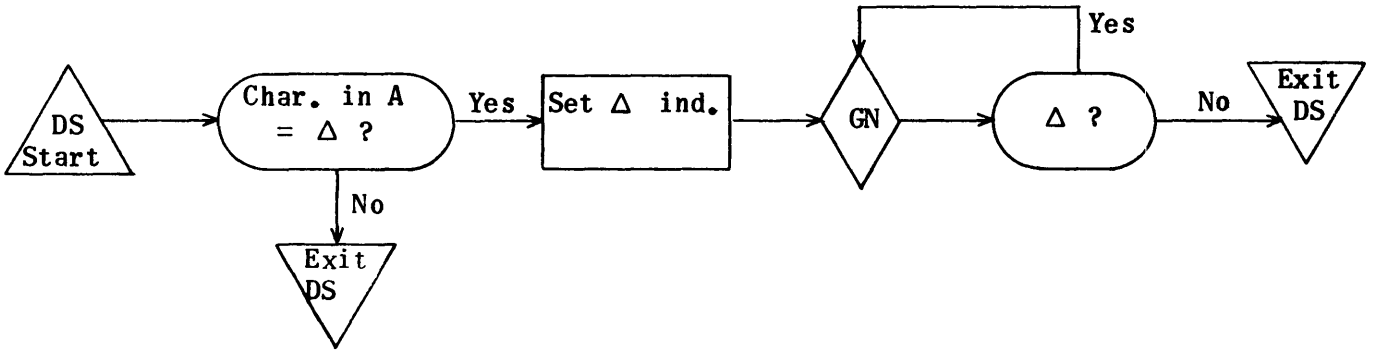
(h) Build Symbol (BS)



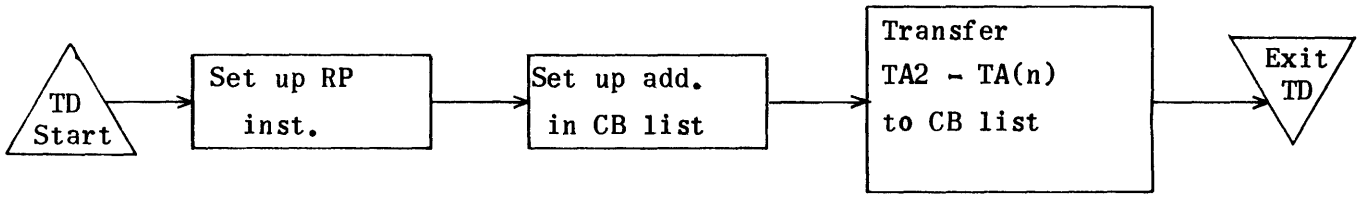
(i) Fill Symbol (FS)



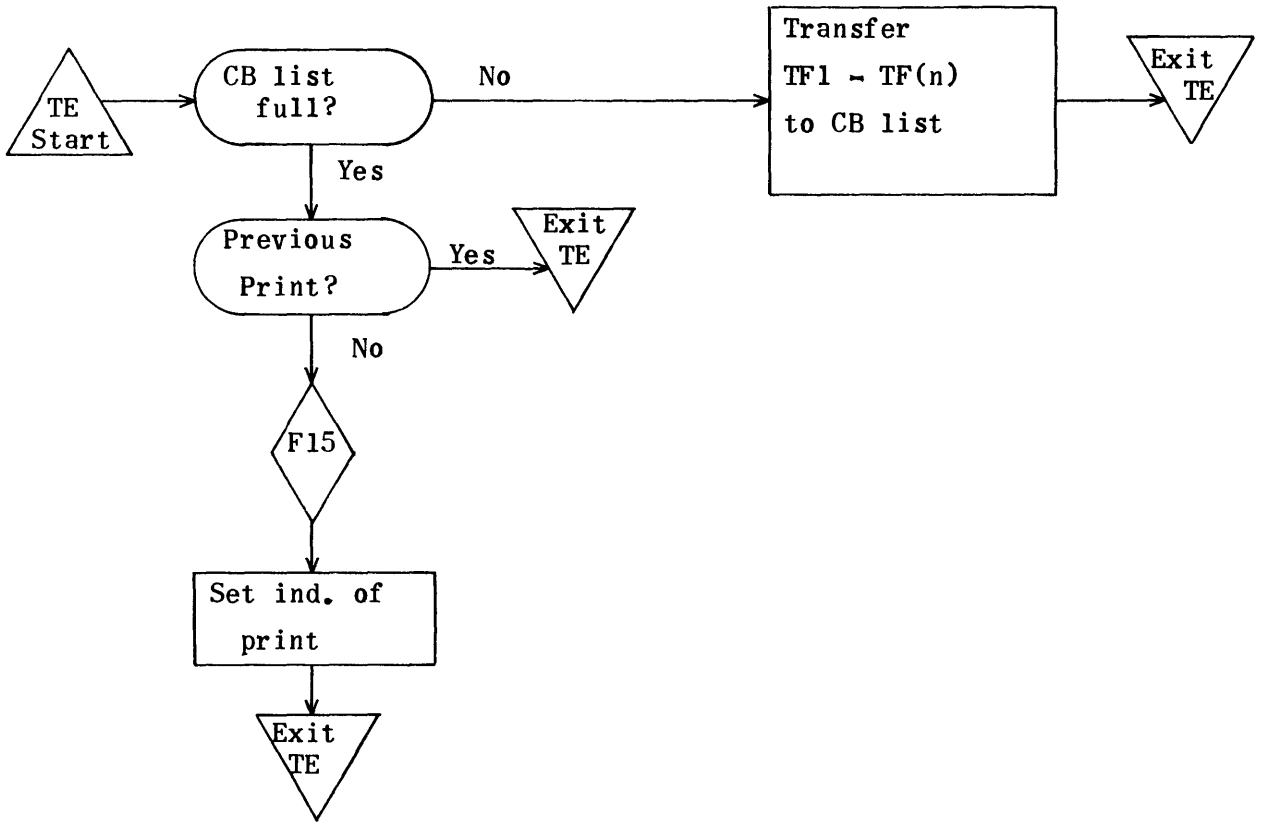
(j) Delete Spaces (DS)



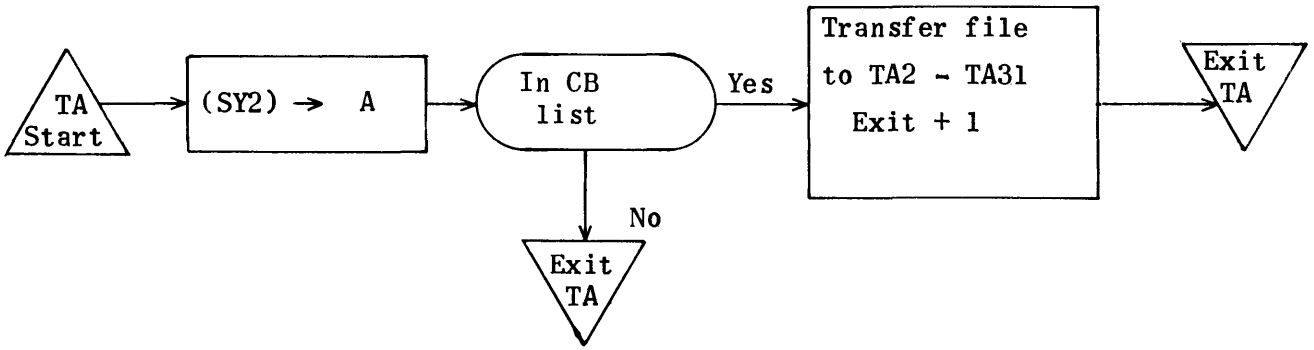
(k) Send File Back to Combination List (TD)



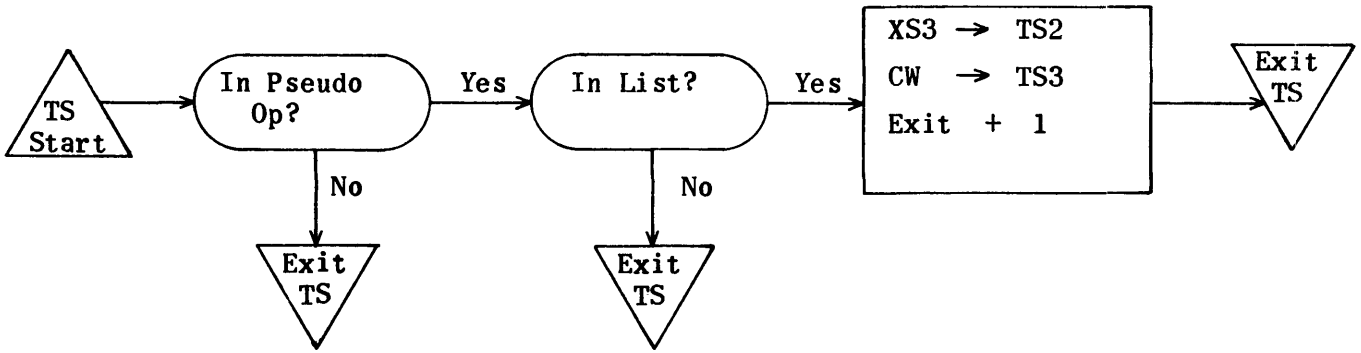
(l) Add File to CB List (TE)



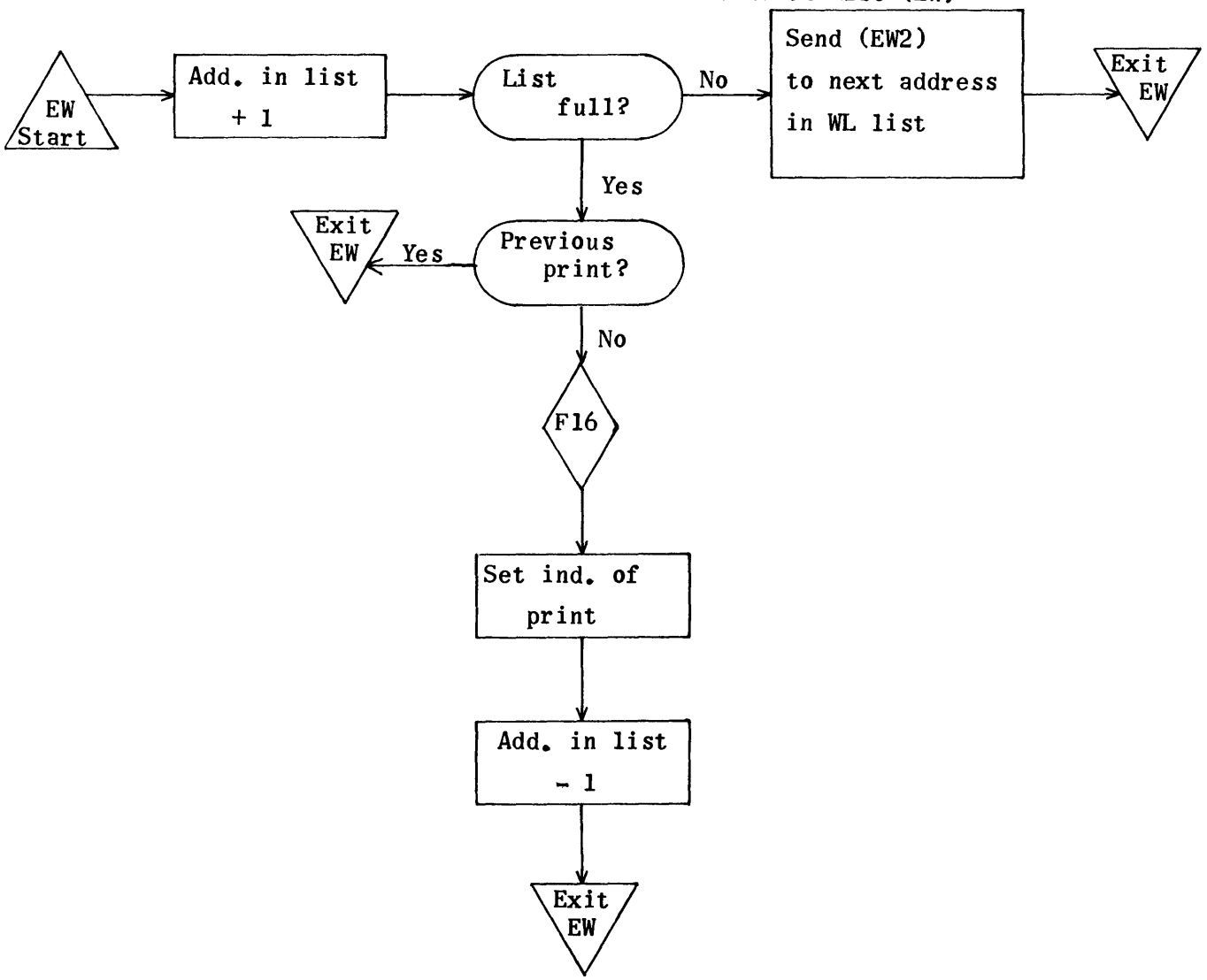
(m) Get File from CB List (TA)



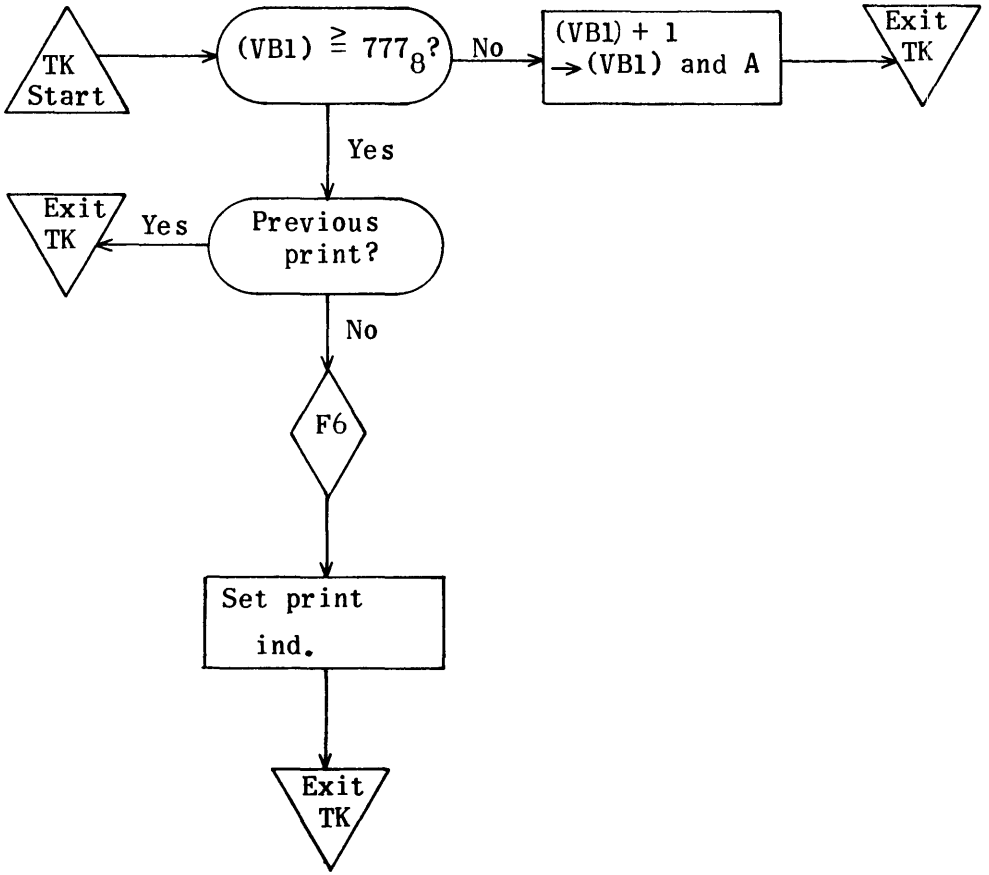
(n) Get Call Word from Pseudo Operation Dummy List (TS)



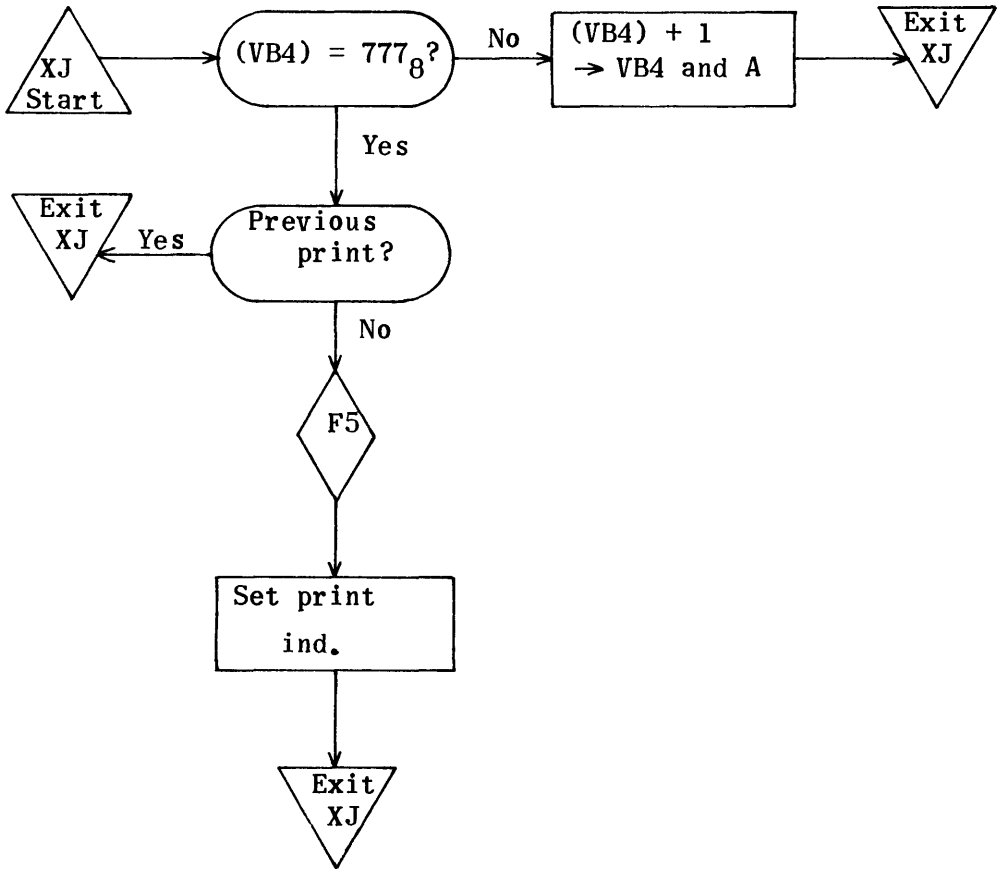
(o) Send Call Word to Translation List (EW)



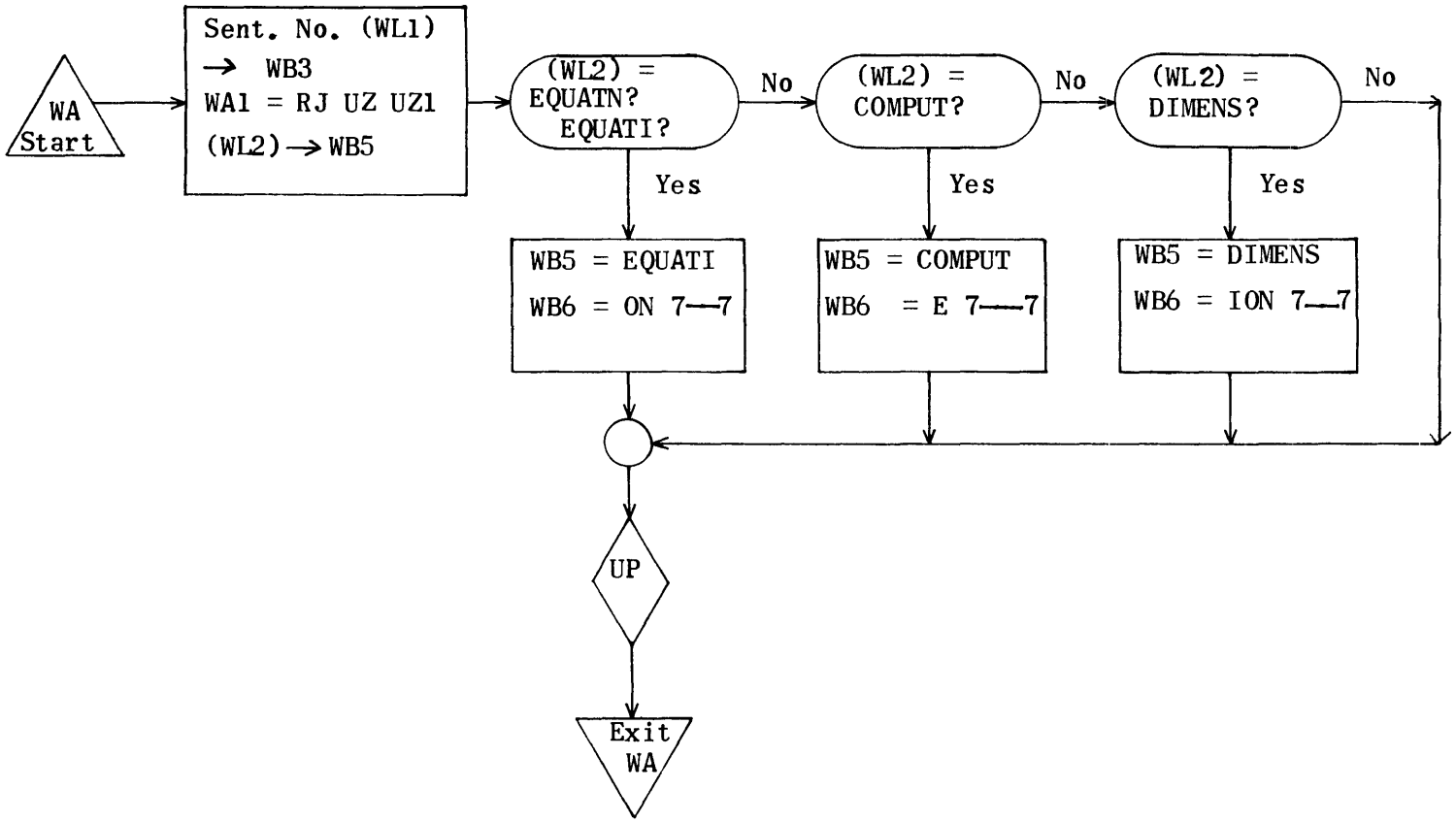
(p) Increase 66XXX, 65XXX, 64XXX Call Word Counter (TK)



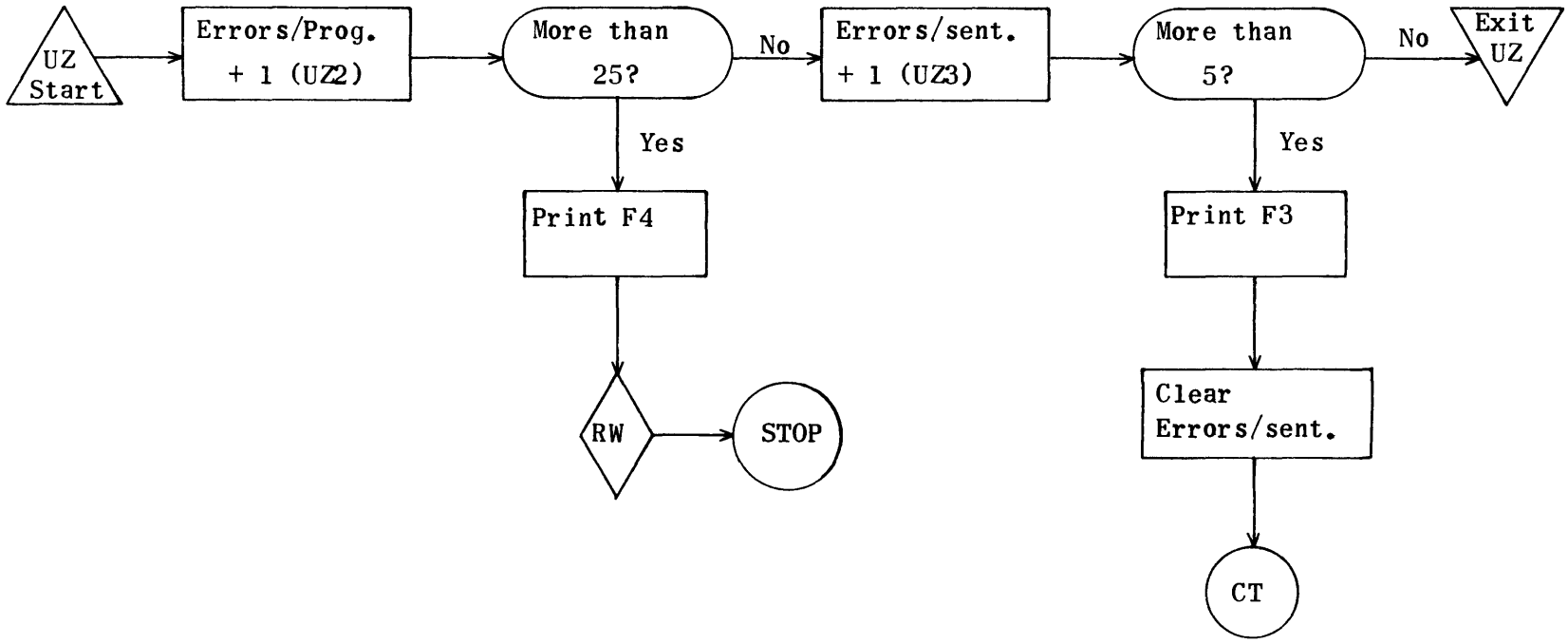
(q) Increase 26XXX, 27XXX, 22XXX Sentence CW Counter (XJ)



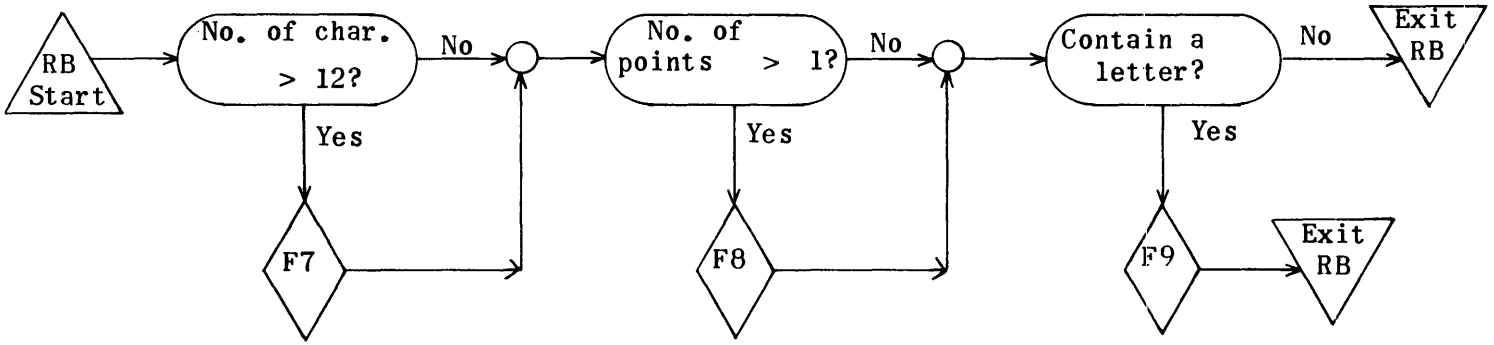
(r) Print Error Heading (WA)



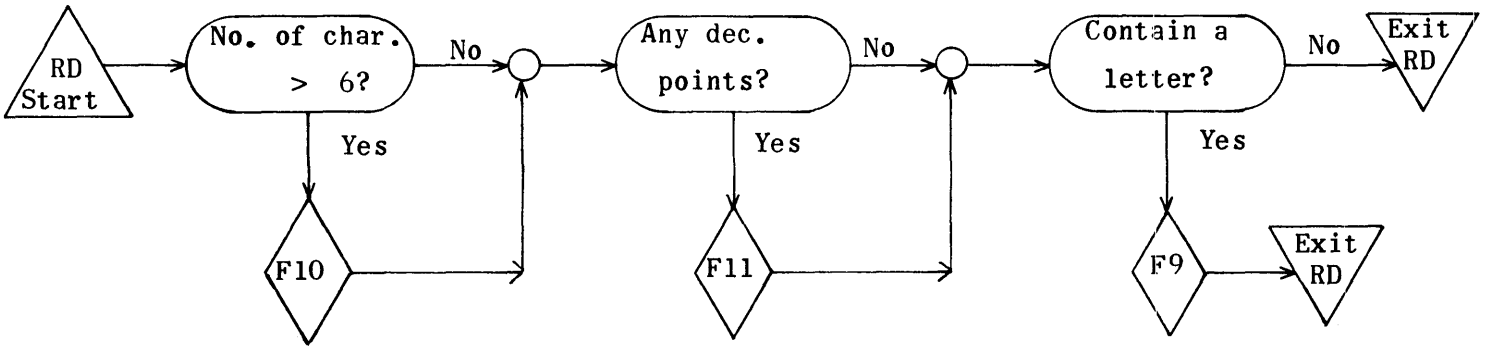
(s) Error Routine (UZ)



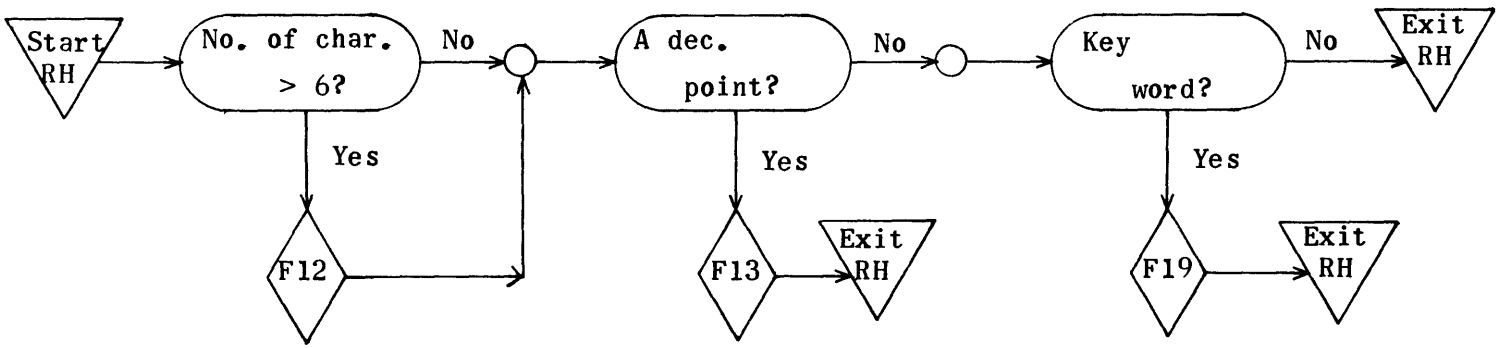
(t) Check Floating Point Constant (RB)



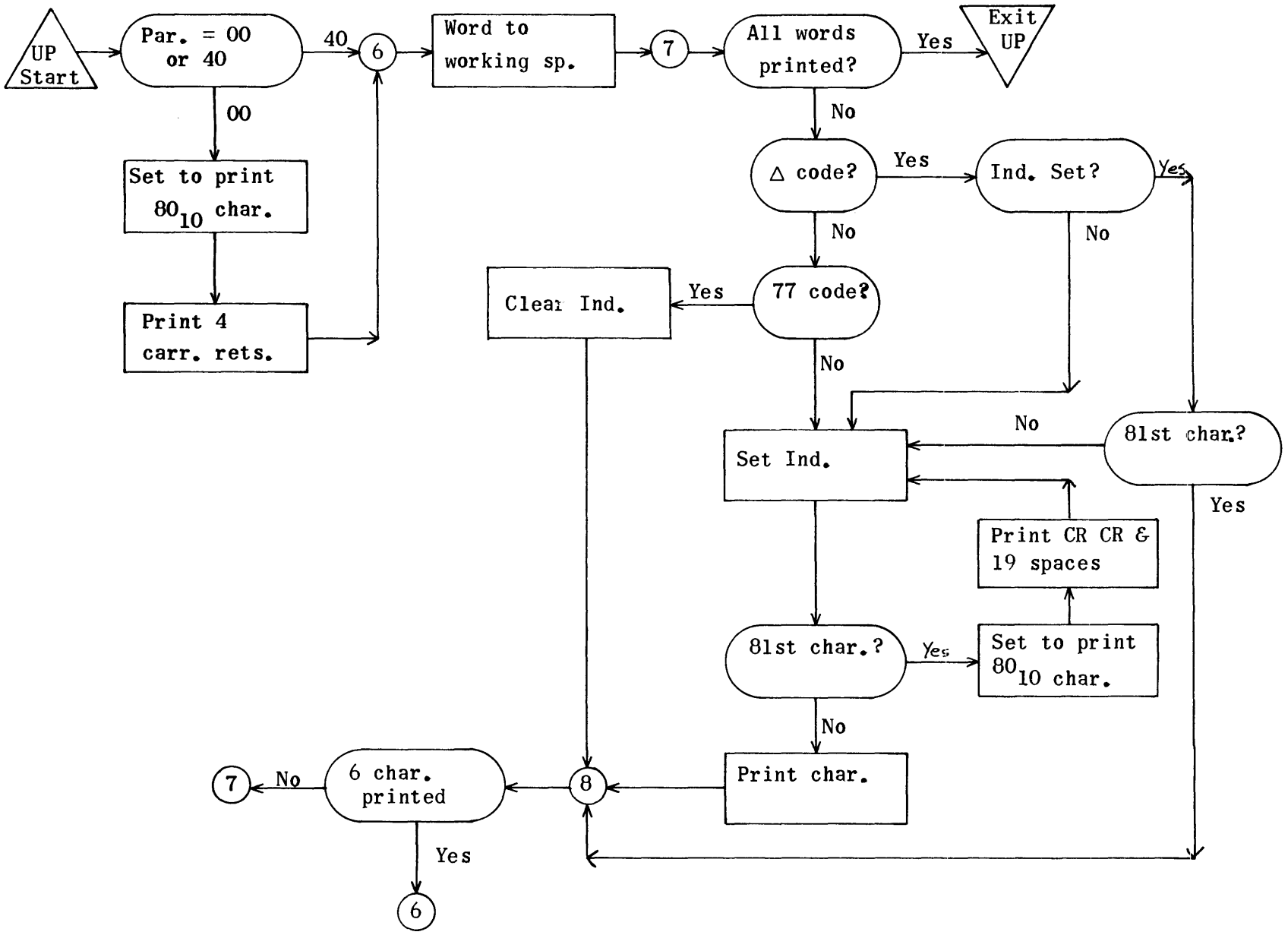
(u) Check Fixed Point Constant (RD)



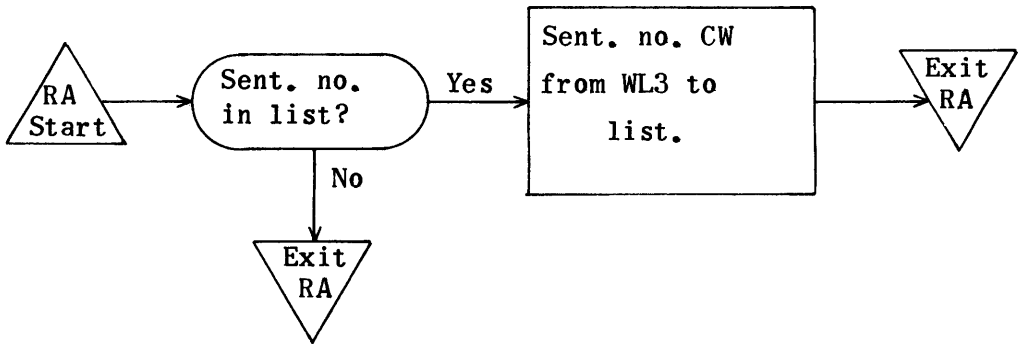
(v) Check Variable Type Symbol (RH)



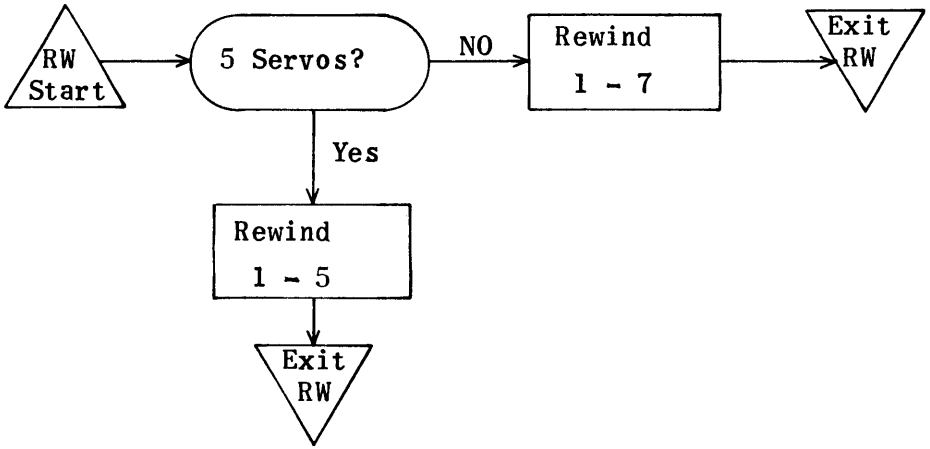
(w) Print Text (UP)



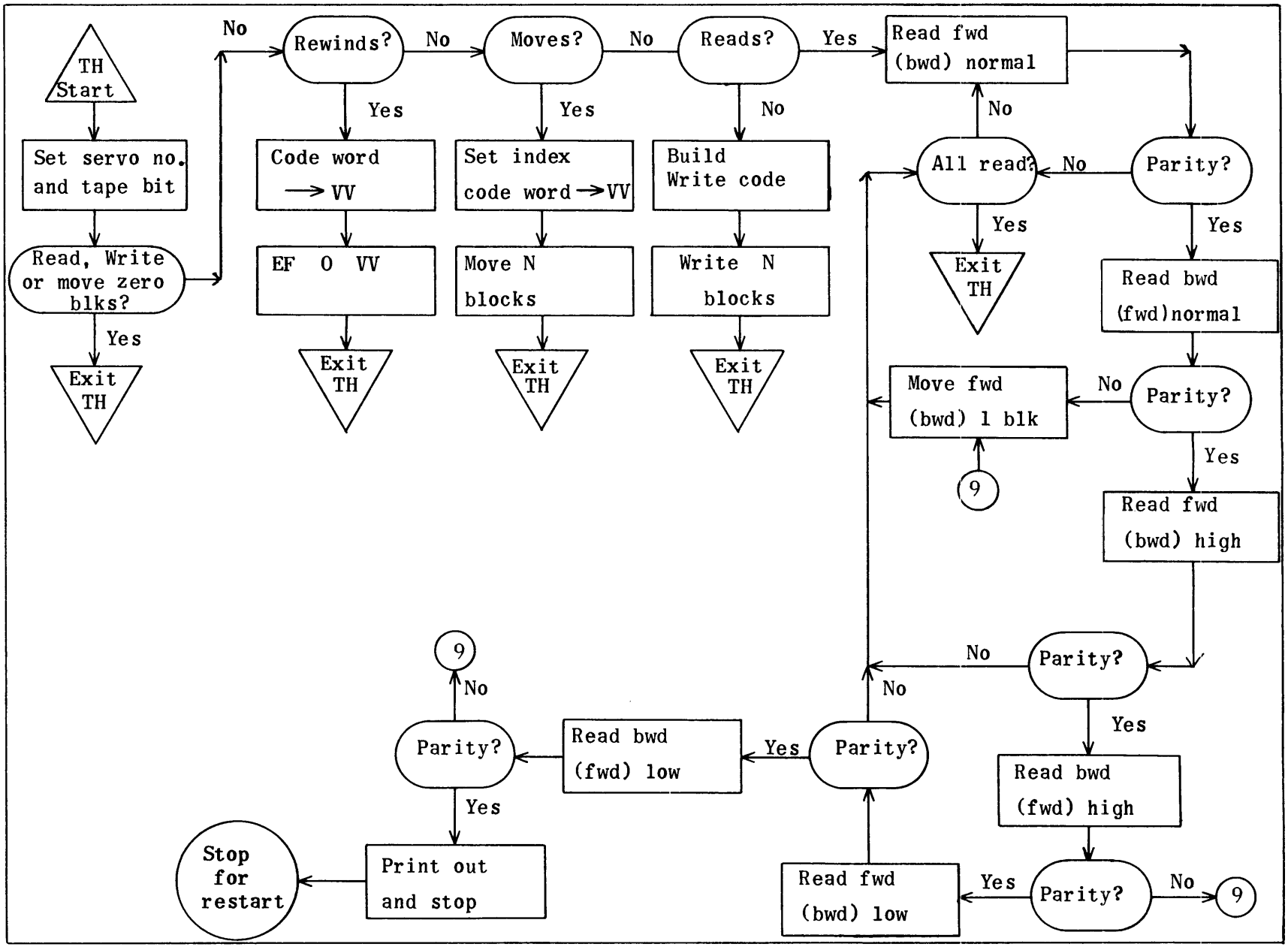
(x) Put Call Word in Sentence Number Reference List (RA)



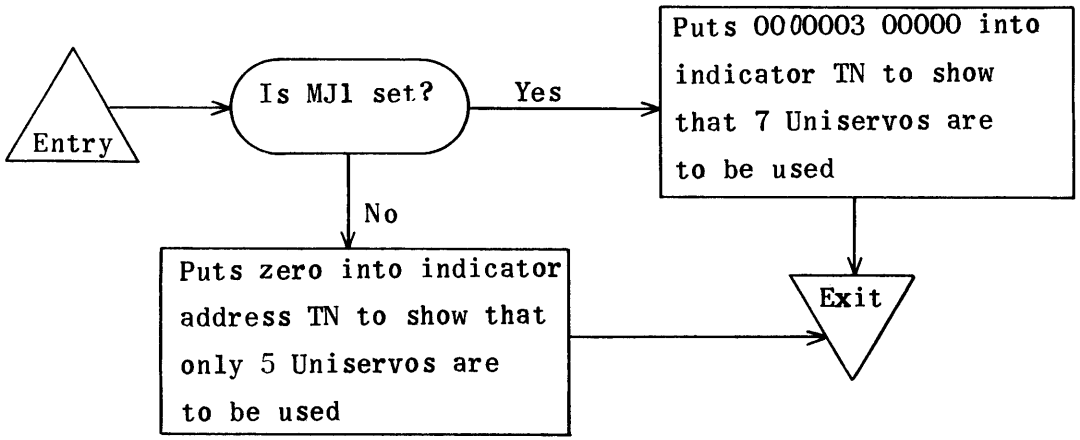
(y) Rewind All Tapes (RW)



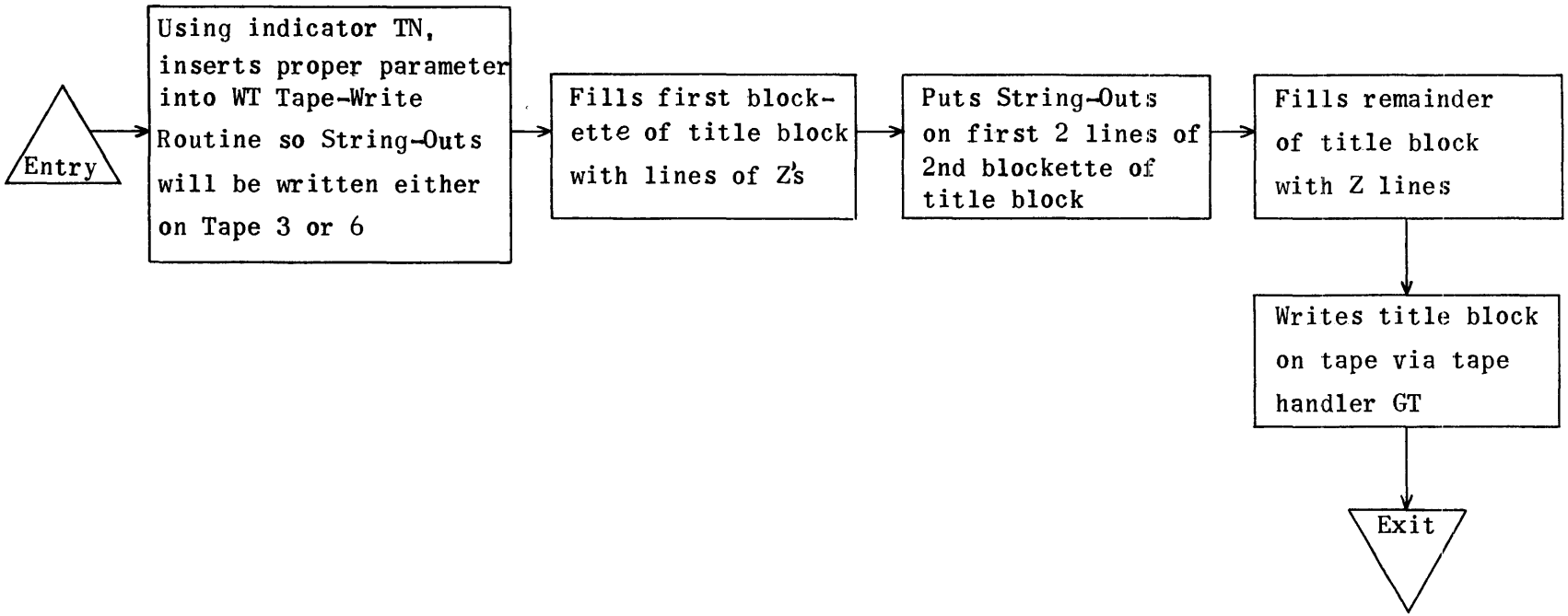
(z) Tape Handlers (TH) 1105 and 1103A



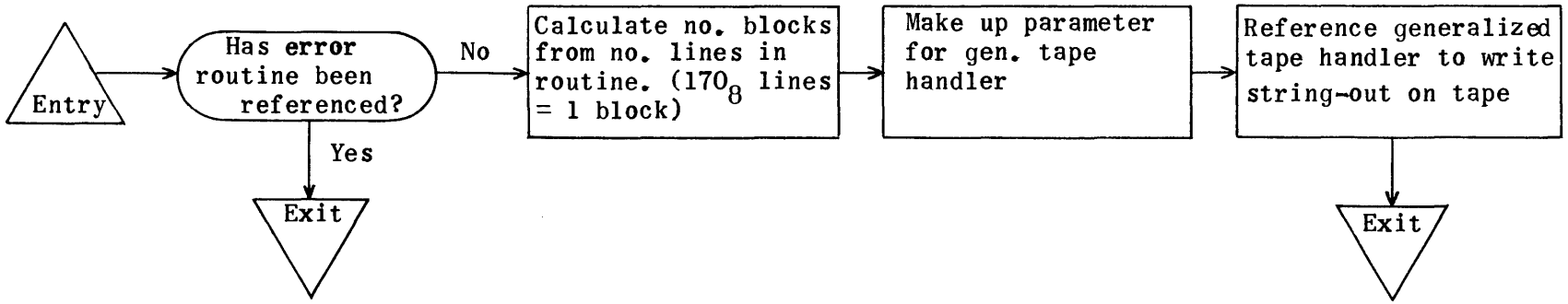
(aa) Routine To Set TN Indicator for 5 or 7 (MJ1) Uniservos (OT)



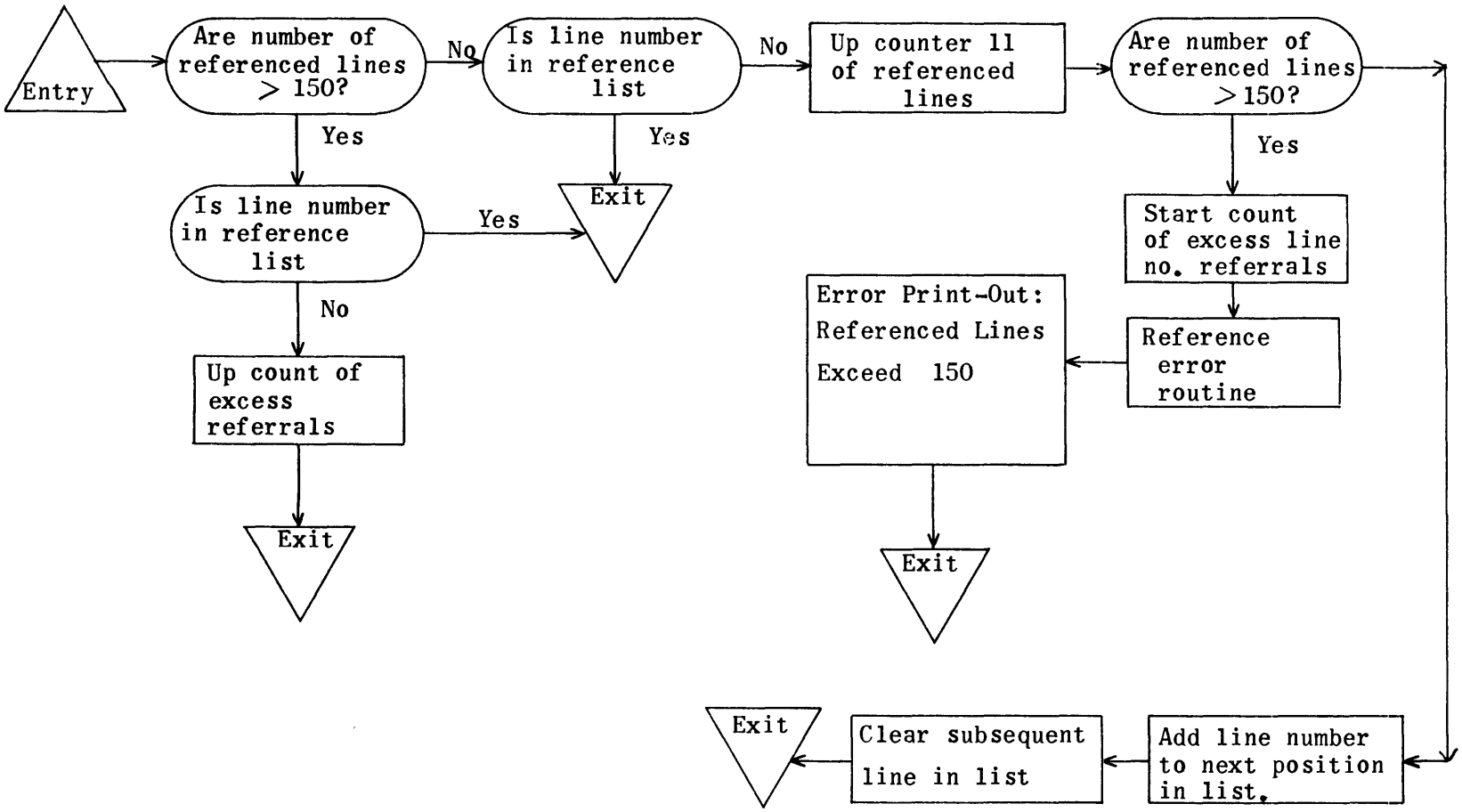
Routine To Write Title Block of String-Out Tape and Select Proper Parameter for WT Tape-Write Routine Depending on Value in TN or Availability of 5 or 7 Uniservos (UB)



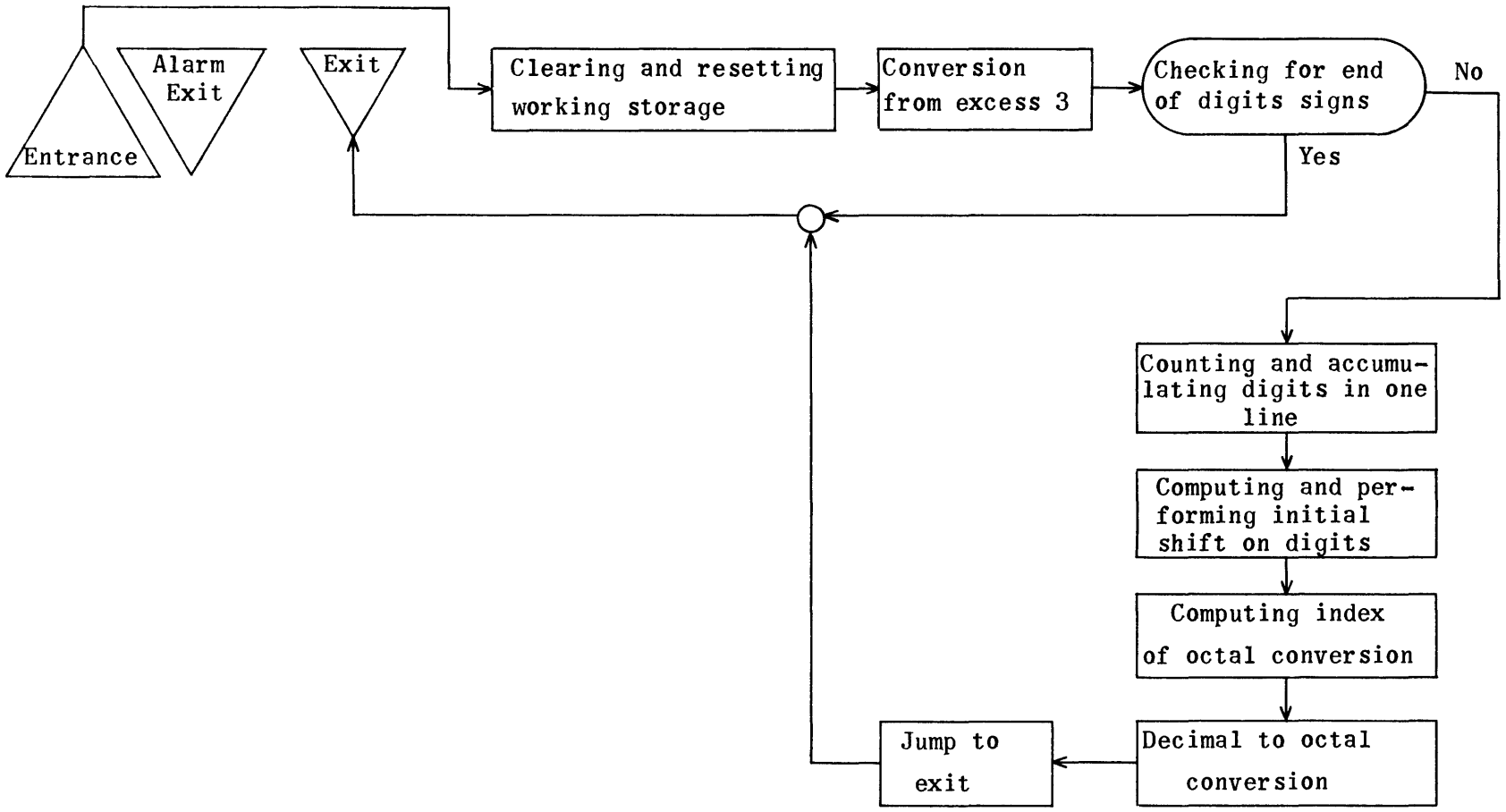
(ab) Translation Tape Write Routine (SS or WT)



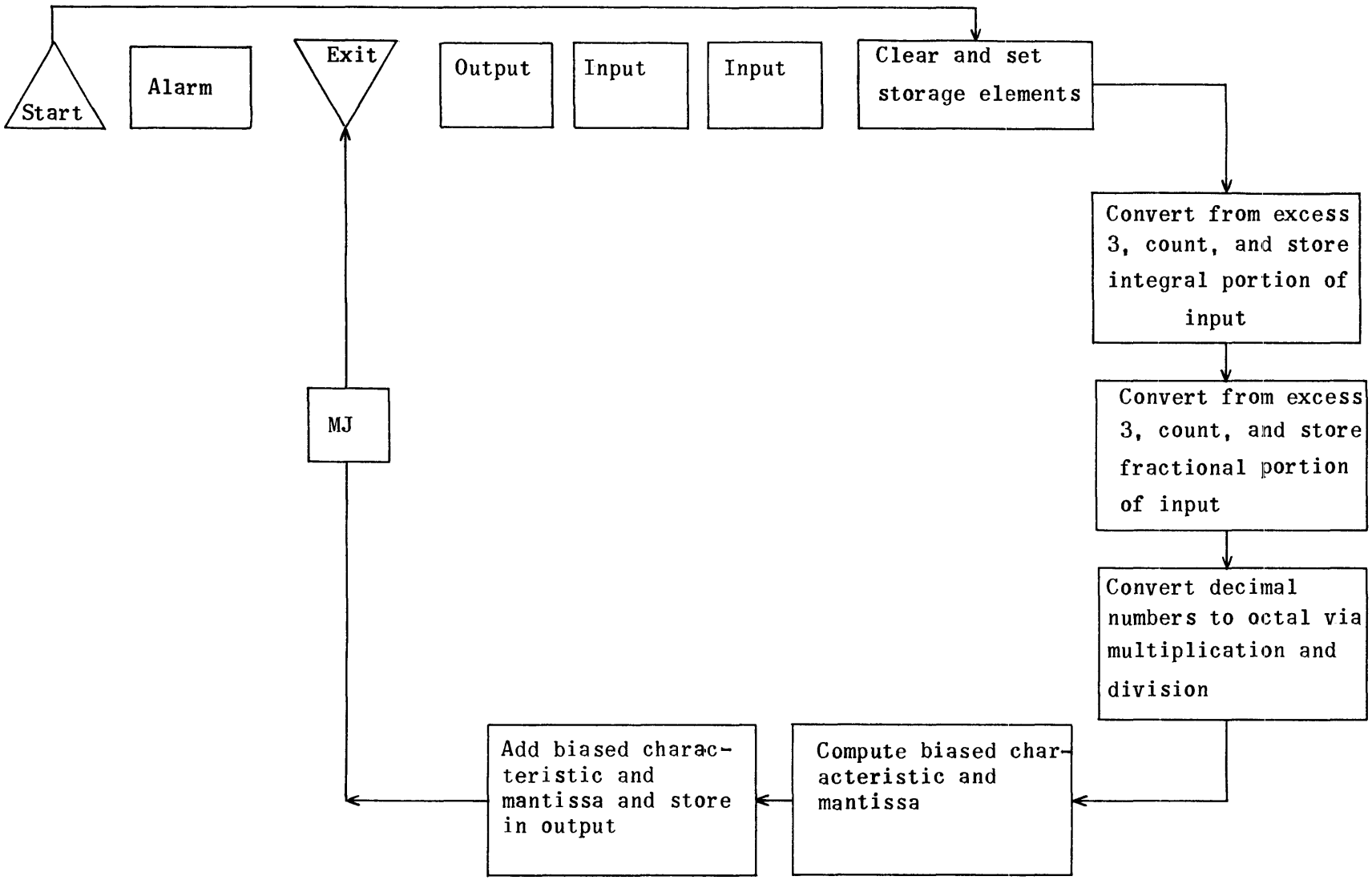
(ac) Put Referenced Sentence Number Into List IZ (IX)



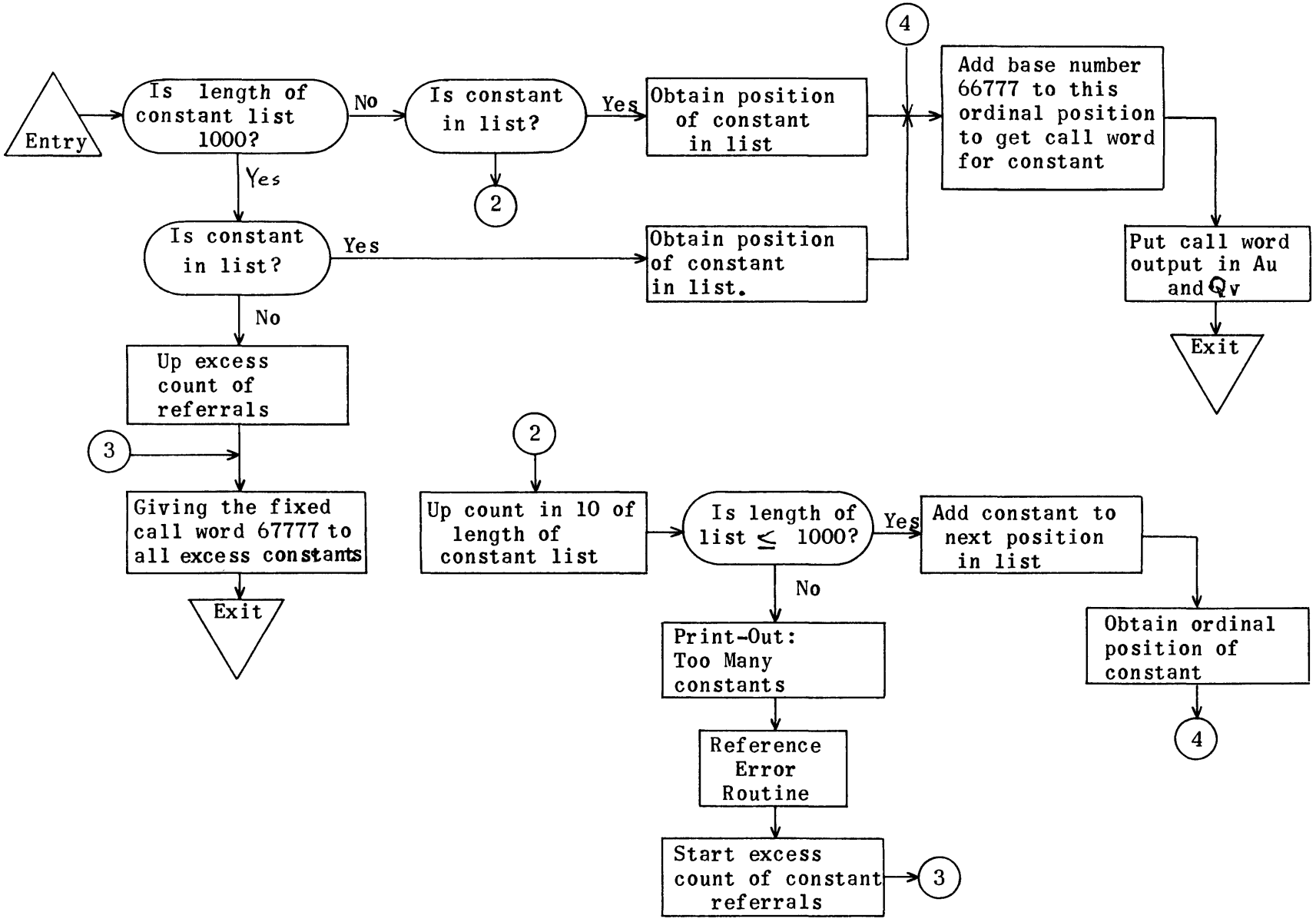
(ad) Excess-Three Decimal to Octal (RS)



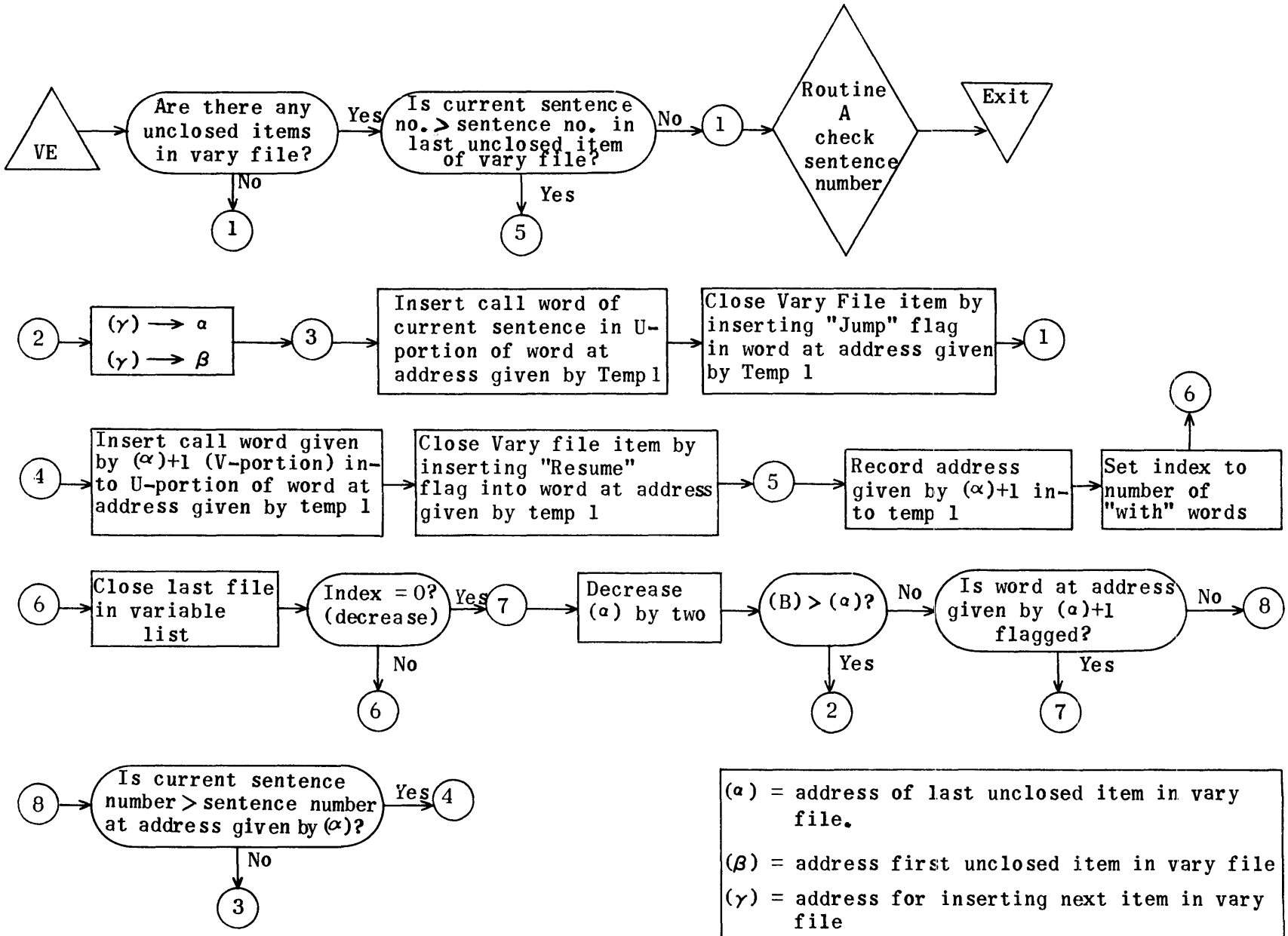
(ae) Excess-Three Decimal to Floating Point (GG)



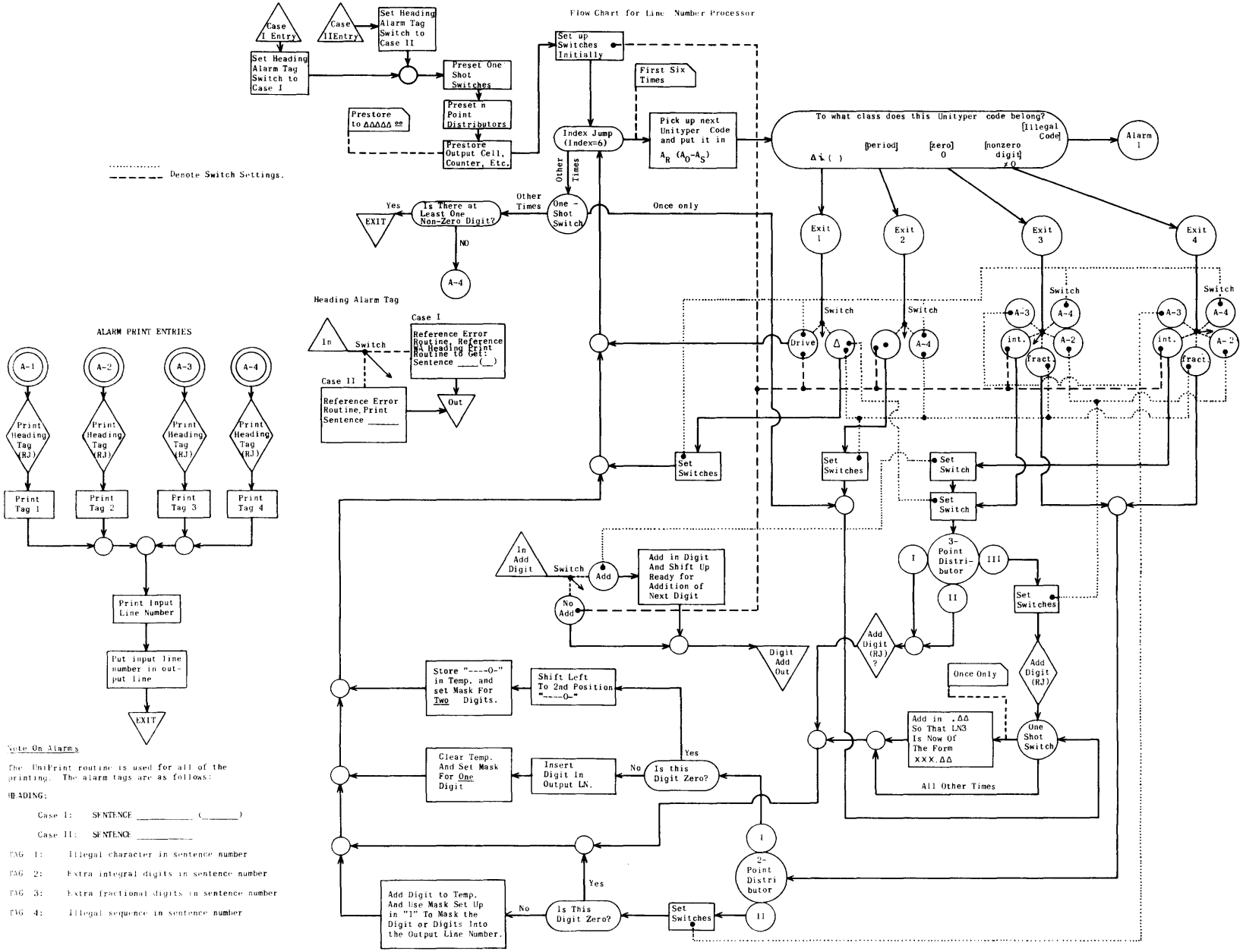
(af) Assign Constant Call Word (GW)



(ag) Close VARY File (VE)



Close Vary File and Variable List



Note On Alarms

The UniPrint routine is used for all of the printing. The alarm tags are as follows:

HEADING:

Case I: SENTENCE _____ (_____)

Case II: SENTENCE _____

TAG 1: Illegal character in sentence number

TAG 2: Extra integral digits in sentence number

TAG 3: Extra fractional digits in sentence number

TAG 4: Illegal sequence in sentence number

(10) Coding of Translation Subroutines
(a) Regions Setup Translation
(Translation Subroutine Regions necessary also)

```
ZZ7230
    40
CE7270
    41
ZY7331
    44
ZC7375
    41
  7436

BB7116

CR5075                  Entrance dim #2

0Q4400                  Entrance dim #1  - 1

0V37020                 No. wds in buffer for RP3

0Z75413                 = FA + 3 x (no. wds in buffer)

0Y30165                 No. wds in last blk of translators

TI7610                  Tape image
```

Setup Translation Phase
(the 2 blocks after the merge)

```
     IA   ZZ                          7230
 0   TP   CE32     BB2       }       Read in 2nd blk
 1   RJ   BB       BB1       }
 2   TP   CE36     CE35              Set index
 3   TV   CE37     ZZ7               Set address
 4   TP   CE3      ZC2       )
 5   RJ   ZC       ZC1       }       Read 30(10) of S.O. --> drum
 6   RP   0V       ZZ10      }
 7   TP   1        (30000)   )
10   RA   ZZ7      CE40              Increase address
11   IJ   CE35     ZZ4               Read 90(10) blks -->drum
12   TP   CE4      ZC2       )
13   RJ   ZC       ZC1       }       Last of S.O. -->drum
14   RP   0Y       ZZ16      }
15   TP   1        0Z        )
16   TP   CE33     BB2       )
17   RJ   BB       BB1       }       FC --> drum
20   RP   30100    ZZ22      }
21   TP   TI       FC        )
22   TP   CE5      ZC2       }       Read in S.O.S.
23   RJ   ZC       ZC1       }
24   TP   CE34     TH3       }       Read in dim. # 1 & # 2
25   RJ   TH2      TH        }
26   RJ   ZY       ZY1               Setup S.O.S.
27   TP   CE6      TH3       }
30   RJ   TH2      TH        }       Move # 5 fwd 1 blk
31   RJ   GS       GS1               Get sentence
32   RJ   SY       SY1               Get sym --> A
33   EJ   CE7      ZZ35              DIMENS --> ZZ35  no ↓
34   MJ   0        CR                --> Dim. # 2
35   TP   SY2      WL2               WL2 = DIMENS
36   RJ   SS       SS1               Send S.O. --> tape
37   MJ   0        0Q1               --> dim. # 1
     CA   ZZ40
```

Constants

	IA	CE		
0	0	0	0	Zero
1	0	0	1	One in V
2	0	1	0	One in U
3	0	1	7020	S.O. parameters 30_{10} blks
4	0	1	165	Last S.O. par.
5	0	TH	ZS27	S.O.S. par.
6	30	105	0	Move # 5 fwd 1 blk
7	27	34473	05065	D I M E N S
10	0	0	37700	VBO
11	0	20000	0	
12	77	77777	77776	VB1 - VB4
13	0	CB1	CB1	
14	0	BF165	BF165	
15	01	22777	77777	Δ .
16	0	VL1	VL1	
17	0	CE20	12	
20	01	01010	10101	Δ Δ Δ Δ Δ Δ
21	52	24656	50134	P A S S Δ I
22	22	01010	10101	. Δ Δ Δ Δ Δ
23	01	66542	45065	Δ T R A N S
24	46	24663	45150	L A T I O N
25	02	02305	45451	- - E R R O
26	54	01273	06630	R Δ D E T E
27	26	66345	15001	C T I O N Δ
30	24	50270	17124	A N D Δ W A
31	54	50345	03265	R N I N G S
32	0	170	ZZ170	Read 2nd blk
33	0	170	TI	Read in FC
34	50	00401	OQ	Dim. # 1 & # 2 par.
35	0	0	0	Index
36	0	0	2	Set index
37	0	0	FA	
40	0	0	7020	
	CA	CE41		

Setup Translation Subroutines

```
    IA   ZY
 0  MJ   0       (30000)      Exit
 1  TP   CE10    VB           Start, VB = 0 0 37700
 2  RP   10004   ZY4    }
 3  TP   CE12    VB1    }     = -1
 4  TP   CE13    TE2          TE2 = 0 CB1 CB1
 5  TP   CE11    CB           CB = 0 20000 0
 6  TP   CE14    GN2          GN2 = 0 BF165 BF165
 7  RP   10004   ZY11   }
10  TP   CE      GN3    }     = 0
11  TP   CE15    SY2          SY2 = Δ.
12  TP   CE      EW4    )
13  TP   CE      TE3    |
14  TP   CE      TS4    |
15  TP   CE      UZ2    }     = 0
16  TP   CE      UZ3    |
17  TP   CE      TK2    |
20  TP   CE      XJ2    )
21  TP   CE      VD           = 0
22  RP   10017   ZY24   }     = 0
23  TP   CE      1      }
24  RP   10005   ZY26   }     = 0  20000  0
25  TP   CE11    5      }
26  TP   CE      7            = 0
27  TP   CE      VD1          = 0
30  RJ   OT      OT1    }     Set up TN, write heading on tape.
31  RJ   UB      UB1    }
32  TP   CE1     13           No. blks = 1
33  TP   CE17    UP3    }     Print "Pass I etc."
34  RJ   UP2     UP     }
35  TP   CE16    VL           VL = 0  VL1  VL1
36  TP   CE11    JN           JN = 0  20000  0
37  TP   CE11    WR           WR = 0  20000  0
40  TP   CE      FX13         = 0
41  TP   CE      IX47         = 0
42  TP   CE11    VF           = 0  20000  0
43  MJ   0       ZY
    CA   ZY44
```

Read n blks to storage

```
     IA   ZC
 0   MJ   0       (30000)      Exit
 1   MJ   0       ZC3
 2   0    30000   30000        0   1st address   last address
 3   TP   ZC2     Q            Par. → Q
 4   QT   ZC34    ZC35         Store 1st address in u
 5   LQ   Q       17           Last address → Q_u
 6   QT   ZC34    A
 7   ST   ZC35    A       }    Last - 1st + 1 → A_u
10   AT   ZC37    A       }
11   LQ   Q       6            1st add → Q_v
12   TV   Q       BB2          Set up 1st add.
13   DV   ZC40    ZC33         n/170_8 → index
14   TP   A       ZC32         Store remainder
15   TP   Q       A       }    Quotient = 0 → ZC26
16   ZJ   ZC17    ZC26    }            no ↓
17   RS   ZC33    ZC36         Index - 1
20   TU   ZC40    BB2          No. wds = 170_8
21   RJ   BB      BB1          Read blk
22   RA   BB2     ZC31         Address + 170_8
23   IJ   ZC33    ZC21         n blks
24   TP   ZC32    A       }    Remainder = 0 → exit
25   ZJ   ZC26    ZC      }            no ↓
26   TU   ZC32    BB2          Remainder → par.
27   RJ   BB      BB1          Read remainder
30   MJ   0       ZC           Exit
31   0    0       170          120_10
32   0    0       0            Remainder
33   0    0       0            Index
34   0    77777   0
35   0    (30000) 0
36   0    0       1
37   0    1       0
40   0    170     0
     CA   ZC41
```

(b) Translation Control

```
     IA   CT
 0   TP   WL2     A       }   PRINT ⟶ CT6
 1   EJ   KB10    CT6     }     no ↓
 2   TP   SY2     A
 3   EJ   KK55    CT6         Δ. ⟶ CT6     no ↓
 4   RJ   SY      SY1         Sym ⟶ A
 5   MJ   0       CT3
 6   RJ   GS      GS1         Get sent. 1
 7   RJ   SY      SY1         Sym ⟶ A
10   EJ   KB      Z0          D I M E N S ⟶ Z0
11   TP   VD      A       }   After START ⟶ HA
12   ZJ   HA      CT13    }   Before ↓
13   TP   SY2     A           Sym ⟶ A
14   TP   KB27    WL2         WL2 = E Q U A T N
15   TP   SY7     Q       }
16   QJ   CT17    UA      }   Var. type ↓ no ⟶ UA
17   EJ   KB1     FW30        S T A R T ⟶ FW30   no ↓
20   RP   20023   CT22    )
21   EJ   KB      CT24    )
22   RP   20004   HA33    )
23   EJ   KB33    CT24    )   Key word ↓ no ⟶ HA33
24   RJ   UZ      UZ1     )
25   TP   WL1     FA3     )
26   TP   SY2     FA13    )   F1
27   TP   FA      UP3     )
30   RJ   UP2     UP      )
31   TP   SY2     A           Sym ⟶ A
32   EJ   KB16    FW24        END ⟶ FW24  no ↓
33   MJ   0       CT2         Return
34   RP   30004   FW24    }
35   TP   KA20    WL      }   END heading ⟶ string
     CA   CT36

     IA   Z0
 0   RJ   WA      WA1     )
 1   TP   NO10    UP3     )   F20
 2   RJ   UP2     UP      )
 3   MJ   0       CT2         ⟶ Control
     CA   Z04
```

After START Control

```
     IA    HA
 0   RJ    XJ      XJ1          Stm't CW + 1
 1   TP    SY7     Q       }
 2   QJ    HA3     UA      }    VAR, P.O., LIB ↓  no → UA
 3   TP    SY2     A       }
 4   TP    A       WL2     }    Sym → A → heading
 5   EJ    KB2     HB           VARY → HB
 6   TP    VB4     A            Stm't CW count → A
 7   TP    TS4     Q       }
10   QJ    HA11    HA13    }    In P.O. ↓ no → HA14
11   AT    KB25    WL3          CW = 22XXX
12   MJ    0       HA14         Skip HA13
13   AT    KB24    WL3          CW = 27XXX
14   RJ    VE      VE1          Routine A
15   TP    SY2     A            Sym → A
16   EJ    KB3     FW           Compute
17   EJ    KB4     FW2          Read
20   EJ    KB6     FW4          Type
21   EJ    KB7     FW6          List
22   EJ    KB10    FW10         Print
23   EJ    KB11    FW12         If
24   EJ    KB12    FW14         Replace
25   EJ    KB13    FW16         Resume
26   EJ    KB14    FW20         Jump
27   EJ    KB15    FW22         Stop
30   EJ    KB16    FW24         End
31   EJ    KB17    FW26         Exit
32   EJ    KB1     FW30         START
33   RJ    TA      TA1          Check CB list
34   MJ    0       FW32         Not in CB list → Eq.   in ↓
35   TP    TA4     Q       }
36   QT    KB30    A       }    C.W. → A
37   EJ    KB31    FW34         P.O. → heading translator
40   MJ    0       FW32         no    Eq.
     CA    HA41
```

```
    IA   UA                          Not variable type symbol
0   RJ   UZ       UZ1     )
1   TP   WL1      FA30    |
2   TP   SY2      FA34    }    F2
3   TP   FA25     UP3     |
4   RJ   UP2      UP      )
5   MJ   0        CT2          ⟶ Control
    CA   UA6

                                 Vary ↓

    IA   HB
0   TP   VB4      A
1   TP   TS4      Q       }    In P.O. — HB3 ↓
2   QJ   HB3      HB5     }    No ⟶ HB5
3   AT   KB25     WL3          CW  =  22XXX
4   MJ   0        HB6
5   AT   KB26     WL3          CW  =  26XXX
6   RJ   VE       VE1          Rout A.
7   MJ   0        FW14         ⟶ Vary
    CA   HB10
```

(c) Switch to Translator List

```
     IA   FW
 0   RP   OA      OQ1   }
 1   TP   CP      OQ    }     Compute
 2   RP   OC      OQ1   }     Read
 3   TP   RE      OQ    }
 4   RP   OD      OQ1   }     Type
 5   TP   TL      OQ    }
 6   RP   OE      OR1   }     List
 7   TP   LM      OR    }
10   RP   OF      OQ1   }     Print
11   TP   PS      OQ    }
12   RP   OG      OS    }     If
13   TP   KP      OQ17  }
14   RP   OH      OQ1   }     Vary
15   TP   VY      OQ    }
16   RP   OI      OQ1   }     Resume
17   TP   RV      OQ    }
20   RP   OJ      OU1   }     Jump
21   TP   SJ      OU    }
22   RP   OK      OQ    }     Stop
23   TP   SP      OQ    }
24   RP   OL      OQ1   }     End
25   TP   EU      OQ    }
26   RP   OM      OQ    }     Exit
27   TP   EZ      OQ    }
30   RP   ON      OQ    }     Start
31   TP   ST      OQ    }
32   RP   OO      OR1   }     Equation
33   TP   YA      OR    }
34   MJ   O       FW35  }
35   RP   OP      OQ1   }     Pseudo op. heading
36   TP   HE      OQ    }
     CA   FW37
```

Translation Control Constants

	IA	KB		
0	27	34473	05065	D I M E N S
1	65	66245	46677	S T A R T
2	70	24547	37777	V A R Y
3	26	51475	26766	C O M P U T
4	54	30242	77777	R E A D
5	46	34656	67777	L I S T
6	66	73523	07777	T Y P E
7	46	34656	67777	L I S T
10	52	54345	06677	P R I N T
11	34	31777	77777	I F
12	46	34656	67777	L I S T
13	54	30656	74730	R E S U M E
14	44	67475	27777	J U M P
15	65	66515	27777	S T O P
16	30	50277	77777	E N D
17	30	72346	67777	E X I T
20	52	51717	77777	P O W
21	52	51717	77777	P O W
22	50	51667	77777	N O T
23	0	0	1	
24	0	0	27000	
25	0	0	22000	
26	0	0	26000	
27	30	53672	46650	E Q U A T N
30	0	0	70000	
31	0	0	40000	
32	40	0	0	
33	66	24523	07777	T A P E
34	71	34663	37777	W I T H
35	66	33305	07777	T H E N
36	24	50277	77777	A N D
	CA	KB37		

(d) Get Next Sentence

```
     IA   GS
 0   MJ   0       (30000)
 1   TP   KA12    UZ3          Clear no. errors/sentence
 2   TP   GN2     A       }
 3   RP   20006   GS7     }    Not special case → GS7
 4   EJ   KA34    GS5     }    Special case ↓ beg. of blockette
 5   RA   GN2     KA31         Address + 1
 6   MJ   0       GS15
 7   TP   KA15    A       }    In last blkette → GR
10   TJ   GN2     GR      }    No ↓
11   SP   GN6     17      }    Address in GN2 = 2nd word of blkette.
12   AT   VD3     GS13    }        TP   KA1 + GN2
13   0    0       0       }
14   RA   GN6     KA16         Blkette count + 1
15   TP   KA12    GN3          Clear shift
16   TP   KA13    GN5          Char. count = 6
17   TU   GN2     GS21    }
20   RS   GS21    KA14    }    Line no. → A
21   TP   (30000) A       }
22   EJ   KA7     GU           Δ's → GU       no ↓
23   TP   KA17    Q       }
24   QT   A       A       }    Z's → CT34
25   EJ   KA10    CT34    }                   no ↓
26   TU   GS21    GS27    }
27   TP   (30000) LN4     }    Standardize line no.
30   RJ   LN2     LN1     }
31   TP   KA24    WL           Word count = 4
32   TP   LN3     WL1          Line no. → WL1
33   TP   KA12    EW4          Clear print ind. in EW.
34   RJ   GN      GN1          Get char.
35   MJ   0       GS           Exit
     CA   GS36

     IA   GR
 0   TP   KA6     TH3     }    Read 1 blk of tape.
 1   RJ   TH2     TH      }
 2   RA   13      KA16         Count blks.
 3   RP   10004   GR5     }
 4   TP   KA12    GN3     }    Set GN
 5   TP   KA      GN2     }
 6   MJ   0       GS16
     CA   GR7
```

```
     IA   GX                      Δ's within line
0    RA   GU2      KA14  }       Check 19(10) words for Δ's
1    IJ   VD2      GU2   }
2    MJ   0        GS2           → Get next sentence
     CA   GX3

     IA   GU                      Δ's as line #
0    TP   KA33     VD2           Set index
1    TU   GN2      GU2   )
2    TP   (30000)  A     }       Δ's → GX
3    EJ   KA7      GX    )       No ↓
4    RJ   UZ       UZ1           Set error
5    TP   WL1      FH21  )
6    TP   FH11     UP3   }       F14
7    RJ   UP2      UP    )
10   TP   KA7      WL1           Δ's → line #
11   TP   KA24     WL            Count = 4
12   MJ   0        GS33          → Exit
     CA   GU13
```

(e) Get Next Character

```
                IA    GN
          0     MJ    0       (30000)      Exit
          1     MJ    0        GN7
          2     0     (0)     (0)          Address in buffer
          3     0     0       (0)          Shift
          4     0     0       (0)          Last character
          5     0     0       (0)          Character count
          6     0     0       (0)          Blockette count
          7     TP    KA26     A           119(10) ⟶ A

         10     TJ    GN5     GN21         Char. count ≧ 120 → GN21
         11     TP    KA32    A            35(10) ⟶ A

         12     TJ    GN3     GN42         Shift ≧ 36(10) → GN42
         13     RA    GN3     KA13         Shift + 6
         14     RA    GN5     KA16         Char. count + 1
         15     TU    GN2     GN16    }    Shift word by 6
         16     LQ    (30000) 6       }
         17     QT    KA30    GN4          Char. ⟶ A ⟶ GN4
         20     MJ    0       GN           Exit
Char.    21     TP    KA24    A            4 ⟶ A
Count≧   22     TJ    GN6     GN45         Blkette ≧ 5 ⟶ GN45
120      23     RA    GN6     KA16         Blkette count + 1
         24     RA    GN2     KA31         Address + 1
         25     TP    KA13    GN5          Char. count = 6
         26     TP    KA12    GN3          Clear shift
         27     TU    GN2     GN30    }    Line no. ⟶ A
         30     TP    (30000) A       }
         31     EJ    KA7     GN40         No line nos. ⟶ GN40    yes ↓
         32     RJ    WA      WA2     )
         33     TP    FI4     UP3     }    F17
         34     RJ    UP2     UP      )
         35     TP    KA12    UZ3          Clear errors/sentence
         36     RA    UZ2     KA16         Errors + 1
         37     MJ    0       CT6          ⟶ Control
No.      40     RA    GN2     KA31         Address + 1
Line     41     MJ    0       GN13
Nos.     42     TP    KA13    GN3          Shift = 6
Shift≧   43     RA    GN2     KA31         Address + 1
36       44     MJ    0       GN14         ⟶ Get char.
block-   45     TP    KA12    GN6
ette≧    46     TP    KA6     TH3     }    Read 1 blk. of raw data
5        47     RJ    TH2     TH      }
         50     TP    KA34    GN2          Address = 0 BF  BF
         51     RA    13      KA16         Count blks
         52     MJ    0       GN25         ⟶ Return
                CA    GN53
```

Constants
Get next sentence and
Get next character

	IA	KA0		
0	0	BF1	BF1	2nd address of 1st blockette
1	0	BF25	BF25	2nd address of 2nd blockette
2	0	BF51	BF51	2nd address of 3rd blockette
3	0	BF75	BF75	2nd address of 4th blockette
4	0	BF121	BF121	2nd address of 5th blockette
5	0	BF145	BF145	2nd address of 6th blockette
6	50	00105	BF0	Read 1 blk. of corr. tape
7	01	01010	10101	Δ's
10	00	74747	47400	0 Z Z Z Z 0
11	0	0	2	
12	0	0	0	Zero
13	0	0	6	
14	0	1	0	
15	0	BF144	BF144	
16	0	0	1	
17	00	77777	77700	Mask
20	0	0	4	Count } END
21	74	74747	47474	Z's } string-out
22	30	50277	77777	Sentinel END } corr.
23	0	0	23000	C.W. }
24	0	0	4	
25	0	23	23	
26	0	0	167	119_{10} changed from 120_{10}
27	0	0	44	36_{10}
30	0	0	77	Mask
31	0	1	1	
32	0	0	43	Changed from 0 BF0 BF0
33	0	0	22	
34	0	BF0	BF0	
35	0	BF24	BF24	
36	0	BF50	BF50	
37	0	BF74	BF74	
40	0	BF120	BF120	
41	0	BF144	BF144	
	CA	KA42		

(f) Get Next Symbol

```
     IA    SY
 0   MJ    0       (30000)      Exit
 1   MJ    0        SY15        Start
 2   0     0        0      }
 3   0     0        0      }    Symbol
 4   0     0        0      }
 5   0     0        0           No. of characters
 6   0     0        0           No. of decimal points
 7   0     0        0           1st char. a letter
10   0     0        0           1st char. I, J, K, L, M
11   0     0        0           If a constant
12   0     0        0           If symbol contains a letter
13   0     0        0           If symbol is superscript
14   0     0        0           Space indicator
15   RP    30013    SY17   }    Sym --> XZ2
16   TP    SY2      SZ2    }
17   RP    10013    SY21   }    Clear output
20   TP    KK54     SY2    }
21   TP    GN4      A           Char. --> A
22   RJ    DS       DS1         Delete spaces
23   EJ    KK       XM          Period --> XM
24   EJ    KK1      UD          Upper decimal pt. --> UD
25   RP    20012    SY27   }    Upper digit --> UD1
26   EJ    KK2      UD1    }
27   EJ    KK14     UM     }    Upper - or / --> UM
30   EJ    KK15     UM     }
31   RP    20032    SY33   }    Letter --> LL
32   EJ    KK16     LL     }        No ↓
33   TP    KK50     A      }    Digit --> DG
34   TJ    GN4      DG     }        No ↓
35   RJ    BS       BS1         Build sym
36   RJ    FS       FS1         Fill sym
37   RJ    GN       GN1         Get character
40   TP    SY2      A           Sym --> A
41   MJ    0        SY          Exit
     CA    SY42
```

Digit

```
     IA    DG
 0   TP    KK51     A      }    Not a digit --> return
 1   TJ    GN4      SY35   }    Yes ↓
 2   TP    KK57     SY11        Set constant
 3   RJ    RL       RL1         Get rest of lower
 4   MJ    0        SY40        Exit
     CA    DG5
```

Lower Period or Point

```
     IA   XM0
 0   TP   SY14    Q       }     Δ ind. set → XM
 1   QJ   XM2     XM11    }
 2   TP   KK55    SY2     }     Δ. → output
 3   TP   KK56    SY3     }
 4   TP   KK56    SY4     }
 5   TP   KK50    SY5     }
 6   TP   KK54    SY14    }     Rest of output
 7   TP   KK53    SY6     }
10   MJ   0       SY40          Exit

11   TP   KK57    SY11          Set constant
12   RA   SY6     KK53          Pts + 1
13   RJ   RL0     RL1           Get rest of lower
14   MJ   0       SY40          Exit
     CA   XM15
```

Upper Point

```
     IA   UD
 0   RA   SY6     KK53          Pts + 1
 1   TP   KK57    SY11          Set constant
 2   RJ   RU0     RU1           Rest of upper
 3   TP   KK57    SY13          Set upper
 4   MJ   0       SY40          Exit
     CA   UD5
```

Upper - and slash

```
     IA   UM
 0   TP   KK57    SY13          Set upper
 1   MJ   0       SY35
     CA   UM2
```

Letter

```
     IA   LL
 0   TP   KK57     SY7          Set 1st char. a letter
 1   TP   KK57    SY12          Set letter
 2   RP   20005   LL5     }
 3   EJ   KK26    LL4     }     I J K L M ↓ no → LL5
 4   TP   KK57    SY10          Set IJ...bit
 5   RJ   RL      RL1           Get rest of lower
 6   MJ   0       SY40          Exit
     CA   LL7
```

Get rest of lower Symbol

```
     IA    RL0
 0   MJ    0        (30000)      Exit
 1   RJ    BS0      BS1          Build symbol
 2   TP    KK75     A       }    No. char.> 17 --> RL25
 3   TJ    SY5      RL25    }    < 17 ↓
 4   RJ    GN0      GN1
 5   RP    20015    RL7     }    Upper case --> RL23
 6   EJ    KK1      RL23    }    No ↓
 7   RP    20015    RL11    }    Other separator --> RL23
10   EJ    KK60     RL23    }    No ↓
11   EJ    KK0      RL21         . --> RL21
12   TP    KK51     A            14 --> A
13   TJ    GN4      RL17         Not a digit --> RL17
14   TP    GN4      A            3 --> A
15   TJ    KK104    RL17         Not a digit --> RL17  digit ↓
16   MJ    0        RL1
17   TP    KK57     SY12         Set a letter
20   MJ    0        RL1
21   RA    SY6      KK53         .    No. Pts + 1
22   MJ    0        RL1
23   RJ    FS0      FS1          Fill symbol
24   MJ    0        RL0          Exit
25   RJ    GN0      GNI          Get char.
26   RP    20015    RL30    }
27   EJ    KK1      RL23    }    Separator --> RL23
30   RP    20015    RL25    }    No. --> RL25
31   EJ    KK60     RL23    }
     CA    RL60
```

Get rest of upper symbol

```
     IA   RU0
 0   MJ   0       (30000)      Exit
 1   RJ   BS0     BS1          Build symbol
 2   TP   KK75    A       }    Char. count > 17 ⟶ RU12
 3   TJ   SY5     RU12    }                  no ↓
 4   RJ   GN0     GN1          Get char.
 5   EJ   KK1     RU15         Upper . ⟶ RU15
 6   RP   20012   RU10    }    Upper digit ⟶ RU1
 7   EJ   KK2     RU1     }
10   RJ   FS0     FS1          Fill symbol
11   MJ   0       RU0          Exit
12   RJ   GN0     GN1          Get char.
13   RP   20012   RU10
14   EJ   KK1     RU12
15   RA   SY6     KK53         Pts. + 1
16   MJ   0       RU1
     CA   RU17
```

(g) Build Symbol

```
     IA   BS0
 0   MJ   0       (30000)      Exit
 1   MJ   0       BS4          Start
 2   0    0       (0)          Shift
 3   0    (SY2)   (SY2)        Address of word SY2 - SY4
 4   TP   SY5     A       }    Char. count = 6 ⟶ BS17
 5   EJ   KK76    BS17    }
 6   EJ   KK77    BS22         = 12₁₀ ⟶ BS22
 7   TU   BS3     BS10    }    Left shift by 6
10   LQ   (30000) 6       }
11   TP   KK52    Q            Mask ⟶ Q
12   TV   BS3     BS13    }    Char. ⟶ sym.
13   QS   GN4     (30000) }
14   RA   BS2     KK76         Shift + 6
15   RA   SY5     KK53         Char. count + 1
16   MJ   0       BS           Exit
17   TP   KK100   BS3          Set for 2nd word
20   TP   KK54    BS2          Clear shift
21   MJ   0       BS7          Return
22   TP   KK101   BS3          Set for 3rd word
23   TP   KK54    BS2          Clear shift
24   MJ   0       BS7          Return
     CA   BS25
```

(h) Fill Symbol

```
    IA   FS0
 0  MJ   0       (30000)      Exit
 1  TP   BS2     A            Start
 2  EJ   KK102   FS13         Shift = 36 → FS13
 3  TU   BS3     FS4     }    Shift word by 6
 4  LQ   (30000) 6       }
 5  TP   KK52    Q            Mask → Q
 6  TV   BS3     FS7     }    77 fill
 7  QS   KK52    (30000) }
10  RA   BS2     KK76         Shift + 6
11  EJ   KK102   FS24         Shift = 36 → FS24
12  MJ   0       FS4          Return
13  TP   BS3     A
14  EJ   KK103   FS21         1st word →
15  EJ   KK100   FS22         2nd word →
16  TP   KK54    BS2          Clear shift
17  TP   KK103   BS3          Set for 1st word
20  MJ   0       FS0          Exit
21  TP   KK56    SY3          Fill 2nd word
22  TP   KK56    SY4          Fill 3rd word
23  MJ   0       FS16
24  TP   BS3     A       }    3rd word → FS16
25  EJ   KK101   FS16    }
26  MJ   0       FS14         No → FS14
    CA   FS27
```

(i) Delete Spaces

```
    IA   DS
 0  MJ   0       (30000)      Exit
 1  EJ   CJ13    DS3          Δ → DS3
 2  MJ   0       DS           No → exit
 3  TP   KK57    SY14         Set Δ ind.
 4  RJ   GN      GN1          Get next char.
 5  EJ   KK60    DS4          Δ → GN    no ↓
 6  MJ   0       DS           Exit
    CA   DS7
```

(j) Constants

SY, BS, FS

	IA	KKO		
0	0	0	22	.
1	0	0	62	$
2	0	0	60	0
3	0	0	61	1
4	0	0	40	2
5	0	0	20	3
6	0	0	41	4
7	0	0	35	5
10	0	0	55	6
11	0	0	75	7
12	0	0	36	8
13	0	0	57	9
14	0	0	00	-
15	0	0	15	/
16	0	0	24	A
17	0	0	25	B
20	0	0	26	C
21	0	0	27	D
22	0	0	30	E
23	0	0	31	F
24	0	0	32	G
25	0	0	33	H
26	0	0	34	I
27	0	0	44	J
30	0	0	45	K
31	0	0	46	L
32	0	0	47	M
33	0	0	50	N
34	0	0	51	O
35	0	0	52	P
36	0	0	53	Q
37	0	0	54	R
40	0	0	65	S
41	0	0	66	T
42	0	0	67	U
43	0	0	70	V
44	0	0	71	W
45	0	0	72	X
46	0	0	73	Y
47	0	0	74	Z
50	0	0	02	
51	0	0	14	
52	0	0	77	

```
 53    0     0       1
 54    0     0       0
 55    01    22777   77777       Δ .
 56    77    77777   77777
 57    40    0       0
 60    0     0       01          Δ
 61    0     0       63          +
 62    0     0       02          -
 63    0     0       56          *
 64    0     0       64          /
 65    0     0       23          ;
 66    0     0       21          ,
 67    0     0       42          |
 70    0     0       76          =
 71    0     0       16          >
 72    0     0       37          <
 73    0     0       17          (
 74    0     0       43          )
 75    0     0       21          17(10)
 76    0     0       6
 77    0     0       14          12(10)
100    0     SY3     SY3
101    0     SY4     SY4
102    0     0       44
103    0     SY2     SY2
104    0     0       03
       CA    KK105
```

(k) Send File Back to CB list

```
    IA   TD
0   MJ   0        (30000)        Exit
1   LQ   TA31     25
2   TV   Q        TD6
3   TP   TB6      Q
4   QS   TA47     TD5
5   RP   (30000)  TD       }     Exit
6   TP   TA2      (30000)  }     File → list
    CA   TD7
```

(l) Add File to CB List

```
    IA   TE
0   MJ   0        (30000)        Exit
1   MJ   0        TE4            Start
2   0    (30000)  (30000)        Next add. in CB list
3   0    0        0              Ind. of prev. point
4   SP   TE2      0        }     Increase address
5   SA   TF       0        }
6   TJ   TB4      TG             OK → TG    no ↓
7   TP   TE3      Q        }     Prev. print → exit
10  QJ   TE       TE11     }     No ↓
11  RJ   UZ       UZ1      )
12  TP   WL1      FH46     }     F15
13  TP   FH22     UP3      }
14  RJ   UP2      UP       )
15  TP   TB5      TE3            Set ind.
16  MJ   0        TE             Exit
    CA   TE17

    IA   TG
0   TV   TE2      TG7            Set 1st address
1   TP   TB6      Q        }     Set repeat
2   QS   TF       TG6      }
3   TP   CB       A        }     Set CB for search
4   QA   TF       CB       }
5   RA   TE2      TF             Increase address
6   RP   (30000)  TE       }     Exit
7   TP   TF1      (30000)  }     Send file to list
    CA   TG10
```

(m) Get File From CB List

```
RJ  TA  TA1
MJ  0   0       not in list
MJ  0   0       in list
```

```
    IA   TA
 0  MJ   0       (30000)      Exit
 1  MJ   0       TA32
 2  CA   TA2                  Drum add. in u if from dim.
 3                             XS3 of sym
 4                (     )     CW
 5                              )
 :                              |
 :                              |  Format etc.
 :                              }  TA5 - TA31
 :                              |
31                              )
    IA   TA32                  Store address
32  TU   CB      TA34          Set up repeat
33  TP   SY2     A             Sym → A
34  RP   (20000) TA             )  Not in list → exit
35  EJ   CB1     TA36            }  In list ↓
36  LQ   Q       17              )
37  TP   CB      A               |
40  ST   Q       A               }  Set up address
41  AT   TB2     TA46            |
42  RS   TA46    TB3             )
43  TU   TA46    TA31          Store address
44  RA   TA      TB1           Set exit for "in list"
45  RP   30004   TA50
46  0    0       0             TP   CB+     TA2
47  0    (0)     0             No. words in file
50  TP   TB7     TA47          No. words = 4
51  TP   TA4     Q               )
52  QT   TB10    A               }
53  EJ   TB10    TA57            )  CW = 77XXX → TA57
54  QT   TB11    A               )  CW = 4XXXX → TA64
55  EJ   TB12    TA64            }
56  MJ   0       TA            Exit
57  TU   TA46    TA63          77XXX
60  RA   TA63    TB7           Set add. from CB list.
61  RA   TA47    TB14          Count + 2
62  RP   30002   TA             )  → Exit
63  TP   (30000) TA6            }  2 more wds → output
64  TU   TA46    TA70          4XXXX
65  RA   TA70    TB7
66  RA   TA47    TB13          Count + 19(10)
67  RP   30023   TA              )
70  TP   (30000) TA6             }  19(10) more words → output
    CA   TA71
```

Constants

	IA	TB		
0	RP	0	TA23	
1	0	0	1	
2	TP	CB	TA2	
3	0	1	0	
4	0	CB6000	CB6000	Max. of CB list + 1
5	40	0	0	Ind.
6	0	7777	0	Mask
7	0	4	0	
10	0	0	77000	
11	0	0	70000	
12	0	0	40000	
13	0	23	0	
14	0	2	0	
	CA	TB15		

(n) Get CW From Dummy P.O. List

Y	RJ	TS	TS1
Y +	1	(not in list)	
Y +	2	(in list)	

	IA	TS			
0	MJ	0	(30000)		Exit
1	MJ	0	TS5		Start
2	0	0	0		XS3
3	0	0	(0)		CW
4	0	0	0		Indicates within P.O.
5	TP	TS4	Q	}	Not in P.O. → exit
6	QJ	TS7	TS	}	in ↓
7	TP	SY2	A		Sym → A
10	RP	20132	TS	}	Not in list → exit
11	EJ	DP	TS12	}	in ↓
12	SP	TS21	0	}	
13	SS	Q	17	}	r - 1 → A_u
14	AT	TS22	TS17	}	
15	RA	TS	TS20		Set exit for in list
16	RP	30002	TS	}	Exit
17	0	0	0	}	File → output
20	0	0	1		
21	0	0	20131		J N - 1
22	TP	DP	TS2		
	CA	TS23			

Translation Variables

	IA	VB		
0	0	0	37700	Next P.O C.W.
1	77	77777	77776	Next 66, 65, 64 C.W.
2	77	77777	77776	Not used
3	77	77777	77776	Next Dummy in Equation
4	77	77777	77776	Next stm't C.W. 26, 27, 22
	CA	VB5		

	IA	VD		
0	0	0	0	No. of STARTS
1	0	0	0	EXIT indicator
2	0	0	0	Index
3	TP	KA1	GN2	
	CA	VD4		

Explanation of Temporary Build File Area

TF	0	0	(0)	(0)	No. of words in file
	1	0	0	0	XS3 of symbol
	2	0	0	(0)	C.W.
		Words TF3 - TF27		}	Formats, etc.

(o) Call Word ⟶ Translation List

```
     IA    EW
 0   MJ    0        (30000)      Exit
 1   MJ    0         EW5         Start
 2   0     0        (0     )     C.W.
 3   0    (30000)  (30000)      Address of last C.W.
 4   0     0         0           Ind. one print-out
 5   RA    EW3      CJ7         Add + 1
 6   TJ    CJ10     EW17        OK ⟶ EW17  no ↓
 7   TP    EW4      Q       }   Prev. print ⟶ EW15
10   QJ    EW15     EW11    }   No ↓
11   RJ    WA       WA1     )
12   TP    FI       UP3     }   F16
13   RJ    UP2      UP      )
14   TP    CJ2      EW4         Set bit
15   RS    EW3      CJ7         Add - 1
16   MJ    0        EW          Exit
17   TV    EW3      EW20    )   CW ⟶ string
20   TP    EW2     (30000)  }
21   MJ    0        EW      )   Exit
     CA    EW22
```

(p) Increase 66, 65 64 Call Word Counter

```
     IA    TK
 0   MJ    0        (30000)      Exit
 1   MJ    0         TK3         Start
 2   0     0         0           Indicator
 3   TP    VB1      A       }   OK ⟶ TK16
 4   TJ    CJ3      TK16    }   No ↓
 5   TP TK2         Q       }   Prev. Print ⟶ exit
 6   QJ    TK       TK7     }   No ↓
 7   RJ    UZ       UZ1     )   Error
10   TP WL1         FD43    )
11   TP    SY2      FD36    }   F6
12   TP    FD16     UP3     )
13   RJ    UP2      UP      )
14   TP    CJ2      TK2         Set ind.
15   MJ    0        TK          ⟶ exit
16   RA    VB1      CJ1         CW + 1
17   MJ    0        TK          Exit
     CA    TK20
```

(q) Increase Sentence C.W. Counter (Output-A or VB4,)
26, 27, 22

```
     IA   XJ
 0   MJ   0       (30000)      Exit
 1   MJ   0        XJ4         Start
 2   0    0        0           Indicator
 3   0    0        777         Constant
 4   TP   VB4      A           OK --> XJ17
 5   TJ   XJ3      XJ17        No  v
 6   TP   XJ2      Q           Prev. print --> exit
 7   QJ   XJ       XJ10        No  v
10   RJ   UZ       UZ1         Set error
11   TP   WL1      FD15
12   TP   FD       UP3         F5
13   RJ   UP2      UP
14   TP   VB4      A           CW --> A
15   TP   CJ2      XJ2         Set ind.
16   MJ   0        XJ          Exit
17   RA   VB4      CJ1         CW + 1 --> A
20   MJ   0        XJ          Exit
     CA   XJ21
```

(r) Print Error Heading

RJ WA WA1 Set Error
WA2 Don't Set Error

```
     IA   WA
 0   MJ   0       (30000)      Exit
 1   RJ   UZ       UZ1        Set error
 2   TP   WL1      WB3        Line number
 3   TP   WL2      A
 4   TP   A        WB5        Sentence type ⟶A ⟶file
 5   TP   WB10     WB6        Fill
 6   EJ   WB15     WA15       E  Q  U  A  T  N
 7   EJ   WB11     WA16       E  Q  U  A  T  I
10   EJ   WB16     WA20       C  O  M  P  U  T
11   EJ   WB17     WA22       D  I  M  E  N  S
12   TP   WB       UP3     }  Print heading
13   RJ   UP2      UP      }
14   MJ   0        WA         Exit
15   TP   WB11     WB5     }  Load  E  Q  U  A  T  I
16   TP   WB12     WB6     }        0  N  77 77 77 77
17   MJ   0        WA12
20   TP   WB13     WB6           E  77 77 77 77 77
21   MJ   0        WA12
22   TP   WB14     WB6           I  O  N  77 77 77
23   MJ   0        WA12
     CA   WA24
```

Constants

```
     IA   WB
 0   0    WB1      7
 1   65   30506    63050      S  E  N  T  E  N
 2   26   30010    17777      C  E  Δ  Δ  77 77
 3   0    0        0          Line #
 4   01   01011    77777      Δ  Δ  Δ  (  77 77
 5   0    0        0
 6   0    0        0
 7   43   01017    77777      )  Δ  Δ 77 77 77
10   77   77777    77777         77  fill
11   30   53672    46634      E  Q  U  A  T  I
12   51   50777    77777      0  N  77 77 77 77
13   30   77777    77777      E  77 77 77 77 77
14   34   51507    77777      I  O  N  77 77 77
15   30   53672    46650      E  Q  U  A  T  N
16   26   51475    26766      C  O  M  P  U  T
17   27   34473    05065      D  I  M  E  N  S
     CA   WB20
```

(s) Error Routine

```
     IA    UZ
 0   MJ    0        (30000)      Exit
 1   MJ    0        UZ4          Start
 2   0     0        0            No. of errors for entire project
 3   0     0        0            No. of errors this sentence
 4   RA    UZ2      CJ1          Errors + 1
 5   TJ    CJ14     UZ13         O.K. ⟶ UZ13  no ↓
 6   RJ    RW       RW1          Rewind tape
 7   TP    WL1      FB37    )
10   TP    FB15     UP3     }    F4
11   RJ    UP2      UP      )
12   MS    0        UZ           Stop on exit
13   RA    UZ3      CJ1
14   TJ    CJ15     UZ           Too many errors ↓ no⟶exit
15   RJ    WA       WA2     )
16   TP    FB       UP3     }    F3
17   RJ    UP2      UP      )
20   TP    UW       UZ3          Clear errors/sentence
21   MJ    0        CT           ⟶ Control
     CA    UZ22
```

(t) Check Floating Point Constant

```
     IA   RB
 0   MJ   0       (30000)      Exit
 1   TP   SY5     A            Start
 2   TJ   CJ4     RB10         No. of char. > 12  ↓ no ⟶ RB10
 3   RJ   WA      WA1
 4   RP   30003   RB6
 5   TP   SY2     FE14         F7
 6   TP   FE      UP3
 7   RJ   UP2     UP
10   TP   SY6     A
11   TJ   CJ5     RB17         No. pts > 1  ↓ no ⟶ RB17
12   RJ   WA      WA1
13   RP   30003   RB15
14   TP   SY2     FE27         F8
15   TP   FE17    UP3
16   RJ   UP2     UP
17   TP   SY12    Q
20   QJ   RB21    RB           Contains a letter ↓ no → exit
21   RJ   WA      WA1
22   RP   30003   RB24
23   TP   SY2     FF7          F9
24   TP   FF      UP3
25   RJ   UP2     UP
26   MJ   0       RB           Exit
     CA   RB27
```

(u) Check Fixed-Point Constant

```
     IA    RD
 0   MJ    0        (30000)      Exit
 1   TP    SY5      A          }
 2   TJ    CJ6      RD10       }  No. of char. > 6 ↓ no → RD10
 3   RJ    WA       WA1        }
 4   RP    30003    RD6        }
 5   TP    SY2      FF25       }  F10
 6   TP    FF12     UP3        }
 7   RJ    UP2      UP         }
10   TP    SY6      A          }
11   ZJ    RD12     RD16       }  No. pts. > 0 ↓  no → RD16
12   RJ    WA       WA1        }
13   TP    SY2      FG10       }
14   TP    FG       UP3        }  F11
15   RJ    UP2      UP         }
16   TP    SY12     Q          }
17   QJ    RD20     RD         }  Contains letter ↓ no → exit
20   RJ    WA       WA1        }
21   RP    30003    RD23       }
22   TP    SY2      FF7        }  F9
23   TP    FF       UP3        }
24   RJ    UP2      UP         }
25   MJ    0        RD             exit
     CA    RD26
```

(v) Check Variable Type Symbol

```
     IA   RH
 0   MJ   0       (30000)      Exit
 1   TP   SY5     A          }
 2   TJ   CJ6     RH10       }  No. of char. > 6 ↓ no → RH10
 3   RJ   WA      WA1        )
 4   RP   30003   RH6        )
 5   TP   SY2     FG24       }  F12
 6   TP   FG11    UP3        )
 7   RJ   UP2     UP         )
10   TP   SY6     A          }
11   ZJ   RH12    RH17       }  Contains a point ↓ no → exit
12   RJ   WA      WA1        )
13   TP   SY2     FH10       }  F13
14   TP   FH      UP3        )
15   RJ   UP2     UP         )
16   MJ   0       RH           Exit
17   TP   SY2     A          )
20   RP   20023   RH22       )
21   EJ   KB      RH24       }
22   RP   20004   RH         )
23   EJ   KB33    RH24       )  Key word ↓ no → exit
24   RJ   WA      WA1        )
25   TP   SY2     WB20       }  F19
26   TP   WB15    UP3        )
27   RJ   UP2     UP         )
30   MJ   0       RH           Exit
     CA   RH31
```

(w) Constants

```
     IA   CJ
 0   0    0       0            Zero
 1   0    0       1            One
 2   40   0       0            Indicator
 3   0    0       777
 4   0    0       15           RC0
 5   0    0       2            RC1
 6   0    0       7            RC2
 7   0    1       1
10   0    WL250   WL250        Limit of string
11   0    0       170
12   71   0       WL           Write @0 in & 128 lines/in
13   0    0       01           Δ
14   0    0       31           Errors/prog.
15   0    0       5            Errors/sentences
     CA   CJ16
```

(x) Print Text

```
    IA   UP
 0  MJ   0        UP4          Start
 1  0    30000    (30000)
 2  MJ   0        (30000)      Exit
 3  0    (30000)  (30000)      Par.
 4  TV   UW15     UQ           } Set up switch to UP
 5  TV   UQ3      UQ4          }
 6  TP   UW       UW20         )
 7  TV   UP3      UW20         ) Set up N counter
10  MP   UW3      UW20         )
11  TP   A        UW20         )
12  TP   UW13     US           } 6 char. word pick up
13  TU   UP3      US           }
14  PR   0        UW7          Shift up
15  TP   UP3      Q            Par. → Q
16  QJ   US       UP17         00 ↓ 40 → US
17  TP   UW4      UW21         Counter = 80(10)
20  RP   4        US           } 4 cr's
21  PR   0        UW11         }
    CA   UP22
```

```
    IA   UQ
 0  RJ   UQ       (30000)      DOWN          )
 1  RJ   UQ4      US25         Exit          )
 2  PR   0        UW7          Shift up      ) Up-down
 3  MJ   0        UQ5                        ) switch
 4  RJ   UQ4      (30000)      UP            )
 5  RJ   UQ       US25         Exit          )
 6  PR   0        UW10         Shift down    )
 7  MJ   0        UQ1
    CA   UQ10
```

```
    IA   US
 0  0    (30000)  (30000)      Six char. pick up
 1  RA   US       UW2          Modify pick up
 2  TP   UW5      UW22         Set up 5 count
 3  IJ   UW20     US6          N counter
 4  PR   0        UW7          Shift up
 5  MJ   0        UP2          Exit
 6  LQ   UW23     6            Unityper code → Q
 7  QT   UW6      UW16         → A → UW16
```

```
        10   EJ   UW1      US34          Δ ⟶ US34     No ↓
        11   EJ   UW6      US36          77 ⟶ US36
        12   TV   US37     US35          Test for suppress
        13   IJ   UW21     US21          80 counter
        14   TP   UW17     UW21          Char. count = 66(10)
        15   PR   0        UW11   }      CR. CR.
        16   PR   0        UW11   }
        17   RP   23       US12   }      19 spaces
        20   PR   0        UW12   }
        21   SP   UW16     17            XS3 ⟶ Au
        22   AT   UW14     US23   }
        23   0    (30000)  (30000)  }    Codes ⟶ A shift 44(8)
        24   SJ   UQ       UQ4
        25   LA   A        3      }      Codes ⟶ position
        26   SS   A        6      }
        27   PR   0        A      }      Print codes
        30   SS   A        6      }
        31   ZJ   US27     US32          Loop
        32   IJ   UW22     US3           5 count
        33   MJ   0        US            ⟶ new word
Δ's     34   TP   UW21     A      }      81st ↓ no ⟶ US12
        35   ZJ   US12     (US32) }      Or ⟶ US12
77's    36   TV   US17      US35)        Print  Δ's
        37   MJ   0         US32
             CA   US40
```

```
             IA   UW
        0    0    0         0            Zero
        1    0    0         1            1 in V and  Δ  code
        2    0    1         0            1 in u
        3    0    0         6            6
        4    0    0         120          80(10)
        5    0    0         5            5 counter
        6    0    0         77           Mask and 77 code
        7    0    0         47           Shift up
        10   0    0         57           Shift down
        11   0    0         45           Carriage return
        12   0    0         4            Space code
        13   TP   (30000)   UW23         Pick up dummy
        14   SP   FC        44           Flex pick up dummy
        15   0    0         UQ6          Set switch
        16   0    0         0            XS3 of char.
        17   0    0         75           61(10)
        20   0    30000     30000        N counter
        21   0    30000     30000        80 counter
        22   0    30000     30000        5 counter
        23   0    30000     30000        Work space
             CA   UW24
```

(y) Sentence CW to Reference List IZ

```
     IA   RA
 0   MJ   0       (30000)      Exit
 1   TP   WL1     A            Line no. ⟶ A
 2   TU   11      RA3
 3   RP   (20000) RA      }     Not in list ⟶ Exit
 4   EJ   IZ      RA5     }    In list ↓
 5   SN   Q       17
 6   SA   11      25           11 = 0  j  n   n
 7   LT   0       A
10   AT   RA13    RA11
11   (0   0       0)           Sentence CW ⟶ list
12   MJ   0       RA           ⟶ Exit
13   TP   WL3     IZ           Constant
     CA   RA14
```

(z) Rewind all tapes

```
     IA   RW
 0   MJ   0       (30000)      Exit
 1   MJ   0       RW10         Start
 2   20   1       0            Parameter
 3   20   1       0            Set up parameter
 4   0    1       0            Modifier
 5   0    0       0            Index
 6   0    0       4       }    Set index
 7   0    0       2       }
10   TP   RW3     RW2          Set parameter
11   TP   RW6     RW5          Set index
12   TP   TN      A       }    5 servos ⟶ RW15
13   ZJ   RW14    RW15    }    7 servos ⟶ RW14
14   RA   RW5     RW7          Increase index
15   TP   RW2     TH3     }    Rewind tapes
16   RJ   TH2     TH      }
17   RA   RW2     RW4          Change parameter
20   IJ   RW5     RW15         Return
21   MJ   0       RW           Exit
     CA   RW22
```

(aa) 1105 Tape Handler Regions

TH21	Control Region
EX70	Exit
RW73	Rewind
IA100	Ignore Address
WB101	Build Write Code
WW113	Write
RR125	Read Subroutine
RE143	Reread If Parity Error
RF206	Read Forward
RB214	Read Backward
RW222	Rewind Region (Redundant - Not Used)
MF227	Move Forward
MB240	Move Backward
PC242	Print
WE263	Constants
CC274	Constants
CE334	Constants
CF352	Constants
CD365	Constants
VV375	Temporaries
CR402	Check for Zero Blocks

(ab) 1105 Tape Handler

```
     IA    TH
 0   MJ    20000    TH4          Start - TCU2
 1   MJ    0        TH10         Start - TCU1
 2   MJ    0        (30000)      Exit
 3   0     30000    30000        Parameter
 4   MJ    50000    TH4          Test buffer 2
 5   TP    CD2      CD           Set TCU2
 6   EF    0        CC3          Bypass buffer 2
 7   MJ    0        TH13
10   MJ    40000    TH10         Test buffer 1
11   TP    CD1      CD           Set TCU1
12   EF    0        CC2          Bypass buffer 1
13   TP    CC27     Q       )
14   RP    10005    TH16    }    Set TCU bit in code words
15   QS    CD       CD3     )
16   EF    0        CD5          Set normal bias
17   TP    TH3      Q       )
20   QT    CE5      A       |    Servo no. VV2
21   LT    41       A       }    Servo no. and TCU bit →VV
22   TP    A        VV2     |
23   AT    CD       VV      )
24   TV    CE15     IA           Set ignore address
25   RJ    CR5      CR           →Test for zero blocks
26   QT    CC4      A       }    Op code →A
27   TP    A        A       )
30   EJ    CC5      RW           Rewind →RW
31   EJ    CC6      RW3          w/interlock →RW3
32   EJ    CC7      MF           Move fwd →MF
33   EJ    CC10     MB           Move bwd →MB
34   EJ    CC11     RF           Read fwd →RF
35   EJ    CC12     RB           Read bwd →RB
36   QT    CC27     A            "A" →A
37   EJ    CC31     WB           A = 2
40   EJ    CC32     WB2          "A" = 4
41   QT    CC30     A            B →A
42   EJ    CC33     WB4          B = 1
43   EJ    CC34     WB6          B = 2
44   EJ    CC35     WB10         B = 3
45   RA    VV       CC21         B = 0
46   MJ    0        WW           →Write
     CA    TH47
```

Exit

```
     IA   EX
 0   TV   CE15    IA          Set ignore
 1   MJ   0       TH2         Exit
 2   0    0       RE33
     CA   EX3
```

Build Write Code Words

```
     IA   WB
 0   RA   VV      CC25        A = 2
 1   MJ   0       TH41        ⟶ B sec.
 2   RA   VV      CC26        A = 4
 3   MJ   0       TH41        ⟶ B sec.
 4   RA   VV      CC22        B = 1
 5   MJ   0       WW
 6   RA   VV      CC23        B = 2
 7   MJ   0       WW
10   RA   VV      CC24        B = 3
11   MJ   0       WW
     CA   WB12
```

Write

```
     IA   WW
 0   RJ   RE5     RE1     }
 1   TP   CE11    Q       }   Set index
 2   QT   VV3     VV3     }
 3   TV   TH3     WW6         Set address
 4   EF   0       VV          Start writing
 5   RP   10170   WW7     }   Write block
 6   EW   10000   (30000) }
 7   RA   WW6     CE12        Add. + 170(8)
10   IJ   VV3     WW4         No. of blocks
11   MJ   0       EX          Exit
     CA   WW12
```

Read Subroutine

```
     IA    RR
 0   MJ    0        (30000)      Exit
 1   MJ    0        RR7          Normal entrance
 2   CC    RR3      CD2          Reread entrance
 3   0     30000    30000        RA (RS)  RR11  CC1
 4   CC    VV       CC36         Reverse direction
 5   CC    RR12     CD2          RA (RS)
 6   RJ    RR6      RR7          One shot exit
 7   TP    CC37     VV4          Set index
10   EF    0        VV           Start reading
11   ER    10000    (30000)      Read
12   0     (30000)  (30000)      RA (RS)  RR11  CC1
13   IJ    VV4      RR11         One block
14   ER    0        A            I O A ⟶ A
15   MJ    0        RR           Exit
     CA    RR16
```

Read Fwd (Bwd)

```
     IA    RE
 0   TV    TH3     RR11        Set address
 1   TP    TH3     Q          )
 2   QT    CE4     A          ) Index = No. of blocks - 1
 3   SS    CE6     17         )
 4   LT    0       VV3        )
 5   RJ    RE5     RE6         Exit
 6   RJ    RR      RR1         Read blk fwd (bwd) (normal)
 7   ZJ    RE10    RE41        Parity ↓ no →RE37
10   RJ    RR      RR2         Read bwd (fwd) (normal)
11   ZJ    RE12    RE32        Parity ↓ no →RE31
12   EF    0       CD6         Set high
13   RJ    RR      RR2         Read fwd (bwd) (high)
14   ZJ    RE15    RE40        Parity ↓ no →RE36
15   RJ    RR      RR2         Read bwd (fwd) (high)
16   ZJ    RE17    RE32        Parity ↓ no →RE31
17   EF    0       CD7         Set low
20   RJ    RR      RR2         Read fwd (bwd) (low)
21   ZJ    RE22    RE40        Parity ↓ no →RE36
22   RJ    RR      RR2         Read bwd (fwd) (low)
23   ZJ    RE24    RE32        Parity ↓ no →RE31
24   TP    CF      PC2        ) Print error
25   RJ    PC      PC1        )
26   EF    0       CD5         Set normal
27   RJ    RR6     RR2         Set to read fwd (bwd)
30   TV    EX2     IA          Set ignore address
31   MS    0       RE6         Stop for rereads
32   RJ    RR6     RR2         Reverse direction
33   TV    CE10    IA          Set ignore add.
34   TP    VV2     A          )
35   AT    VV1     A          ) Move fwd (bwd) 1 blk
36   EF    0       A          )
37   0     30000   30000       RA (RS)  RR11  CE12
40   EF    0       CD5         Set normal
41   IJ    VV3     RE6         No. blks.
42   MJ    0       EX
     CA    RE43
```

Read Fwd

```
     IA    RF
 0   RA    VV      CC17        Set read fwd →VV
 1   TP    CE1     RR3        )
 2   TP    CE1     RR12       ) Set for read fwd
 3   TP    CE2     RE37       )
 4   TP    CD3     VV1        )
 5   MJ    0       RE          →Read
     CA    RF6
```

Read Bwd

```
     IA    RB
0    RA    VV      CC20        Set read bwd→VV
1    TP    CE      RR3    )
2    TP    CE      RR12   }    Set for read bwd
3    TP    CE3     RE37   )
4    TP    CD4     VV1    )
5    MJ    0       RE          →Read
     CA    RB6
```

Rewinds

```
     IA    RW
0    RA    VV      CC13        Rewind
1    EF    0       VV
2    MJ    0       EX          Exit
3    RA    VV      CC14        Rewind w/interlock
4    MJ    0       RW1
     CA    RW5
```

Move Fwd

```
     IA    MF
0    TU    CE7     MF4         Set for fwd
1    RJ    RE5     RE1         Set index
2    TV    CE7     IA          Set ignore address
3    EF    0       CD7         Set low gain
4    TP    (30000) A           Set for fwd (bwd)
5    AT    VV      VV          Set EF word
6    EF    0       VV          Move 1 blk
7    IJ    VV3     MF6         n blks
10   MJ    0       EX          Exit
     CA    MF11
```

Move Bwd

```
     IA    MB
0    TU    CE10    MF4
1    MJ    0       MF1
     CA    MB2
```

Print Out

```
     IA   PC
 0   MJ   0       (30000)       Exit
 1   MJ   0        PC16         Start
 2   0   (30000)  (30000)       Par.= 0 address  n
 3   TP   PC2     Q        )
 4   QT   CE13    A45      }    Set index
 5   ST   CC1     PC20     )
 6   TU   PC2     PC7           Set address
 7   SP  (30000)  52       )
10   PR   0       A        }    Print one word
11   SS   A       6        )
12   ZJ   PC10    PC13     )
13   RA   PC7     CE14          Set for next word
14   IJ   PC20    PC7           Finished ↓ no → PC7
15   MJ   0       PC            Exit
16   RP   4       PC3           4 crs.
17   PR   0       PC4
20   0    0       0             Index
     CA   PC21

     IA   WE
 0   TP   WE3     PC2      }    "Can't ignore.  Rerun."
 1   RJ   PC      PC1      }
 2   MS   0       IA
 3   0    WE4     5
 4   47   16300   60603         ↑ C A N N O
 5   01   04141   30603         T Δ I G N O
 6   12   20574   20404         R E ↓ . Δ Δ
 7   47   12201   23406         ↑ R E R U N
10   57   42040   40404         ↓ . Δ Δ Δ Δ
     CA   WE11

     IA   CC
 0   0    0       0             Zero
 1   0    0       1             One
 2   0    10000   04000         Bypass buff.1
 3   0    20000   04000         Bypass buff.2
 4   70   0       0             Mask off op. code
 5   10   0       0             Rewind
 6   20   0       0             Rewind w/interlock
 7   30   0       0             Move fwd
10   40   0       0             Move bwd
11   50   0       0             Read fwd
12   60   0       0             Read bwd
13   0    200     0             Rewind
14   0    400     0             Rewind w/interlock
```

15	0	4	1	Move fwd	
16	0	14	1	Move bwd	
17	0	602	0	Read fwd	
20	0	612	0	Read bwd	
21	0	606	0	Write low and stop	B = 0 1" blk
22	0	616	0	Write high	B = 1 1" blk
23	0	706	0	Write low	B = 2 2.4 blk
24	0	716	0	Write high	B = 3 2.4 blk
25	0	20	0	A = 2	
26	0	40	0	A = 4	
27	07	0	0	Mask off A	
30	0	70000	0	Mask off B	
31	02	0	0	A = 2	
32	04	0	0	A = 4	
33	0	10000	0	B = 1	
34	0	20000	0	B = 2	
35	0	30000	0	B = 3	
36	0	10	0	To reverse read	
37	0	0	167	Set index	
	CA	CC40			

	IA	CE			
0	RS	RR11	CC1	}	Set RR3 & RR12 fwd
1	RA	RR11	CC1	}	bwd
2	RA	RR11	CE12	}	Set RE34 fwd
3	RS	RR11	CE12	}	bwd
4	7	77700	0		Mask no. of blocks (read)
5	0	77	0		Mask off servo no.
6	0	100	0		Subt. 1 from no. of blks.
7	0	CC15	MF7		
10	0	CC16	RE37		
11	0	0	77		Mask
12	0	0	170		120_{10}
13	0	0	77777		v mask
14	0	1	0		1 in u
15	0	0	WE		
	CA	CE16			

	IA	CF		
0	0	CF1	12	
1	47	12203	02204	↑ R E A D Δ
2	20	12120	31204	E R R O R Δ
3	46	15301	21401	(P A R I T
4	25	04031	20424	Y Δ O R Δ S
5	15	12031	63620	P R O C K E
6	01	42574	24704	T) ↓ . ↑ Δ
7	04	24013	01201	Δ S T A R T
10	04	26031	20412	Δ F O R Δ R
11	20	12203	02224	E R E A D S
12	57	42040	40404	↓ . Δ Δ Δ Δ
	CA	CF13		

	IA	CD		
0	0	0	0	TCU bit
1	1	0	0	Set TCU bit to 1
2	2	0	0	Set TCU bit to 2
3	0	4	1	Move fwd 1 blk
4	0	14	1	Move bwd 1 blk
5	0	1	50000	Set normal
6	0	1	60000	Set high
7	0	1	70000	Set low
	CA	CD10		

Variables

```
     IA   VV
0    0    0       0                    EF code word
1    0    0       0                    Move fwd (bwd) 1 blk
2    0    0       0                    Tape no.
3    0    0       0                    Index (n blks)
4    0    0       0                    Index 1 blk
     CA   VV5
```

Check for Zero Blocks

```
     IA   CR
0    QT   CR7     A
1    TP   A       A
2    EJ   CC4     EX
3    QT   CR10    A
4    TP   A       A
5    RP   20004   (30000)
6    EJ   CC7     EX
7    70   07700   0
10   77   77700   0
     CA   CR11
```

(ac) Regions for 1103A Tape Handler

TH21	Control Region
RR55	Read Subroutine
MB75	Move Backward
IA100	Ignore Address
WB101	Build Write Code
WW113	Write
RE125	Set Index
RA134	Reread If Parity Error
RF173	Read Forward
RB203	Read Backward
RW213	Rewind
MF220	Move Forward
PC231	Print
WE252	Constants
CF263	Constants
CC276	Constants
CE336	Constants
CD361	Constants
VV371	Temporaries
CR377	Check for Zero Blocks

```
     IA    TH
 0   MJ    0       TH4          Start
 1   TV    CE15    IA           Set ignore address
 2   MJ    0       (30000)      Exit
 3   0     30000   30000        Parameter
 4   MJ    0       TH5
 5   TP    TH3     Q          )
 6   QT    CE5     A          |  Servo no. --> VV2
 7   LT    41      A          }  Servo no. & tape bit --> VV
10   TP    A       VV2        |
11   AT    CD      VV         )
12   TV    CE15    IA           Set ignore address
13   RJ    CR10    CR           --> Test for zero blocks
14   TP    A       A            Op. code -->A
15   EJ    CC5     RW           Rewind --> RW
16   EJ    CC6     RW3          W/interlock --> RW3
17   EJ    CC7     MF           Move fwd --> MF
20   EJ    CC10    MB           Move bwd --> MB
21   EJ    CC11    RF           Read fwd --> RF
22   EJ    CC12    RB           Read bwd --> RB
23   QT    CC27    A            "A" --> A
24   EJ    CC31    WB           A = 2
25   EJ    CC32    WB2          A = 4
26   QT    CC30    A            "B" --> A
27   EJ    CC33    WB4          B = 1
30   EJ    CC34    WB6          B = 2
31   EJ    CC35    WB10         B = 3
32   RA    VV      CC21         B = 0
33   MJ    0       WW           --> Write
     CA    TH34
```

Build Write Codes

```
     IA    WB
 0   RA    VV      CC25          "A" = 2
 1   MJ    0       TH26          ⟶ B sec.
 2   RA    VV      CC26          A = 4
 3   MJ    0       TH26          ⟶ B sec.
 4   RA    VV      CC22          B = 1
 5   MJ    0       WW
 6   RA    VV      CC23          B = 2
 7   MJ    0       WW
10   RA    VV      CC24          B = 3
11   MJ    0       WW
     CA    WB12
```

Write

```
     IA    WW
 0   RJ    RE      RE1       }
 1   TP    CE11    Q         }   Set index
 2   QT    VV3     VV3       }
 3   TV    TH3     WW6
 4   EF    0       VV            Start writing
 5   RP    10170   WW7       }
 6   EW    10000   (30000)   }   Write blk
 7   RA    WW6     CE12          Add + 170
10   IJ    VV3     WW4           No. blks
11   MJ    0       TH1           Exit
     CA    WW12
```

Read Sub

```
    IA   RR
0   MJ   0       (30000)      Exit
1   MJ   0        RR7         Start (normal entrance)
2   CC   VV       CC36        Reverse direction
3   CC   RR14     CD2         RA (RS)
4   CC   RR5      CD2         RA (RS)
5   0    (30000)  (30000)     RS (RA)  VV5  CC37
6   RJ   RR6      RR7         One shot
7   TV   VV5      RR13        Set address
10  TV   RR17     IA          Set ignore
11  TP   CC37     VV4         Set index
12  EF   0        VV          Start reading
13  ER   10000    (30000)     Read
14  0    (30000)  (30000)     RA (RS)  RR13  CC1
15  IJ   VV4      RR13        One block
16  ER   0        A           I O A ⟶ A
17  MJ   0        RR          Exit
    CA   RR20
```

Index to VV3

```
    IA   RE
0   MJ   0       (30000)      Exit
1   TV   TH3      VV5         Set address
2   TP   TH3      Q           }
3   QT   CE4      A           } Index to VV3
4   SS   CE6      17          }
5   LT   0        VV3         }
6   MJ   0        RE          Exit
    CA   RE7
```

Read Fwd. or Bwd.

```
     IA    RA
 0   RJ    RR      RR1        Fwd (bwd)       normal
 1   ZJ    RA2     RA35       Parity ↓ no ──→ RA35
 2   RJ    RR      RR2        Bwd (fwd)       normal
 3   ZJ    RA4     RA24       Parity ↓ no ──→ RA24
 4   EF    0       CD6        Set high
 5   RJ    RR      RR2        Fwd (bwd)       high
 6   ZJ    RA7     RA35       Parity ↓ no ──→ RA35
 7   RJ    RR      RR2        Bwd (fwd)       high
10   ZJ    RA11    RA24       Parity ↓ no ──→ RA24
11   EF    0       CD7        Set low
12   RJ    RR      RR2        Fwd (bwd)       low
13   ZJ    RA14    RA35       Parity ↓ no ──→ RA35
14   RJ    RR      RR2        Bwd (fwd)       low
15   ZJ    RA16    RA24       Parity ↓ no ──→ RA24
16   TP    CF      PC2     }
17   RJ    PC      PC1     }  Print PARITY ERROR
20   EF    0       CD5        Set normal
21   RJ    RR6     RR2        Reverse direction
22   TV    CE10    IA         Set ignore address
23   MS    0       RA         Stop for rereads
24   RJ    RR6     RR2        Reverse direction
25   TV    CE20    IA         Set ignore
26   TP    VV2     A       }
27   AT    VV1     A       }  Move fwd (bwd) 1 blk
30   EF    0       A       }
31   0     30000   30000      RA (RS)  VV5  CC1
32   EF    0       CD5        Set normal
33   IJ    VV3     RA         n  blks
34   MJ    0       TH1        Exit
35   0     30000   30000      RA (RS)  VV5  CE12
36   MJ    0       RA32
     CA    RA37
```

Read Fwd.

```
     IA   RF
0    RJ   RE      RE1          Set index and address
1    TP   CE21    RR5    )
2    TP   CE      RR14   )
3    TP   CE2     RA31   )     Set for fwd.
4    TP   CE16    RA35   )
5    TP   CD3     VV1    )
6    RA   VV      CC17   )
7    MJ   0       RA
     CA   RF10
```

Read Bwd.

```
     IA   RB
0    RJ   RE      RE1          Set index and address
1    TP   CE22    RR5    )
2    TP   CE1     RR14   )
3    TP   CE3     RA31   )     Set for bwd.
4    TP   CE17    RA35   )
5    TP   CD4     VV1    )
6    RA   VV      CC20   )
7    MJ   0       RA
     CA   RB10
```

Rewinds

```
    IA   RW
0   RA   VV      CC13        Rewind
1   EF   0       VV
2   MJ   0       TH1         Exit
3   RA   VV      CC14        Rewind with interlock
4   MJ   0       RW1
    CA   RW5
```

Move Fwd.

```
    IA   MF
0   TU   CE7     MF4         Set for fwd.
1   RJ   RE      RE1         Set index
2   TV   CE7     IA          Set ignore address
3   MJ   0       MF4
4   TP   (30000) A           Set for fwd (bwd)
5   AT   VV      VV          Set EF wd.
6   EF   0       VV          Move 1 blk
7   IJ   VV3     MF6         n blks
10  MJ   0       TH1         Exit
    CA   MF11
```

Move Bwd.

```
    IA   MB
0   TU   CE10    MF4         Set for bwd.
1   MJ   0       MF1         ⟶ move fwd.
    CA   MB2
```

Print Sub.

```
     IA   PC
 0   MJ   0       (30000)      Exit
 1   MJ   0       PC16         Start
 2   0    30000   30000        Par. = 0 address     n
 3   TP   PC2     Q            }
 4   QT   CE13    A45          } Set index
 5   ST   CC1     PC20         }
 6   TU   PC2     PC7          Set address
 7   SP   30000   52           }
10   PR   0       A            } Print one word
11   SS   A       6            }
12   ZJ   PC10    PC13         }
13   RA   PC7     CE14         Next word
14   IJ   PC20    PC7          Finished ↓ no → PC7
15   MJ   0       PC           Exit
16   RP   4       PC3
17   PR   0       PC4
20   0    0       0            Index
     CA   PC21

     IA   WE
 0   TP   WE3     PC2          } "Cannot ignore.  Rerun"
 1   RJ   PC      PC1          }
 2   MS   0       IA
 3   0    WE4     5
 4   47   16300   60603        ↑ C A N N O
 5   01   04141   30603        T Δ I G N O
 6   12   20574   20404        R E ↓ . Δ Δ
 7   47   12201   23406        ↑ R E R U N
10   57   42040   40404        ↓ . Δ Δ Δ Δ
     CA   WE11

     IA   CF
 0   0    CF1     12
 1   47   12203   02204        ↑ R E A D Δ
 2   20   12120   31204        E R R O R Δ
 3   46   15301   21401        ( P A R I T
 4   25   04031   20424        Y Δ O R Δ S
 5   15   12031   63620        P R O C K E
 6   01   42574   24704        T ) ↓ . ↑ Δ
 7   04   24013   01201        Δ S T A R T
10   04   26031   20412        Δ F O R Δ R
11   20   12203   02224        E R E A D S
12   57   42040   40404        ↓ . Δ Δ Δ Δ
     CA   CF13
```

```
     IA    CC
 0   0     0        0          Zero
 1   0     0        1          One
 2   0     10000    04000      Bypass no. 1
 3   0     20000    04000      Bypass no. 2
 4   70    0        0          Mask off op. code
 5   10    0        0          Rewind
 6   20    0        0          Rewind with interlock
 7   30    0        0          Move fwd
10   40    0        0          Move bwd
11   50    0        0          Read fwd
12   60    0        0          Read bwd
13   0     200      0          Rewind
14   0     400      0          Rewind with interlock
15   0     4        1          Move fwd
16   0     14       1          Move bwd
17   0     602      0          Read fwd
20   0     612      0          Read bwd
21   0     606      0          Write low and stop  B = 0  1"  blk
22   0     616      0          Write high          B = 1  1"  blk
23   0     706      0          Write low           B = 2  2.4"blk
24   0     716      0          Write high          B = 3  2.4"blk
25   0     20       0          A = 2
26   0     40       0          A = 4
27   07    0        0          Mask off A
30   0     70000    0          Mask off B
31   02    0        0          A = 2
32   04    0        0          A = 4
33   0     10000    0          B = 1
34   0     20000    0          B = 2
35   0     30000    0          B = 3
36   0     10       0          To reverse read
37   0     0        167        Set index
     CA    CC40

     IA    CE
 0   RA    RR13     CC1   }    Set RR14       fwd
 1   RS    RR13     CC1   }                   bwd
 2   RA    VV5      CC1   }    Set RA31       fwd
 3   RS    VV5      CC1   }                   bwd
 4   7     77700    0          Mask no. blks (read)
 5   0     77       0          Mask servo no.
 6   0     100      0          No. of blks - 1 subtractor
 7   0     CC15     MF7
10   0     CC16     RA25
11   0     0        77         Mask
12   0     0        170        120(10)
13   0     0        77777      v mask
14   0     1        0          1 in u
15   0     0        WE
16   RA    VV5      CE12  }    Set RA35    fwd
17   RS    VV5      CE12  }                bwd
20   0     0        RA31
21   RS    VV5      CC37  }    Set RR5     fwd
22   RA    VV5      CC37  }                bwd
     CA    CE23
```

```
     IA    CD
 0   2     0        0            Tape bit
 1   1     0        0
 2   2     0        0
 3   2     4        1            Move fwd 1 blk
 4   2     14       1            Move bwd 1 blk
 5   2     1        50000        Set normal
 6   2     1        60000        High
 7   2     1        70000        Low
     CA    CD10
```

Don't Punch Variables

```
     IA    VV
 0   0     0        0                    EF code word
 1   0     0        0                    Move fwd (bwd) 1 blk (read)
 2   0     0        0                    Tape no.
 3   0     0        0                    Index (n blks)
 4   0     0        0                    Index 1 blk
 5   0     0        0                    Address
     CA    VV6
```

Check for Zero Blocks

```
     IA    CR
 0   QT    CR11     A
 1   TP    A        A
 2   EJ    CC4      TH1
 3   QT    CR12     A
 4   TP    A        A
 5   RP    20004    CR7
 6   EJ    CC7      TH1
 7   QT    CC4      A
10   MJ    0        30000
11   70    07700    0
12   77    77700    0
     CA    CR13
```

(ae) Error Prints of Translation Subroutines

		IA	FA		
F1	0	0	FA1	24	
	1	65	30506	63050	S E N T E N
	2	26	30017	77777	C E Δ 77 77 77
	3	0	0	0	Line no.
	4	01	01253	03151	Δ Δ B E F O
	5	54	30016	56624	R E Δ S T A
	6	54	66210	12530	R T , Δ B E
	7	32	34506	50171	G I N S Δ W
	10	34	66330	12401	I T H Δ A Δ
	11	45	30730	17151	K E Y Δ W O
	12	54	27010	10117	R D Δ Δ Δ (
	13	0	0	0	Key word
	14	43	22010	15430	) . Δ Δ R E
	15	65	66010	10101	S T Δ Δ Δ Δ
	16	01	01010	10101	Δ Δ Δ Δ Δ Δ
	17	01	01010	10151	Δ Δ Δ Δ Δ O
	20	31	01663	33465	F Δ T H I S
	21	01	65305	06630	Δ S E N T E
	22	50	26300	15051	N C E Δ N O
	23	66	01263	33026	T Δ C H E C
	24	45	30272	27777	K E D . 77 77
F2	25	0	FA26	26	
	26	65	30506	63050	S E N T E N
	27	26	30017	77777	C E Δ 77 77 77
	30	0	0	0	Line no.
	31	01	31345	46566	Δ F I R S T
	32	01	65734	72551	Δ S Y M B O
	33	46	21017	77777	L , Δ 77 77 77
	34	0	0	0	sym.
	35	01	21012	75130	Δ , Δ D O E
	36	65	01505	16601	S Δ N O T Δ
	37	34	50273	42624	I N D I C A
	40	66	30014	63032	T E Δ L E G
	41	24	46016	75034	A L Δ U N I
	42	26	51273	00165	C O D E Δ S
	43	30	50663	05026	E N T E N C
	44	30	22010	10101	E . Δ Δ Δ Δ
	45	01	01010	10101	Δ Δ Δ Δ Δ Δ
	46	54	30656	60151	R E S T Δ O
	47	31	01663	33465	F Δ T H I S
	50	01	65305	06630	Δ S E N T E
	51	50	26300	15051	N C E Δ N O
	52	66	01263	33026	T Δ C H E C
	53	45	30272	27777	K E D . 77 77
		CA	FA54		

		IA	FB		
F3	0	40	FB1	14	
	1	47	51543	00166	M O R E Δ T
	2	33	24500	11001	H A N Δ 5 Δ
	3	30	54545	15465	E R R O R S
	4	22	01015	43065	. Δ Δ R E S
	5	66	01513	10166	T Δ O F Δ T
	6	33	34650	16530	H I S Δ S E
	7	50	66305	02630	N T E N C E
	10	01	50516	60101	Δ N O T Δ Δ
	11	01	01010	10101	Δ Δ Δ Δ Δ Δ
	12	01	01010	10101	Δ Δ Δ Δ Δ Δ
	13	26	33302	64530	C H E C K E
	14	27	22777	77777	D . 77 77 77 77
F4	15	0	FB16	22	
	16	47	51543	00166	M O R E Δ T
	17	33	24500	10510	H A N Δ 2 5
	20	01	30545	45154	Δ E R R O R
	21	65	01663	33465	S Δ T H I S
	22	01	52545	13254	Δ P R O G R
	23	24	47220	10154	A M . Δ Δ R
	24	30	54302	42701	E R E A D Δ
	25	65	52302	63431	S P E C I F
	26	34	26246	63451	I C A T I O
	27	50	65220	10152	N S . Δ Δ P
	30	54	51325	42447	R O G R A M
	31	01	50516	60126	Δ N O T Δ C
	32	33	30244	53027	H E C K E D
	33	01	01010	10101	Δ Δ Δ Δ Δ Δ
	34	25	30735	15027	B E Y O N D
	35	01	65305	06630	Δ S E N T E
	36	50	26300	17777	N C E Δ 77 77
	37	0	0	0	line no.
		CA	FB40		

		IA	FD		
F5	0	0	FD1	15	
	1	47	51543	00166	M O R E Δ T
	2	33	24500	11004	H A N Δ 5 1
	3	05	01653	05066	2 Δ S E N T
	4	30	50263	06501	E N C E S Δ
	5	34	50016	63334	I N Δ T H I
	6	65	01525	45132	S Δ P R O G
	7	54	24472	20101	R A M . Δ Δ
	10	10	04066	63301	5 1 3 T H Δ
	11	65	30506	63050	S E N T E N
	12	26	30015	06747	C E Δ N U M
	13	25	30540	13465	B E R Δ I S
	14	01	77777	77777	Δ 77 77 77 77 77
	15	0	0	0	Line no.

F6	16	0	FD17	25		
	17	50	67472	53054	N U M B E R	
	20	01	51310	16750	Δ O F Δ U N	
	21	65	67256	52654	S U B S C R	
	22	34	52663	02701	I P T E D Δ	
	23	70	24543	42425	V A R I A B	
	24	46	30650	16301	L E S Δ + Δ	
	25	31	67502	66634	F U N C T I	
	26	51	50650	13465	O N S Δ I S	
	27	01	32543	02466	Δ G R E A T	
	30	30	54016	63324	E R Δ T H A	
	31	50	01100	40522	N Δ 5 1 2 .	
	32	01	01100	40666	Δ Δ 5 1 3 T	
	33	33	01657	34725	H Δ S Y M B	
	34	51	46010	10101	O L Δ Δ Δ Δ	
	35	34	65017	77777	I S Δ 77 77 77	
	36	0	0	0	Sym.	
	37	22	01017	15154	. Δ Δ W O R	
	40	45	34503	20151	K I N G Δ O	
	41	50	01653	05066	N Δ S E N T	
	42	30	50263	00177	E N C E Δ 77	
	43	0	0	0	Sent. no.	
		CA	FD44			
		IA	FE			
F7	0	40	FE1	16		
	1	47	51543	00166	M O R E Δ T	
	2	33	24500	10405	H A N Δ 1 2	
	3	01	26332	45424	Δ C H A R A	
	4	26	66305	46501	C T E R S Δ	
	5	34	50013	14651	I N Δ F L O	
	6	24	66345	03201	A T I N G Δ	
	7	52	51345	06601	P O I N T Δ	
	10	01	01010	10101	Δ Δ Δ Δ Δ Δ	
	11	01	01010	10101	Δ Δ Δ Δ Δ Δ	
	12	01	01265	15065	Δ Δ C O N S	
	13	66	24506	62201	T A N T . Δ	
	14	0	0	0		
	15	0	0	0	Constant	
	16	0	0	0		
F8	17	40	FE20	12		
	20	47	51543	00166	M O R E Δ T	
	21	33	24500	15150	H A N Δ O N	
	22	30	01273	02634	E Δ D E C I	
	23	47	24460	15251	M A L Δ P O	
	24	34	50660	13450	I N T Δ I N	
	25	01	26515	06566	Δ C O N S T	
	26	24	50662	20101	A N T . Δ Δ	
	27	0	0	0		
	30	0	0	0	Constant	
	31	0	0	0		
		CA	FE32			

		IA	FF								
F9	0	40	FF1	11							
	1	24	65656	74730		A	S	S	U	M	E
	2	27	01265	15065		D	Δ	C	O	N	S
	3	66	24506	60126		T	A	N	T	Δ	C
	4	51	50662	43450		O	N	T	A	I	N
	5	65	01240	14630		S	Δ	A	Δ	L	E
	6	66	66305	42201		T	T	E	R	.	Δ
	7	0	0	0	}						
	10	0	0	0	}	Constant					
	11	0	0	0	}						
F10	12	40	FF13	15							
	13	47	51543	00166		M	O	R	E	Δ	T
	14	33	24500	11101		H	A	N	Δ	6	Δ
	15	26	33245	42426		C	H	A	R	A	C
	16	66	30546	50134		T	E	R	S	Δ	I
	17	50	01313	47230		N	Δ	F	I	X	E
	20	27	01525	13450		D	Δ	P	O	I	N
	21	66	01265	15065		T	Δ	C	O	N	S
	22	66	24506	62201		T	A	N	T	.	Δ
	23	01	01010	10101		Δ	Δ	Δ	Δ	Δ	Δ
	24	01	01010	10101		Δ	Δ	Δ	Δ	Δ	Δ
	25	0	0	0	}						
	26	0	0	0	}	Constant					
	27	0	0	0	}						
		CA	FF30								

		IA	FG			
F11	0	40	FG1	10		
	1	24	01273	02634	A Δ D E C I	
	2	47	24460	15251	M A L Δ P O	
	3	34	50660	13450	I N T Δ I N	
	4	01	31347	23027	Δ F I X E D	
	5	01	52513	45066	Δ P O I N T	
	6	01	26515	06566	Δ C O N S T	
	7	24	50662	20101	A N T . Δ Δ	
	10	0	0	0	Constant	
F12	11	40	FG12	15		
	12	47	51543	00166	M O R E Δ T	
	13	33	24500	11101	H A N Δ 6 Δ	
	14	26	33245	42426	C H A R A C	
	15	66	30546	50134	T E R S Δ I	
	16	50	01702	45434	N Δ V A R I	
	17	24	25463	00166	A B L E Δ T	
	20	73	52300	16573	Y P E Δ S Y	
	21	47	25514	62201	M B O L . Δ	
	22	01	01010	10101	Δ Δ Δ Δ Δ Δ	
	23	01	01010	10101	Δ Δ Δ Δ Δ Δ	
	24	0	0	0	}	
	25	0	0	0	} Symbol	
	26	0	0	0	}	
		CA	FG27			

```
          IA    FH
F13   0   40    FH1     10
      1   70    24543   42425     V A R I A B
      2   46    30016   67352     L E Δ T Y P
      3   30    01657   34725     E Δ S Y M B
      4   51    46012   65150     0 L Δ C 0 N
      5   66    24345   06501     T A I N S Δ
      6   24    01525   13450     A Δ P 0 I N
      7   66    22010   17777     T . Δ Δ 77 77
     10   0     0       0             Symbol
F14   11   0     FH12    10
     12   50    51016   53050     N 0 Δ S E N
     13   66    30502   63001     T E N C E Δ
     14   50    67472   53054     N U M B E R
     15   01    51500   16530     Δ 0 N Δ S E
     16   50    66305   02630     N T E N C E
     17   01    31514   64651     Δ F 0 L L 0
     20   71    34503   20101     W I N G Δ Δ
     21   0     0       0             Line no.
F15   22   0     FH23    24
     23   46    34656   60151     L I S T Δ 0
     24   31    01702   45434     F Δ V A R I
     25   24    25463   06521     A B L E S ,
     26   01    46342   55424     Δ L I B R A
     27   54    73015   45167     R Y Δ R 0 U
     30   66    34503   06521     T I N E S ,
     31   01    31675   02666     Δ F U N C T
     32   34    51506   50124     I 0 N S Δ A
     33   50    27015   26530     N D Δ P S E
     34   67    27510   15152     U D 0 Δ 0 P
     35   30    54246   63451     E R A T I 0
     36   50    65013   32465     N S Δ H A S
     37   01    25302   65147     Δ B E C 0 M
     40   30    01010   16651     E Δ Δ Δ T 0
     41   51    01465   15032     0 Δ L 0 N G
     42   22    01017   15154     . Δ Δ W 0 R
     43   45    34503   20151     K I N G Δ 0
     44   50    01653   05066     N Δ S E N T
     45   30    50263   00177     E N C E Δ 77
     46   0     0       0             Line no.
          CA    FH47
```

```
        IA   FI
F16  0  40   FI1     3
     1  66   51510   14724       T 0 0 Δ M A
     2  50   73016   57347       N Y Δ S Y M
     3  25   51466   52277       B 0 L S . 77
F17  4  40   FI5     5
     5  50   51013   05027       N 0 Δ E N D
     6  01   51310   16530       Δ 0 F Δ S E
     7  50   66305   02630       N T E N C E
    10  01   65734   72551       Δ S Y M B 0
    11  46   22777   77777       L . 77 77 77 77
F18 12  40   FI13    14
    13  34   50265   15454       I N C 0 R R
    14  30   26660   16573       E C T Δ S Y
    15  47   25514   60165       M B 0 L Δ S
    16  30   53673   05026       E Q U E N C
    17  30   22010   10177       E . Δ Δ Δ 77
    20  0    0       0       }
    21  0    0       0       }       1st sym.
    22  0    0       0       }
    23  01   01777   77777       Δ Δ 77 77 77 77
    24  0    0       0       }
    25  0    0       0       }     2nd sym.
    26  0    0       0       }
        CA   FI27
```

```
        IA   NO
F19  0  40   NO1    7
     1  45   30730  17151   K E Y Δ W O
     2  54   27210  17777   R D , Δ 77 77
     3  0    0      0           Word
     4  01   21016  76530   Δ , Δ U S E
     5  27   01246  50124   D Δ A S Δ A
     6  01   70245  43424   Δ V A R I A
     7  25   46302  27777   B L E . 77 77
F20  10 40   NO11   17
     11 27   34473  05065   D I M E N S
     12 34   51500  16530   I O N Δ S E
     13 50   66305  02630   N T E N C E
     14 21   01343  10134   , Δ I F Δ I
     15 66   01245  25230   T Δ A P P E
     16 24   54652  10147   A R S , Δ M
     17 67   65660  12530   U S T Δ B E
     20 01   01010  10101   Δ Δ Δ Δ Δ Δ
     21 31   34546  56622   F I R S T .
     22 01   01543  06566   Δ Δ R E S T
     23 01   51310  16633   Δ O F Δ T H
     24 34   65016  53050   I S Δ S E N
     25 66   30502  63001   T E N C E Δ
     26 50   51660  12633   N O T Δ C H
     27 30   26453  02722   E C K E D .
        CA   NO30
```

(af) Set TN for Five or Seven Uniservos

```
     IA   OT
0    MJ   0       30000      Exit
1    MJ   10000   OT4        Entry.  Jump to OT4 if MJ1 is set.
2    TP   OT7     TN         Puts zero into TN when MJ1 is not set.
3    MJ   0       OT
4    TP   OT6     TN         Puts 0 3 0 into TN when MJ1 is set.
5    MJ   0       OT
6    0    3       0
7    0    0       0

     CA   OT10

     IA   TN
     0    0       0

     CA   TN1
```

(ag) Set up Translation Output Tape

```
     IA   UB
0    MJ   0       30000
1    TP   TN      A          }  Puts proper parameter into tape-write
2    AT   UB17    WT13       }  routine so that String-Outs will be written
                                either on tape 3 or tape 6.
3    RP   10024   UB5        }  Puts 20 lines of Z's into title block
4    TP   UB14    VN         }
5    TP   UB15    VN24       }  String-Outs goes on beginning of 2nd
6    TP   UB16    VN25       }  blockette of block.
7    RP   10142   UB11       }  Filling remainder of title block with
10   TP   UB14    VN26       }  lines of Z's.
11   TP   WT13    GT3        }  Writing title block on either tape 3 or
12   RJ   GT2     GT         }  tape 6.
13   MJ   0       UB         Jump to exit
14   74   74747   47474
15   65   66543   45032      S  T  R  I  N  G
16   02   51676   66501      -  0  U  T  S  Δ
17   71   00103   VN

     CA   UB20
```

(ah) Write Translation List on Tape (also region SS)

```
     IA    WT
0    MJ    0        30000
1    TP    UZ2      A       }
2    ZJ    WT       WT3     }   Has error routine been referenced?
3    TP    VN       A           No. of lines to A
4    DV    WT14     Q           No. of blocks to Q
5    ZJ    WT7      WT6         Is remainder zero?
6    RS    Q        WT15        If so, reduce no. of blocks by one.
7    LA    Q        25          No. of blocks to rt. position in A
10   AT    WT13     GT3     }   Adding in standard parameter with one block
11   RJ    GT2      GT      }   and using generalized tape handler.
12   MJ    0        WT          Exit
13   71    00103    VN          Standard parameter for use in gen. tape
                                 handler.
14   0     0        170         No. of lines to a block.
15   0     0        1

     CA    WT16
```

(ai) Put Referenced Sentence Number in List IZ

```
     IA    IX
 0   MJ    0       30000
 1   TP    A       IX46      Line no. to storage.
 2   TP    11      A         }
 3   TJ    IX40    IX11      } Is 11 ≤ 00 20454 00454?
 4   TP    IX46    A         }
 5   RP    20454   IX7       } Is line no. in ref. list?
 6   EJ    IZ      IX        }
 7   RA    IX47    IX36      Counting excess line no. referrals.
10   MJ    0       IX
11   TU    11      IX13      Setting up repeat for search of IZ accord-
                             ing to length of IZ.
12   TP    IX46    A         }
13   RP    30000   IX15      } Is line no. in ref. list?
14   EJ    IZ      IX        }
15   TV    11      IX26      No. of lines n in IZ to v of IX26
16   RA    11      IX37      Adding 0 2 2 to 11.
17   TJ    IX40    IX25      Is 11 ≤ 0 20454 454?
20   TP    IX34    UP3       } Error Print-Out: Referenced lines
21   RJ    UP2     UP        } exceed 150.
22   RJ    UZ      UZ1       Referral to string-out error routine
23   TP    IX36    IX47      Putting 1 into excess line-referral counter
24   MJ    0       IX
25   RA    IX26    IX33      IZ + n ⟶ v of N1
26   TP    IX46    30000     Line no. goes to next location at end of
                             IZ list
27   TV    IX26    IX31      }
30   RA    IX31    IX36      } Subsequent address of IZ list is
31   TP    IX35    30000     } cleared.
32   MJ    0       IX
33   0     0       IZ
34   0     IX41    5
35   0     0       0
36   0     0       1
37   0     2       2
40   0     20455   455
41   54    30313   05430     R E F E R E
42   50    26302   70146     N C E D Δ L
43   34    50306   50130     I N E S Δ E
44   72    26303   02701     X C E E D Δ
45   04    10037   77777     1 5 0 Δ Δ Δ
46   0     0       0
47   0     00000   0
     CA    IX50
```

(aj) Excess-Three Decimal to Octal

```
     IA    RS
 0   MJ    0       RS5          Entrance
 1   RJ    0       0            Alarm exit
 2   MJ    0       30000        Exit
 3   0     0       0            Output
 4   0     0       0            Input
 5   TP    RS51    RS3          Clearing output
 6   TP    RS51    RS42         Clearing digit counter
 7   TP    RS51    RS43         Clearing decimal number accumulator
10   TP    RS52    RS44         Setting index 5
11   TP    RS54    RS45         Setting 6 in working storage
12   LQ    RS4     6            Masking out each character and converting
13   QT    RS46    RS41         from excess 3.
14   RS    RS41    RS47
15   SJ    RS20    RS16         Below-0 end-of-digits check
16   ST    RS50    A            Above-9 end-of-digits check
17   SJ    RS22    RS20
20   TP    RS42    A            Count-of-digits zero check
21   ZJ    RS26    RS2
22   RA    RS42    RS53         Count of digits
23   LQ    RS43    6            Accumulation of decimal digits in one line
24   RA    RS43    RS41
25   IJ    RS44    RS12
26   RS    RS45    RS42
27   MP    RS45    RS54         Computing and performing initial shift on
30   TV    A       RS31         line of decimal digits.
31   LQ    RS43    0
32   RS    RS42    RS53         Computing index of octal conversion
33   LQ    RS43    6
34   SP    RS3     2            Decimal-to-octal conversion
35   SA    RS3     1
36   QA    RS46    RS3
37   IJ    RS42    RS33
40   MJ    0       RS2          Jump to exit
41   0     0       0            Storage for single decimal digits
42   0     0       0            1st-Counter for digits; 2d-Index for octal
                                 conversion
43   0     0       0            Line of decimal digits is accumulated here.
44   0     0       0            Set at start to index 5
45   0     0       0            Set at start to 6
46   0     0       77           Mask
47   0     0       3            Subtrahend to convert from excess three
50   0     0       12           10(10)
51   0     0       0            Zero constant
52   0     0       5            Index constant
53   0     0       1            Constants
54   0     0       6
     CA    RS55
```

(ak) Excess-Three Decimal to Floating Point

		IA	GG		
	0	MJ	0	GG6	Entrance
	1	RJ	0	0	Alarm exit not used
	2	MJ	0	30000	Exit
	3	0	0	0	Floating point output
	4	0	0	0	Input in excess-three decimal
	5	0	0	0	
	6	TP	CF2	CC	Set indexes
	7	TP	CF3	CC1	
	10	TP	CF6	CC2	Set divisor at 12_8
	11	RP	10006	GG13	Clear 6 places of working storage,
	12	TP	CF13	CC3	CC3-CC11
	13	TU	GG165	GG15	Put GG4 in u part of GG15
	14	TV	GG164	GG33	Put CC13 in v part of GG33
	15	LQ	GG4	6	Mask out character and put in CC12
	16	QT	CF	CC12	
	17	RS	CC12	CF1	Subtract 3
	20	SJ	GG21	GG24	Below zero end-of-digits check
	21	TP	CC3	A	Test for zero number of significant
	22	AT	CC4	A	digits
	23	ZJ	GG66	GG162	
	24	EJ	CF16	GG56	Test for decimal point
	25	ZJ	GG30	GG26	Test for zero
Assembly of digits before decimal point	26	TP	CC3	A	Zero in CC3 counter causes zero in CC12
	27	ZJ	GG32	GG35	to be bypassed as a non-significant zero in count.
	30	RS	A	CF6	Test for above-9 end-of-digits
	31	SJ	GG32	GG21	character
	32	RA	CC3	CF3	Count of digits before decimal point
	33	TP	CC12	CC13	Storage of digits in consecutive addresses
	34	RA	GG33	CF3	Modify previous v to next address
	35	IJ	CC	GG15	6-character index jump
	36	IJ	CC1	GG40	2-line index jump
	37	MJ	0	GG64	Jump to conversion at end of input assembly
	40	TU	GG166	GG15	Set u of GG15 to GG5 for second input line
	41	TP	CF2	CC	Restore 6-character index to 5
	42	MJ	0	GG15	Jump to assembly of second input line
Assembly of digits after decimal point	43	TP	GG15	GG44	
	44	0	0	0	
	45	QT	CF	CC12	
	46	RS	CC12	CF1	
	47	SJ	GG21	GG50	
	50	RS	A	CF6	
	51	SJ	GG52	GG21	
	52	RA	CC4	CF3	
	53	TP	GG33	GG54	
	54	0	0	0	

	55	RA	GG33	CF3	
	56	IJ	CC	GG43	
	57	IJ	CC1	GG61	
	60	MJ	0	GG64	
	61	TU	GG166	GG15	
	62	TP	CF2	CC	
	63	MJ	0	GG43	
	64	TP	CC3	A	
	65	AT	CC4	A	Computing index CC11 for octal
	66	ST	CF3	CC11	conversion
Conversion of all digits to an octal number contained in CC7 and CC10.	67	MP	CC7	CF6	
	70	TP	A	CC1	
	71	MP	CC10	CF6	
	72	AT	CC13	A	
	73	LT	0	CC12	
	74	LT	10000	CC10	
	75	TP	CC12	A	
	76	AT	CC1	CC7	
	77	RA	GG72	CF4	
	100	IJ	CC11	GG67	
	101	TU	GG167	GG72	Restoring CC13 to u part of GG72
	102	TP	CC4	A	Test for zero digits after decimal
	103	ZJ	GG104	GG132	point
	104	EJ	CF7	GG113	Test for 11 digits after decimal point
	105	ST	CF3	CC11	Set index for computation of power-of-10 divisor
Computation of "power-of-10" divisor	106	IJ	CC11	GG110	
	107	MJ	0	GG114	
	110	MP	CF6	CC2	
	111	TP	A	CC2	
	112	MJ	0	GG106	
	113	TP	CF11	CC2	$\frac{10"}{8} \longrightarrow$ CC2. 10" alone would overflow A_R
	114	TP	CF6	A	
	115	ST	CC3	CC	Computation of dividend shift
	116	MP	CC	CF1	needed to get 10 significant fig-
	117	TP	A	CC5	ures in quotient.
	120	SP	CC7	44	
	121	TV	CC5	GG122	Octal number $\longrightarrow$ A with proper shift.
	122	SA	CC10	0	
	123	DV	CC2	CC26	Division by power of 10
	124	TP	CC4	A	If CC4 = 11, additional shift to
	125	EJ	CF7	GG127	give effect of dividing by 8. See
	126	MJ	0	GG130	113 above.
	127	LA	CC26	105	
	130	TP	CC26	A	
	131	MJ	0	GG134	Final converted-to-octal number $\rightarrow$ A
	132	SP	CC7	44	
	133	SA	CC10	0	
	134	SF	A	CC1	Scaling for number size and to put it in fixed position
	135	AT	CF10	CC	Rounding off

```
136   TP      CC          Q          If round-off caused a carry-over
137   QJ      GG140       GG143      from first significant digit 1
140   LQ      CC          33         → CC6 and shifting is altered.
141   TP      CF3         CC6
142   MJ      0           GG144
143   LQ      CC          34
144   TP      CF12        Q          Assembly and insertion of mantissa
145   QT      CC          CC         into output line.
146   TP      CC          GG3
147   TP      CC6         CC3        [(CC6) + 243 - (CC5) + K] shifted
150   RA      CC3         CF5        to position of characteristic and
151   RS      CC3         CC5        put into CC4
152   SA      CC1         33
153   TP      A           CC4
154   TP      CC1         A          If k = 45₈, both characteristic
155   EJ      CF14        GG162      and mantissa = 0.
156   TJ      CF14        GG160      If 45₈ > k, 110₈ is subtracted
157   RS      CC4         CF15       from CC4
160   RA      GG3         CC4        Final assembly of non-zero float-
                                     ing point number.
161   MJ      0           GG2        Jump to exit.
162   TP      CF13        GG3        Clearing output line to zero
163   MJ      0           GG2        Exit
164   0       0           CC13
165   0       GG4         0
166   0       GG5         0
167   0       CC13        0
      CA      GG170


      IA      CF
  0   0       0           77
  1   0       0           3
  2   0       0           5
  3   0       0           1
  4   0       1           0
  5   0       0           243
  6   0       0           12
  7   0       0           13
 10   0       0           200
 11   13      51035       56400
 12   00      07777       77777
 13   0       0           0
 14   0       0           45
 15   11      0           0
 16   0       0           17
      CA      CF17
```

Sequential Uses of Working Storage CC

0 { Index 5 for 6 characters in a line of input.
Varying multiplier to determine size of fractional part in quotient.
Temporary storage for first parts of result.

1 { Index 1 for 2 lines of input.
Storage for $10 \cdot A_L$.
k of scale factor.

2 Varying product and final divisor.

3 { Number of digits before decimal point.
Used to accumulate characteristic.

4 { Number of digits after decimal point.
Used to accumulate characteristic.

5 Shift to give fractional part in quotient.

6 Used to denote carry-over in round off.

7 A_L in octal conversion.

10 A_R in octal conversion.

11 { Varying index for conversion to binary.
Varying index for multiplication in computing divisor.

12-25 For accumulation of first 11 decimal numbers after subtracting 3.

26 { Used for 12th decimal number.
Used for quotient after division.

(a1) Assign Constant Call Word

```
     IA    GW
 0   MJ    0       30000     Exit
 1   TP    A       FX12      Entry.  To store constant temporarily
 2   TP    10      A         Test if number of constants ≤ 1000.
 3   TJ    FX      ZS
 4   TP    FX12    A         Test if constant is in list that is
 5   RP    21000   GW7       already 1000 in length.
 6   EJ    CL      GW13
 7   RA    FX13    FX2       Count of number of referrals of con-
                             stants not in list after alarm.
10   TP    FX3     Q         Giving the maximum call word 67777 to
11   SP    FX3     17        all of these referral constants.
12   MJ    0       GW
13   SN    Q       17        [- j, - (n - r)] ⟶ (A_R)_u
14   SA    FX11    0         [- j, (r - n)]  +  [j, n]  = r ⟶ (A_R)_u
15   MJ    0       ZS22
     CA    GW16

     IA    ZS
 0   TP    FX12    A         Constant ⟶ A
 1   TU    10      ZS2       Sets up u of Repeat by constant list
                             count in 10.
 2   RP    30000   ZS7       Test if constant is in list CL.
 3   EJ    CL      ZS4
 4   SN    Q       17        [- j, - (n - r)] ⟶ (A_R)_u
 5   SA    10      0         [- j, (r - n)]  +  [j,n]  = r ⟶ (A_R)_u
 6   MJ    0       ZS22
 7   TV    10      ZS20      n ⟶ (ZS20)_v
10   RA    10      FX1       Counter 10 increased by 1.
11   TJ    FX      ZS17      Is call word list of constants ≤ 1000?
12   TP    ZS27    UP3       Alarm print-out -- too many constants
13   RJ    UP2     UP
14   RJ    UZ      UZ1       References error routine.
15   TP    FX2     FX13      Starts excess count of referrals in FX13.
16   MJ    0       GW10
17   RA    ZS20    ZS26      (n + c1) ⟶ (ZS26)_v
20   TP    FX12    30000     Constant added to next position in list
21   SP    10      17        n ⟶ (A_R)_u
22   AT    FX4     Q         (r + 66777) ⟶ q_u
23   QT    FX5     Q         Irrelevant material in q masked out.
                             Call word formed in A_u
24   LQ    Q       25        Call word formed in Q_v
25   MJ    0       GW
26   0     0       CL
27   0     FX6     3         Parameter for error print-out:  too many
                             constants
     CA    ZS30

     IA    FX
 0   00    21001   01001     Threshold constant for check on size of
                             CL.
```

```
 1    0     1       1
 2    0     0       1
 3    0     0       67777     Maximum call word. Given to all new
                              constants exceeding 1000.
 4    0     66777   0         Base number from which call words are
                              determined by adding to position in list
                              CL.
 5    0     77777   0         Mask
 6    66    51510   14724     T 0 0 Δ M A
 7    50    73012   65150     N Y Δ C 0 N
10    65    66245   06665     S T A N T S
11    0     21000   0
12    0     0       0
13    0     0       0
      CA    FX14
```

(am) Close VARY File Item and Variable List File Item

```
      IA    VE
 0    MJ    0       30000
 1    SP    VI1     0         Are there any unclosed items in the Vary
                              File?
 2    EJ    VI2     VE6
 3    TU    VI      VE4       Yes. Is the current sentence number >
 4    SP    30000   0         the sentence number in the last unclosed
 5    TJ    WL1     VG6       item of the Vary File?
(1)   6    RJ    RA      RA1       No, so reference Routine A and then exit.
 7    MJ    0       VE
(2)  10    TP    VI2     VI        (γ) ⟶ α
     11    TP    VI2     VI1       (γ) ⟶ β
(3)  12    TV    VI3     VE14      Insert call word of current sentence in-
     13    SP    WL3     17        to u-portion of word at address given by
     14    TU    A       30000     (temp 1)
     15    TU    VI3     VE16      Close Vary File item by inserting JUMP
     16    RA    30000   VH3       flag (bit 35) in word at address given
                                   by (temp 1)
     17    MJ    0       VE6
           CA    VE20

           IA    VG
(4)   0    TU    VG21    VG2
      1    TV    VI3     VG3       Insert call word given by (α) + 1 (v
      2    SP    30000   17        portion) into u-portion of word at addr.
      3    TU    A       30000     given by (temp 1)
      4    TU    VI3     VG5       Close Vary File item by inserting RESUME
      5    RA    30000   VH4       flag (bits 35,34) into word given by
(5)   6    SP    VI      0         (temp 1)
      7    AT    VH      VI3       Record address (α) + 1 into temp 1.
     10    TU    A       VG11
     11    SP    30000   0         Set index to no. of WITH words.
     12    LT    6       VI4
(6)  13    RS    VL      VH2       Close last file of variables
```

```
(7)  14   IJ   VI4     VG13
     15   RS   VI      VH1      Decrease ( α ) by two
     16   TJ   VI1     VE10     (β) > ( α )?
     17   SA   VH      0        ┐
     20   TU   A       VG21     │ Is word at address given by ( α ) + 1
     21   TP   30000   Q        ┘ flagged?
     22   QJ   VG15    VG23     ┘
(8)  23   TU   VI      VG24     No.  Is the current sentence number >
     24   SP   30000   0        the sentence number at address given
     25   TJ   WL1     VG       by ( α ) ?
     26   MJ   0       VE12
          CA   VG27
```

Close VARY File Item and Variable List File Item (cont.)

	IA	VH		
0	0	1	1	
1	0	2	2	
2	0	3	3	
3	40	0	0	JUMP flag
4	60	0	0	RESUME flag
	CA	VH5		

	IA	VI		
0	0	VF1	VF1	α
1	0	VF1	VF1	β
2	0	VF1	VF1	γ
3	0	0	0	Temp 1
4	0	0	0	Index
	CA	VI5		

(an) Flex Codes for Print Text

	IA	FC		XS3
0	5	60000	0	00 = Superscript minus
1	0	40000	0	01 = Space
2	45	60000	0	02 = minus
3	43	70000	0	03 = 0
4	45	20000	0	04 = 1
5	47	40000	0	05 = 2
6	47	0	0	06 = 3
7	46	40000	0	07 = 4
10	46	20000	0	10 = 5
11	46	60000	0	11 = 6
12	47	20000	0	12 = 7
13	46	0	0	13 = 8
14	43	30000	0	14 = 9
15	5	40000	0	15 = Superscript /
16	41	30112	0	16 = Gtr >
17	4	60000	0	17 = open parenthesis (
20	7	0	0	20 = Superscript 3
21	44	60000	0	21 = Comma ,
22	44	20000	0	22 = Period .
23	44	60000	0	23 = Semicolon ;
24	3	0	0	24 = A
25	2	30000	0	25 = B
26	1	60000	0	26 = C
27	2	20000	0	27 = D
30	2	0	0	30 = E
31	2	60000	0	31 = F
32	1	30000	0	32 = G
33	0	50000	0	33 = H
34	1	40000	0	34 = I
35	6	20000	0	35 = Superscript 5
36	6	0	0	36 = Superscript 8
37	41	12412	0	37 = lsr <
40	7	40000	0	40 = Superscript 2
41	6	40000	0	41 = Superscript 4
42	45	0	0	42 = absolute value |
43	4	20000	0	43 = closed parenthesis)
44	3	20000	0	44 = J
45	3	60000	0	45 = K
46	1	10000	0	46 = L
47	0	70000	0	47 = M
50	0	60000	0	50 = N
51	0	30000	0	51 = O
52	1	50000	0	52 = P
53	3	50000	0	53 = Q
54	1	20000	0	54 = R
55	6	60000	0	55 = Superscript 6

56	42	76154	0	56 = *
57	3	30000	0	57 = Superscript 9
60	3	70000	0	60 = Superscript 0
61	5	20000	0	61 = Superscript |
62	4	40000	0	62 = Superscript !
63	45	40000	0	63 = +
64	5	40000	0	64 = /
65	2	40000	0	65 = S
66	0	10000	0	66 = T
67	3	40000	0	67 = U
70	1	70000	0	70 = V
71	3	10000	0	71 = W
72	2	70000	0	72 = X
73	2	50000	0	73 = Y
74	2	10000	0	74 = Z
75	7	20000	0	75 = Superscript 7
76	44	40000	0	76 = =
77	7	70000	0	77 = ignore
	CA	FC100		

(ao) Line Number Processor

```
                IA    LN
           0    MJ    0        LN5       Case I entry
           1    MJ     0       LN7       Case II entry
           2    MJ    0        30000     Exit
           3    00    30000    30000     Output
           4    00    30000    30000     Input
           5    TV    RC14     PA1       Set alarm switch to Case I
           6    MJ    0        DR0       Process line number
           7    TV    RC15     PA1       Set alarm switch to Case II
          10    MJ    0        DR        Process line number
          11    TP    LN4      LN3       Enter here after alarm print and put
                                           input line number in output line.
          12    MJ    0        LN2       Exit
                CA    LN13

                IA    DR
SETUP     0     TV    RC6      MR27      Preset ".ΔΔ" one shot switch
AND       1     TV    DR       DR15      Preset ".ΔΔ" one shot switch in driver
DRIVER    2     RJ    MR12     MR12      Preset 3-point distributor
          3     RJ    MR43     MR43      Preset 2-point distributor
          4     TP    UC23     LN3       Prestore output word.
          5     TP    UC4      WS3       Prestore master counter
          6     TP    LN4      WS        Set up line number in temporary
          7     TV    RC10     MR3       Set Exit 1 to master counter
         10     TV    RC13     MR4       Set Exit 2 to period routine
         11     TV    RC12     MR5       Set Exit 3 to integral part = 0
         12     TV    RC11     MR7       Set Exit 4 to integral part ≠ 0
         13     TV    RC1      MR16      Set ADD/NO ADD switch to NO ADD
         14     IJ    WS3      MR        Master counter
         15     RJ    DR15     30000     One shot switch to add ".ΔΔ"
         16     TP    LN3      A         Determine if input line number has at
         17     EJ    UC23     PA30      least one nonzero digit; if not
         20     EJ    UC24     PA30      ALARM 4
         21     MJ    0        LN2       Input line number O.K., exit.
                CA    DR22

                IA    MR
CODE PICK 0     LQ    WS       6         Shift line number 6 and pick up next
UP AND    1     QT    UC20     A         character in A
SORT      2     RP    20004    MR4
          3     EJ    UC14     30000     Exit 1 Δ, ,(, )
          4     EJ    UC13     30000     Exit 2  .
          5     EJ    UC1      30000     Exit 3  0
          6     RP    20011    PA11      Alarm 1:  ILLEGAL CHARACTER
          7     EJ    UC2      30000     Exit 4 1, 2, 3, 4, 5, 6, 7, 8, 9
INTEGRAL  10    TV    RC       MR16      Set ADD/NO ADD switch to ADD
PART      11    TV    RC2      MR3       Set Exit 1 to "Δ" routine
          12    RJ    MR12     MR13
          13    RJ    MR12     MR22      I   3-Point Distributor
```

```
                14   RJ   MR12    MR22     II  ┐
                15   MJ   0       MR24     III ┘
                16   MJ   0       30000    ADD/NO ADD switch ┐ ADD/NO ADD Subrou-
                17   SA   LN3     6        Add char. to      ┘ tine
                20   TP   A       LN3      Output and
                21   MJ   0       30000    exit
                22   RJ   MR21    MR16     I = II Add digit if it is significant.
                23   MJ   0       DR14     Jump to master counters
                24   TV   RC3     MR5      III Set Exit 3 and 4 to Alarm 2
                25   TV   RC3     MR7
                26   RJ   MR21    MR16     Add digit if it is significant.
                27   RJ   MR27    30000    One shot switch, set initially to MR31
                30   MJ   0       DR14     Jump to master counter
                31   SP   LN3     0        Output → A
                32   SA   UC13    6        Add .
                33   SA   UC14    6        Add Δ
                34   AT   UC14    LN3      Add Δ and replace in LN3
                35   MJ   0       DR14     Jump to master counter
PERIOD          36   TV   RC2     MR3      Set Exit 1 to Δroutine
ROUTINE         37   TV   RC5     MR4      Set Exit 2 to Alarm 4
                40   TV   RC7     MR5      Set Exit 3 to fract. part
                41   TV   RC7     MR7      Set Exit 4 to fract. part
                42   MJ   0       MR27     Jump to one-shot add in ".ΔΔ"
FRACTIONAL      43   RJ   MR43    MR44     ┐
PART            44   RJ   MR43    MR46     I  } 2-Point Distributor
ROUTINE         45   MJ   0       MR60     II ┘
                46   EJ   UC1     MR55     I  Is this digit zero?  No, → MR47
                47   LA   A       6        Shift to 2nd position "----X-"
                50   TP   UC21    Q        Mask → Q
                51   QS   A       LN3      Mask digit into LN3 = Output
                52   TP   UC      WS1      Clear TEMP
                53   TP   UC20    WS2      Set Mask
                54   MJ   0       DR14     Jump to master counter
                55   LT   10006   WS1      Digit is zero:  shift to "----X-" and
                                           store in TEMP
                56   TP   UC22    WS2      Set Mask
                57   MJ   0       DR14     Jump to master counter
                60   TV   RC4     MR5      II ┐  Set Exits 3, 4 to
                61   TV   RC4     MR7        ┘     Alarm 3
                62   EJ   UC1     DR14     Is this digit zero?  Yes, → master
                                                                    counter
                63   AT   WS1     WS1      No:  add digit to TEMP
                64   TP   WS2     Q        Mask → Q
                65   QS   WS1     LN3      Mask digit or digits into Output
                66   MJ   0       DR14     Jump to master counter
SPACE           67   TV   RC10    MR3      Set Exit 1 to master counter
ROUTINE         70   TV   RC5     MR4      Set Exit 2 to Alarm 4
                71   TV   RC5     MR5      Set Exit 3 to alarm 4
                72   TV   RC5     MR7      Set Exit 4 to Alarm 4
                73   MJ   0       DR14     Jump to master counter
                     CA   MR74
```

	IA	RC		
0	0	0	MR17	Sets ADD/NO ADD switch to ADD
1	0	0	MR21	Sets ADD/NO ADD switch to NO ADD
2	0	0	MR67	Sets Exit 1 to Δ routine
3	0	0	PA16	Alarm 2 entry
4	0	0	PA23	Alarm 3 entry
5	0	0	PA30	Alarm 4 entry
6	0	0	MR31	Sets add ".ΔΔ " one-shot switch
7	0	0	MR43	Fractional part routine entry
10	0	0	DR14	Sets Exit 1 to master counter
11	0	0	MR10	Integral part entry ≠ 0
12	0	0	MR11	Integral part entry = 0
13	0	0	MR36	Period routine entry
14	0	0	PA3	Case I alarm switch
15	0	0	PA5	Case II alarm switch
	CA	RC16		

		IA	UC		
	0	0	0	0	
EXCESS-THREE CODES	1	0	0	3	0
	2	0	0	4	1
	3	0	0	5	2
	4	0	0	6	3
	5	0	0	7	4
	6	0	0	10	5
	7	0	0	11	6
	10	0	0	12	7
	11	0	0	13	8
	12	0	0	14	9
	13	0	0	22	.
	14	0	0	1	Δ
	15	0	0	77	′
	16	0	0	17	(
	17	0	0	43	)
	20	0	0	77	Masks
	21	0	0	7700	
	22	0	0	7777	

```
23    1     1010     10100        Δ Δ Δ Δ 0 0
24    1     1012     20101        Δ Δ Δ . Δ Δ
      CA    UC25

      IA    WS
 0    0     30000    30000        "TEMP" to take apart input line number
 1    0     30000    30000        Temporary to store fractional part of
                                   digits
 2    0     30000    30000        Holds mask for inserting fractional part
                                   of digits
 3    0     30000    30000        Master counter (set to 6)
      CA    WS4

      IA    PA
 0    RJ    UZ0      UZ1          Reference error routine
 1    MJ    0        0            Switch to case I or case II
 2    MJ    0        0            Sub sub-routine exit
 3    RJ    WA       WA2          Print error heading case I
 4    MJ    0        PA2          Exit to print alarm comment
 5    TP    LN4      CD3          Fill input Line Number in heading case II
 6    TP    CD       UP3   }      Print heading with
 7    RJ    UP2      UP    }        Uniprint
10    MJ    0        PA2          Exit to print alarm comment
11    RJ    PA2      PA           Print (alarm 1 entry)
12    TP    LN4      CD15  }
13    TP    CD5      UP3   }      Alarm comment 1
14    RJ    UP2      UP    }
15    MJ    0        LN11
16    RJ    PA2      PA           Print (alarm 2 entry)
17    TP    LN4      CD27  }
20    TP    CD17     UP3   }      Alarm comment 2
21    RJ    UP2      UP    }
22    MJ    0        LN11
23    RJ    PA2      PA           Print (alarm 3 entry)
24    TP    LN4      CD42  }
25    TP    CD31     UP3   }      Alarm comment 3
26    RJ    UP2      UP    }
27    MJ    0        LN11
30    RJ    PA2      PA           Print (alarm 4 entry)
31    TP    LN4      CD54  }
32    TP    CD       UP    }      Alarm comment 4
33    RJ    UP2      UP    }
34    MJ    0        LN11
      CA    PA35
```

```
     IA    CD
 0   00    CD1     4          Parameter
 1   65    30506   63050      S E N T E N }  Heading
 2   26    30010   17777      C E Δ Δ     }
 3   01    01010   10101      Δ Δ Δ Δ Δ Δ    Input Line Number
 4   01    01017   77777      Δ Δ Δ
 5   40    CD6     11         Parameter
 6   34    46463   03224      I L L E G A    Comment 1
 7   46    01263   32454      L Δ C H A R
10   24    26663   05401      A C T E R Δ
11   34    50016   53050      I N Δ S E N
12   66    30502   63001      T E N C E Δ
13   50    67472   53054      N U M B E R
14   01    77777   77717      Δ           (
15   01    01010   10101      Δ Δ Δ Δ Δ Δ    Input Line Number
16   43    77777   77777      )
17   40    CD20    11         Parameter
20   30    72665   42401      E X T R A Δ    Comment 2
21   34    50663   03254      I N T E G R
22   24    46012   73432      A L Δ D I G
23   34    66650   13450      I T S Δ I N
24   01    65305   06630      Δ S E N T E
25   50    26300   15067      N C E Δ N U
26   47    25305   40117      M B E R Δ (
27   01    01010   10101      Δ Δ Δ Δ Δ Δ    Input Line Number
30   43    77777   77777      )
31   40    CD32    12         Parameter
32   30    72665   42401      E X T R A Δ    Comment 3
33   31    54242   66634      F R A C T I
34   51    50244   60127      O N A L Δ D
35   34    32346   66501      I G I T S Δ
36   34    50016   53050      I N Δ S E N
37   66    30502   63001      T E N C E Δ
40   50    67472   53054      N U M B E R
41   01    77777   77717      Δ           (
42   01    01010   10101      Δ Δ Δ Δ Δ Δ    Input Line Number
43   43    77777   77777      )
44   40    CD45    11         Parameter
45   34    46463   03224      1 L L E G A    Comment 4
46   46    01653   05367      L Δ S E Q U
47   30    50263   00134      E N C E Δ I
50   50    01653   05066      N Δ S E N T
```

```
51   30   50263   00150      E  N  C  E  Δ  N
52   67   47253   05401      U  M  B  E  R  Δ
53   17   77777   77777      (
54   01   01010   10101      Δ  Δ  Δ  Δ  Δ  Δ     Input Line Number
55   43   77777   77777      )
     CA   CD56
```

b. Translators

The two dimension translators are placed first in the group that follows. All other translators have been inserted in the order in which they are stored on the drum. See the regional assignments of translators in the preceding subdivision for this order.

Dimension String-Out. No. 1

Dimension String-out No. 1 is entered when Dimension Statement occurs as the 1st Sentence of a program. The routine uses the information in the dimension statement to assign a drum address and call word for each variable, and to build a file for each variable. Each file is added to the Dimension List. When End of Sentence is encountered the routine exists to Dimension String-out No. 2.

Successive operands in a sentence are obtained by referencing the Get Next Symbol routine. Variable name is picked up, assigned the current drum address, and given a call word of 77 _ _ _ type. Subscripts are picked up and used to form modulus and multipliers for the variable. When closing parenthesis is picked up, the file is sent to the Dimension List. Drum address for the next variable is formed by adding modulus to old drum address and next call word is set up for assignment to next variable.

Error Print-outs: This routine references Error Routine (WA) for print-out of sentence number and sentence type, before all error print-outs. Several types of errors are detected:

If more than four subscripts are given for a variable, the routine prints: TOO MANY SUBSCRIPTS FOR VARIABLE '_ _ _'

If a new variable name is picked up before closing parenthesis for preceding variable is found, routine prints: INFORMATION FOR VARIABLE '_ _ _' IS INCOMPLETE

If more than 512 variables appear in a sentence, the routine prints: TOO MANY VARIABLES

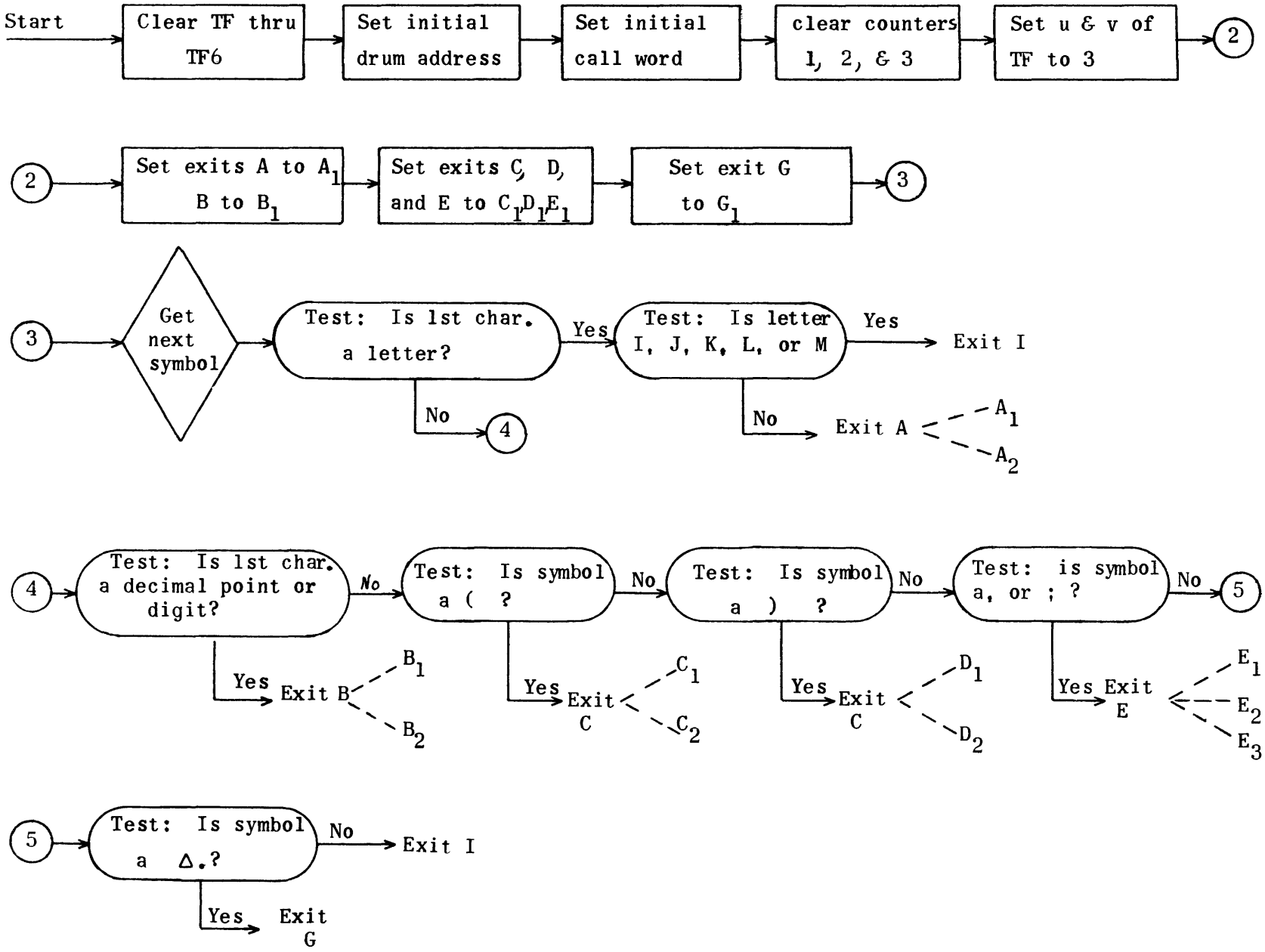
If available drum storage is exceeded, the routine prints: VARIABLES REQUIRE TOO MUCH STORAGE

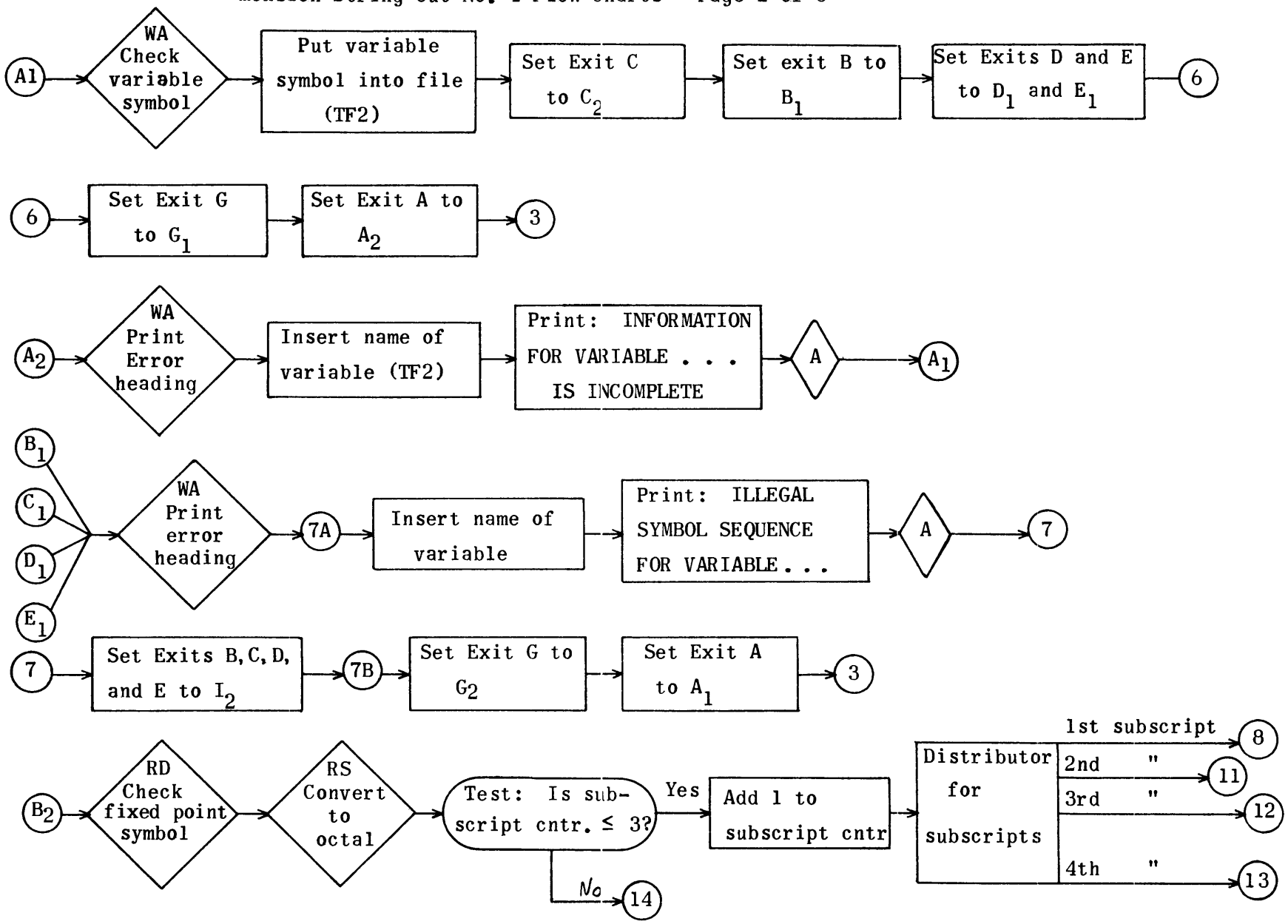
If a fixed-point variable name is given in a sentence, the routine prints: ILLEGAL SYMBOL

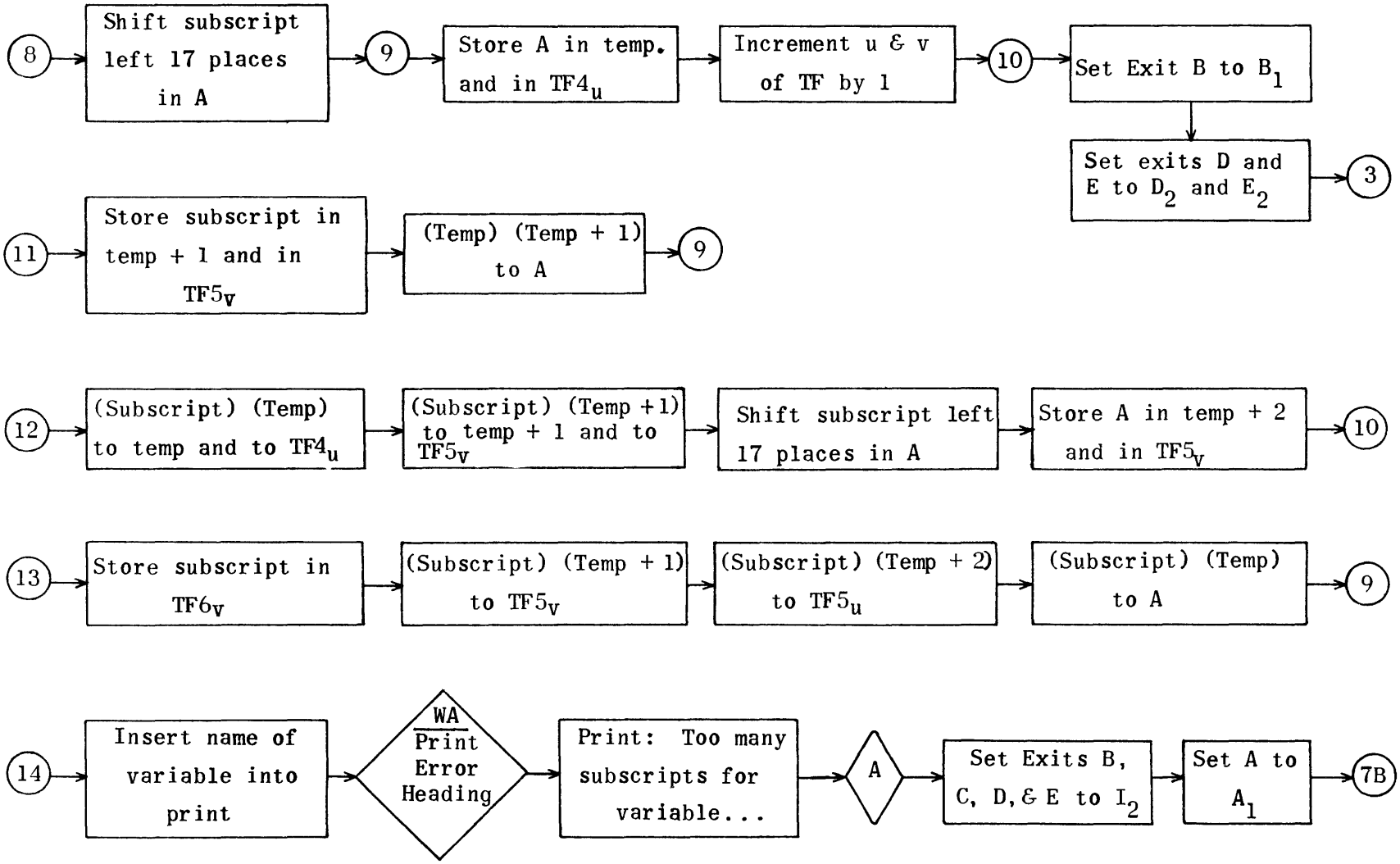
If an opening parenthesis or comma between subscripts is missing, the routine prints: ILLEGAL SYMBOL SEQUENCE FOR VARIABLE '_ _ _ _'

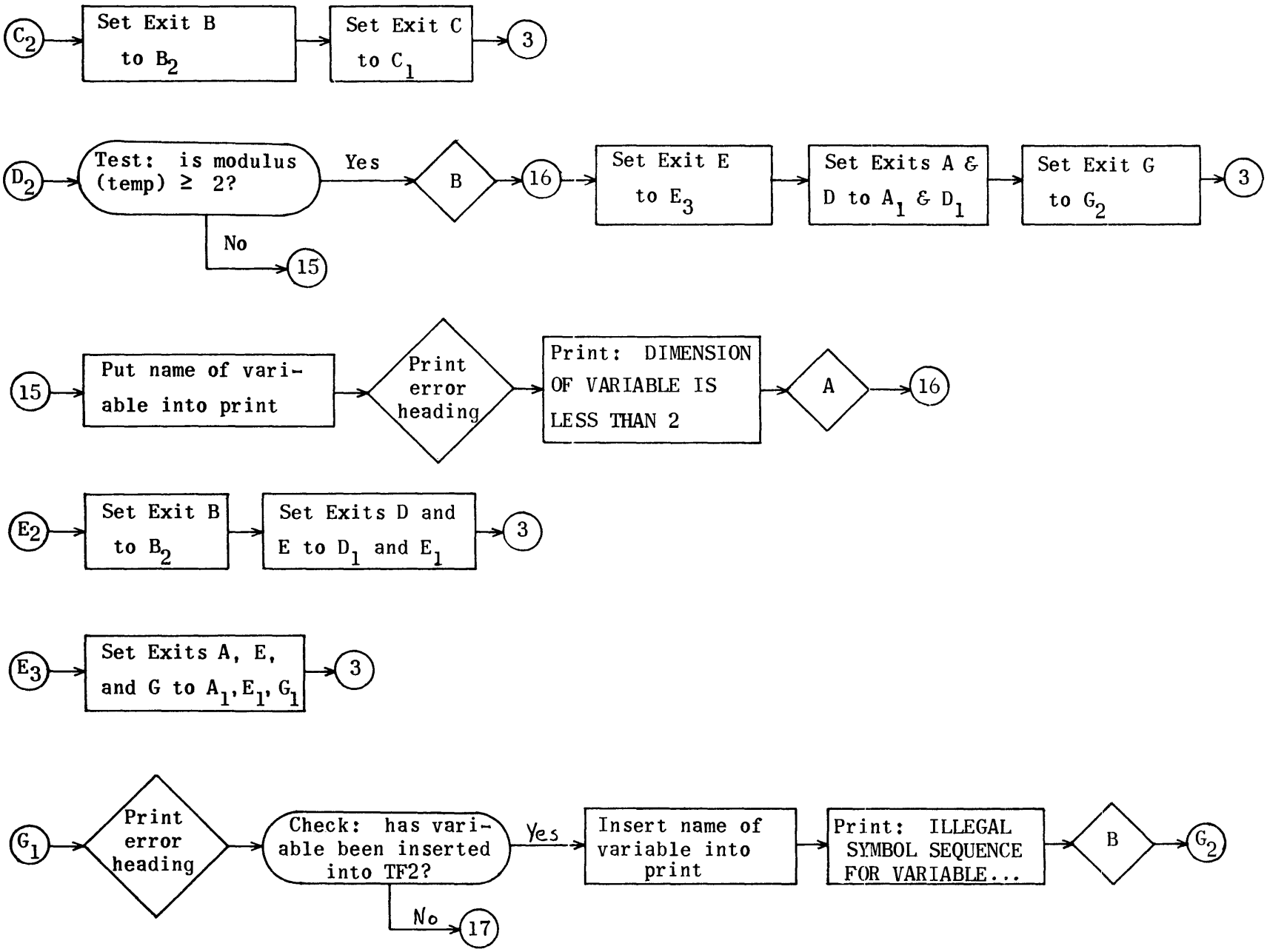
If a modulus for a variable is less than 2, the routine prints: DIMENSION OF VARIABLE '_ _ _' IS LESS THAN 2.

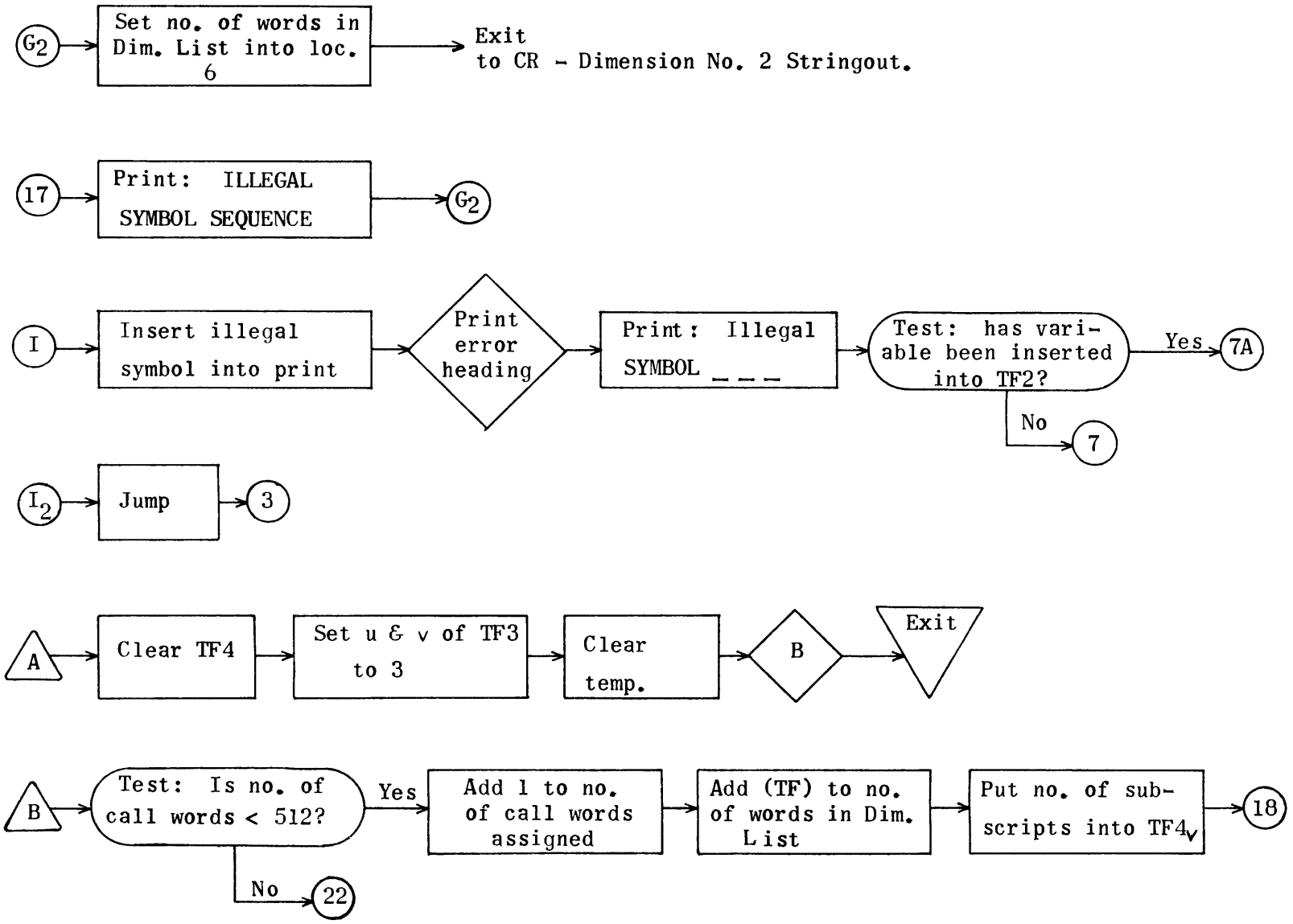
When an error is found for a variable, the symbols following the variable are ignored until a new variable name is found. A file containing only the variable name, drum address and call word is added to the Dimension List.

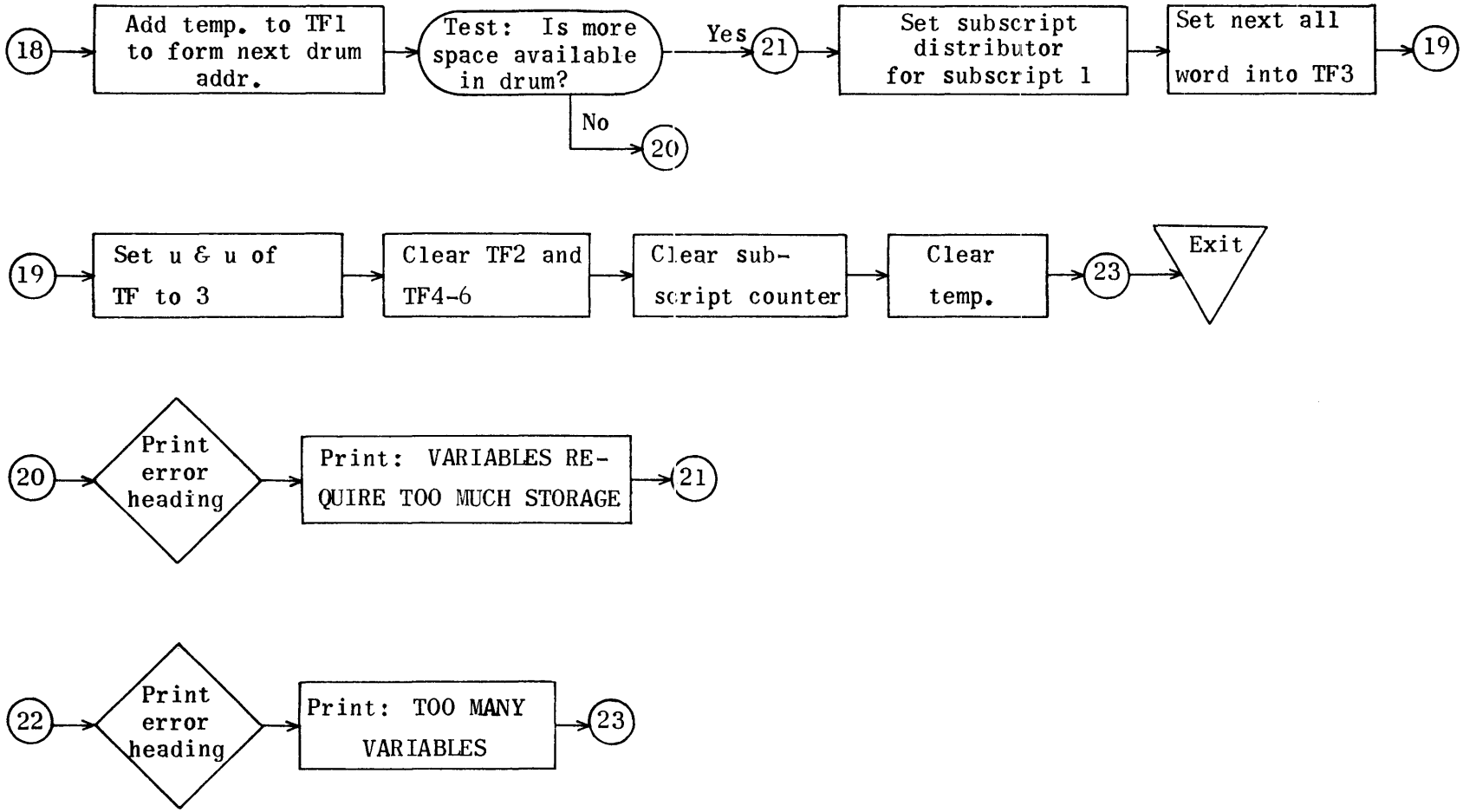

Start
Clear TF thru TF6
Set initial drum address
Set initial call word
clear counters 1, 2, & 3
Set u & v of TF to 3
2
2
Set exits A to A1 B to B1
Set exits C, D, and E to C1D1E1
Set exit G to G1
3
3
Get next symbol
Test: Is 1st char. a letter?
Yes
Test: Is letter I, J, K, L, or M
Yes
Exit I
No
4
No
Exit A
A1
A2
4
Test: Is 1st char. a decimal point or digit?
No
Test: Is symbol a (?
No
Test: Is symbol a) ?
No
Test: is symbol a, or ; ?
No
5
Yes
Exit B
B1
B2
Yes
Exit C
C1
C2
Yes
Exit C
D1
D2
Yes
Exit E
E1
E2
E3
5
Test: Is symbol a Δ.?
No
Exit I
Yes
Exit G


A1
WA Check variable symbol
Put variable symbol into file (TF2)
Set Exit C to C_2
Set exit B to B_1
Set Exits D and E to D_1 and E_1
6
6
Set Exit G to G_1
Set Exit A to A_2
3
A_2
WA Print Error heading
Insert name of variable (TF2)
Print: INFORMATION FOR VARIABLE . . . IS INCOMPLETE
A
A_1
B_1
C_1
D_1
E_1
WA Print error heading
7A
Insert name of variable
Print: ILLEGAL SYMBOL SEQUENCE FOR VARIABLE . . .
A
7
7
Set Exits B, C, D, and E to I_2
7B
Set Exit G to G_2
Set Exit A to A_1
3
B_2
RD Check fixed point symbol
RS Convert to octal
Test: Is sub-script cntr. ≤ 3?
Yes
Add 1 to subscript cntr
No
14
Distributor for subscripts
1st subscript
8
2nd "
11
3rd "
12
4th "
13


8
Shift subscript left 17 places in A
9
Store A in temp. and in TF4u
Increment u & v of TF by 1
10
Set Exit B to B1
Set exits D and E to D2 and E2
3
11
Store subscript in temp + 1 and in TF5v
(Temp) (Temp + 1) to A
9
12
(Subscript) (Temp) to temp and to TF4u
(Subscript) (Temp +1) to temp + 1 and to TF5v
Shift subscript left 17 places in A
Store A in temp + 2 and in TF5v
10
13
Store subscript in TF6v
(Subscript) (Temp + 1) to TF5v
(Subscript) (Temp + 2) to TF5u
(Subscript) (Temp) to A
9
14
Insert name of variable into print
WA
Print Error Heading
Print: Too many subscripts for variable...
A
Set Exits B, C, D, & E to I2
Set A to A1
7B


C2
Set Exit B to B2
Set Exit C to C1
3
D2
Test: is modulus (temp) ≥ 2?
Yes
B
16
Set Exit E to E3
Set Exits A & D to A1 & D1
Set Exit G to G2
3
No
15
15
Put name of variable into print
Print error heading
Print: DIMENSION OF VARIABLE IS LESS THAN 2
A
16
E2
Set Exit B to B2
Set Exits D and E to D1 and E1
3
E3
Set Exits A, E, and G to A1, E1, G1
3
G1
Print error heading
Check: has variable been inserted into TF2?
Yes
Insert name of variable into print
Print: ILLEGAL SYMBOL SEQUENCE FOR VARIABLE...
B
G2
No
17


G2
Set no. of words in Dim. List into loc. 6
Exit to CR - Dimension No. 2 Stringout.
17
Print: ILLEGAL SYMBOL SEQUENCE
G2
I
Insert illegal symbol into print
Print error heading
Print: Illegal SYMBOL _ _ _
Test: has variable been inserted into TF2?
Yes
7A
No
7
I2
Jump
3
A
Clear TF4
Set u & v of TF3 to 3
Clear temp.
B
Exit
B
Test: Is no. of call words < 512?
Yes
Add 1 to no. of call words assigned
Add (TF) to no. of words in Dim. List
Put no. of subscripts into TF4v
18
No
22


18
Add temp. to TF1 to form next drum addr.
Test: Is more space available in drum?
Yes
21
Set subscript distributor for subscript 1
Set next all word into TF3
19
No
20
19
Set u & u of TF to 3
Clear TF2 and TF4-6
Clear sub-script counter
Clear temp.
23
Exit
20
Print error heading
Print: VARIABLES RE-QUIRE TOO MUCH STORAGE
21
22
Print error heading
Print: TOO MANY VARIABLES
23

Dimension String-Out No. 1

```
      RE   DY4400
      RE   DT4424
      RE   DU4502
      RE   DV4564
      RE   DW4622
      RE   DX4643
      RE   DZ4707
      RE   CR5075
```

String-Out Subroutine Regions Are Needed to Assemble This Tape

```
          IA   DV
    0     RP   10007   DV2     } Clear TF-TF6
    1     TP   DZ      TF      }
    2     TP   DZ1     TF1     Set initial drum address
    3     TP   DZ2     TF3     Set initial call word
    4     RP   10003   DV6     }
    5     TP   DZ      DZ111   } Clear counters
    6     TP   DZ22    TF      Set u & V of TF to 3
    7     RJ   DX      DX      Set exit A to A1
   10     RJ   DX16    DX16    Set exit B to B1
   11     RJ   DV17    DV14    Set exits C, D & E to C1, D1, & E1.
   12     RJ   DY2     DY2     Set exit G to G1
   13     MJ   0       DW      Back to get next symbol
   14     TV   DW16    DW7
   15     TV   DW16    DW10
   16     TV   DW16    DW15
   17     MJ   0       30000

   20     RJ   DW15    30000   Set exit E to E2
E1 21     RJ   DU      DU      Set exit B to B2
   22     TV   DZ24    DW10    Set exit D to D1
   23     TV   DZ24    DW15    Set exit E to E1
   24     MJ   0       DW      Back to get next symbol
   25     RJ   DW15    30000   Set exit E to E3
E3 26     RJ   DX      DX      Set exit A to A1
   27     TV   DZ24    DW15    Set exit E to E1

   30     RJ   DY2     DY2     Set G to G1
   31     MJ   0       DW      Back to get next symbol
   32     RJ   DW7     30000   Set exit C to C2
C2 33     RJ   DU      DU      Set exit B to B2
   34     TV   DZ24    DW7     Set exit C to C1
   35     MJ   0       DW      Back to get next symbol
          CA   DV36
```

```
        IA    DY
    0   MJ    0       CR        Exit to Dim. String-out No. 2
    1   MJ    0       DV        Jump to start
    2   RJ    DW13    30000     Set exit G to G1
G1  3   RJ    WA      WA1       Print Error heading
    4   TP    TF2     A         Test: Has variable name been inserted into
    5   ZJ    DY6     DY13      TF2?
    6   TP    A       DZ60      If so, insert name of variable into print.
    7   TP    DZ11    UP3       Print: ILLEGAL SYMBOL SEQUENCE FOR VARIABLE
    10  RJ    UP2     UP        _ _ _
    11  RJ    DT55    DT51      Send incomplete file to Combination List
    12  MJ    0       DY17      Jump to G2
    13  TP    DZ12    UP3       Print: ILLEGAL SYMBOL SEQUENCE
    14  RJ    UP2     UP
    15  MJ    0       DY17      Jump to G2
    16  RJ    DW13    30000     Set exit G to G2
G2  17  SP    DZ112   17
    20  SA    DZ117   0         Set location 6 No. of words in Dim. List to u
    21  TU    A       6
    22  TV    DZ113   6                       No. of call words to v
    23  MJ    0       DY        Jump to exit
        CA    DY24

        IA    DT
    0   TP    DZ113   A         Test: Have less than 512 call words been
    1   TJ    DZ20    DT6       assigned?
    2   RJ    WA      WA1       If not, print error heading
    3   TP    DZ14    UP3       Print: TOO MANY VARIABLES
    4   RJ    UP2     UP
    5   MJ    0       DT25      Jump to subroutine exit.
    6   RA    DZ113   DZ21
    7   RA    DZ112   TF

    10  TV    DZ111   TF4
    11  RJ    TE      TE1       Send file to combination list
    12  RA    TF1     DZ114
    13  TJ    DZ26    DT15
    14  MJ    0       DT45
    15  RJ    DU7     DU7
    16  RA    TF3     DZ21
    17  TP    DZ22    TF

    20  TP    DZ      TF2
    21  RP    10003   DT23
    22  TP    DZ      TF4
    23  TP    DZ      DZ111     Clear subscript counter
    24  TP    DZ      DZ114     Clear Temp.
    25  MJ    0       30000     Subroutine exit
    26  RJ    DW10    30000
```

D_2	27	TP	DZ114	A	Test: Is modulus ≥ 2?
	30	TJ	DZ25	DT37	
	31	RJ	DT25	DT	If so, send file to Comb. List & set for next file
	32	RJ	DV25	DV25	Set exit E to E_3
	33	RJ	DX	DX	Set exit A to A_1
	34	RJ	DY16	DY16	Set exit G to G_2
	35	TV	DX24	DW10	Set exit D to D_1
	36	MJ	0	DW	Back to get next symbol.
	37	TP	TF2	DZ77	Put name of variable into print
	40	RJ	WA	WA1	Print error heading
	41	TP	DZ17	UP3	Print: DIMENSION OF VARIABLE '_ _ _' IS LESS THAN 2
	42	RJ	UP2	UP	
	43	RJ	DT55	DT51	Send incomplete file to Combination List
	44	MJ	0	DT32	Back to set new exits
	45	RJ	WA	WA1	Print error heading
	46	TP	DZ16	UP3	Print: VARIABLES REQUIRE TOO MUCH STORAGE
	47	RJ	UP2	UP	
	50	MJ	0	DT15	
	51	TP	DZ	TF4	
	52	TP	DZ22	TF	
	53	TP	DZ	DZ114	
	54	RJ	DT25	DT	
	55	MJ	0	30000	
		CA	DT56		
		IA	DU		
	0	RJ	DW16	30000	
B_2	1	RJ	RD	RD1	Check fixed pt. symbol
	2	TP	SY2	RS4	Convert XS3 to octal
	3	RJ	RS2	RS	
	4	TP	DZ27	A	Test: Is no. of subscripts ≤ 3
	5	TJ	DZ111	DU51	
	6	RA	DZ111	DZ21	If so, add 1 to subscript counter
	7	RJ	DU7	DU10	Distributor for subscripts
	10	RJ	DU7	DU14	
	11	RJ	DU7	DU24	
	12	RJ	DU7	DU30	
	13	MJ	0	DU42	
8	14	LA	RS3	17	1st subscript to Temp. and to $TF4_u$
	15	TP	A	DZ114	
	16	TU	A	TF4	
	17	RA	TF	DZ23	Increment TF
	20	RJ	DX16	DX16	Set B to B_1, E to E_2, D to D_2
	21	RJ	DV20	DV20	
	22	RJ	DT26	DT26	
	23	MJ	0	DW	

```
11          24   TP   RS3     DZ115   }   2nd subscript to temp.+ 1 and to TF5v
            25   TV   RS3     TF5     }
            26   MP   DZ114   DZ115       (Temp.) (Temp. + 1) → A
            27   MJ   0       DU15

            30   MP   DZ114   RS3     }   3rd subscript x temp. → temp. and to
            31   TP   A       DZ114   }   TF4u
            32   TU   A       TF4     }
            33   MP   DZ115   TS3     }   3rd subscript x temp. 1 → temp. and to
            34   TP   A       DZ115   }   TF5v
            35   TV   A       TF5     }
            36   LA   RS3     17      }
            37   TP   A       DZ116   }   3rd subscript to temp. + 2u and TF5u
            40   TU   A       TF5     }
            41   MJ   0       DU20
            42   TV   RS3     TF6         4th subscript to TF6v
            43   MP   DZ115   RS3     }   (4th subscript) temp. + 1 → TF5v
            44   TV   A       TF5     }
            45   MP   DZ116   RS3     }   (4th subscript) temp. + 2 → TF5u
            46   TU   A       TF5     }
            47   MP   DZ114   TS3         (4th subscript) temp. → A

            50   MJ   0       DU15
            51   TP   TF2     DZ36        Insert name of variable into print
            52   RJ   WA      WA1         Print error heading
            53   TP   DZ15    UP3     }   Print:  TOO MANY SUBSCRIPTS FOR
            54   RJ   UP2     UP      }   VARIABLE _ _ _
            55   RJ   DT55    DT51        Send incomplete file to Combination List
            56   RJ   DW17    DW17        Set B to I2
            57   RJ   DV17    DV14        Set C, D, & E to I2
            60   RJ   DX      DX          Set exit A to A1
            61   MJ   0       DX30
                 CA   DU62

                 IA   DW
             0   RJ   SY      SY1         Get next symbol
             1   TP   SY7     Q       }   Test:  Is 1st char. a letter?
             2   QJ   DW3     DW5     }
             3   TP   SY10    Q       }   Test:  Is 1st char. I, J, K, L, or M?
Exit A       4   QJ   DW14    30000   }
             5   TP   SY11    Q       }   Test:  Is 1st char. decimal point or
             6   QJ   DW16    DW7     }   digit?
Exit C       7   EJ   DZ3     30000       Test:  Is symbol (?

Exit D      10   EJ   DZ4     30000       Test:  Is symbol )?
            11   EJ   DZ5     DW15        Test:  Is symbol ,?
            12   EJ   DZ6     DW15        Test:  Is symbol ;?
Exit G      13   EJ   DZ7     30000       Test:  Is symbol Δ .?
Exit I      14   MJ   0       DX36        Illegal symbol exit
Exit E      15   MJ   0       30000
Exit B      16   MJ   0       30000
            17   RJ   DW16    30000       Set exit B to I2
```

Exit I_2	20	MJ	0	DW	
		CA	DW21		
		IA	DX		
	0	RJ	DW4	30000	Set exit A to A_1
A_1	1	RJ	RH	RH1	Check variable symbol
	2	TP	SY2	TF2	Put variable symbol in file
	3	RJ	DV32	DV32	Set exit C to C_2
	4	RJ	DX16	DX16	Set exit B to B_1
	5	RJ	DV17	DV15	Set exits D and E to D_1 and E_1
	6	RJ	DY2	DY2	Set exit G to G_1
	7	RJ	DW4	DW	Set exit A to A_2 & go to pick up next symbol
A_2	10	RJ	WA	WA1	Print error heading
	11	TP	TF2	DZ45	Insert name of variable into print
	12	TP	DZ10	UP3	Print: INFORMATION FOR VARIABLE _ _ _
	13	RJ	UP2	UP	IS INCOMPLETE
	14	RJ	DT55	DT51	Send incomplete file to Combination List
	15	MJ	0	DX1	Jump to A_1
	16	RJ	DW16	30000	Set exit B to B_1
B,C,D,E	17	RJ	WA	WA1	Print error heading
	20	TP	TF2	A	Test: Has variable been inserted into
	21	ZJ	DX22	DX33	TF2?
(7A)	22	TP	A	DZ60	If not, put name of variable into print
	23	TP	DZ11	UP3	Print: ILLEGAL SYMBOL SEQUENCE FOR
	24	RJ	UP2	UP	VARIABLE _ _ _
	25	RJ	DT55	DT51	Send incomplete file to Combination List
(7)	26	RJ	DW17	DW17	Set B to I_2
	27	RJ	DV17	DV14	Set C, D, and E to I_2
	30	RJ	DY16	DY16	Set G to G_2
	31	RJ	DX	DX	Set A to A_1
	32	MJ	0	DW	Back to get next symbol
	33	TP	DZ12	UP3	Print: ILLEGAL SYMBOL SEQUENCE
	34	RJ	UP2	UP	
	35	MJ	0	DX26	Back to (7)
	36	TP	SY2	DZ65	Insert illegal symbol into error print
	37	RJ	WA	WA1	Print error heading
	40	TP	DZ13	UP3	Print: ILLEGAL SYMBOL _ _ _
	41	RJ	UP2	UP	
	42	TP	TF2	A	
	43	ZJ	DX22	DX26	
		CA	DX44		
		IA	DZ		
	0	0	0	0	
	1	0	40001	0	
	2	0	0	77000	
	3	17	77777	77777	
	4	43	77777	77777	

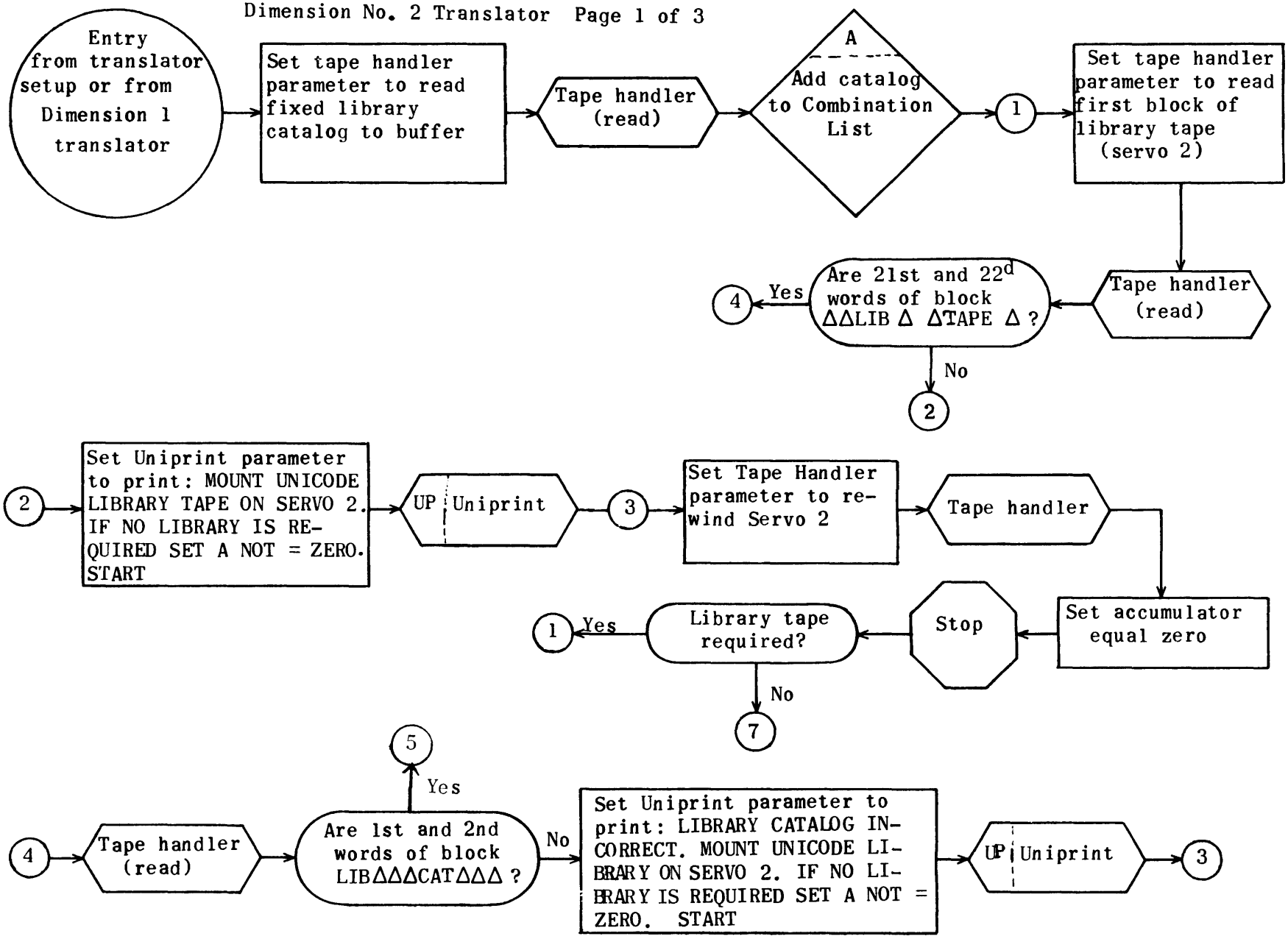
```
5     21    77777    77777
6     23    77777    77777
7     01    22777    77777

10    40    DZ40     11      ┐
11    40    DZ51     11      │
12    40    DZ51     4       │
13    40    DZ62     5       │   U P parameters
14    40    DZ67     4       │
15    40    DZ30     10      │   for alarm
16    40    DZ103    6       │
17    40    DZ73     10      ┘   prints

20    0     0        1000
21    0     0        1
22    0     3        3
23    0     1        1
24    0     0        DX17
25    0     2        0
26    0     77777    0
27    0     0        3

30    66    51510    14724   ┐   XS3 codes
31    50    73016    56725   │   TOO MANY SUBSCRIPTS FOR VARIABLES ---
32    65    26543    45266   │
33    65    01315    15401   │
34    70    24543    42425   │
35    46    30014    37777   │
36    01    01010    10101   │
37    17    22777    77777   ┘

40    34    50315    15447   ┐
41    24    66345    15001   │
42    31    51540    17024   │
43    54    34242    54630   │   INFORMATION FOR VARIABLE --- IS INCOMPLETE
44    01    43777    77777   │
45    01    01010    10101   │
46    17    01346    50134   │
47    50    26514    75246   │
                             │
50    30    66302    27777   ┘
51    34    46463    03224   ┐
52    46    01657    34725   │
53    51    46016    53053   │
54    67    30502    63001   │
55    31    51540    17024   │
56    54    34242    54630   │   ILLEGAL SYMBOL SEQUENCE FOR VARIABLE ---
57    01    43777    77777   │
                             │
60    01    01010    10101   │
61    17    22777    77777   ┘
62    34    46463    03224   ┐
```

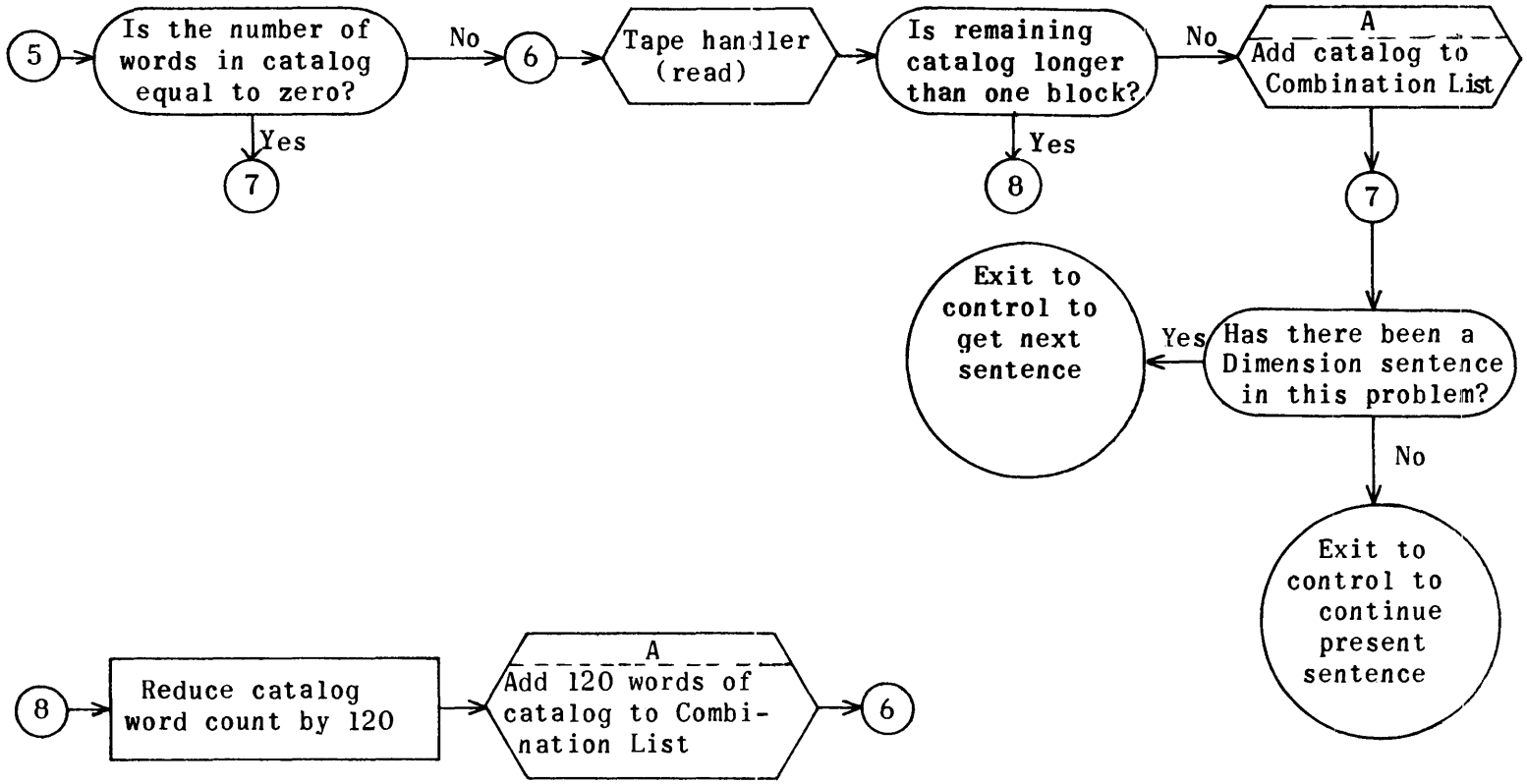
```
63    46    01657    34725
64    51    46014    37777
65    01    01010    10101    ILLEGAL SYMBOL ---
66    17    22777    77777
67    66    51510    14724

70    50    73017    02454    TOO MANY VARIABLES
71    34    24254    63065
72    22    77777    77777
73    27    34473    05065
74    34    51500    15131
75    01    70245    43424
76    25    46300    14377
77    01    01010    10101    DIMENSION OF VARIABLE --- IS LESS THAN 2

100   17    01346    50146
101   30    65650    16633
102   24    50010    52277
103   70    24543    42425
104   46    30650    15430
105   53    67345    43001
106   66    51510    14767    VARIABLES REQUIRE TOO MUCH STORAGE
107   26    33016    56651

110   54    24323    02277
111   0     0        0        Subscript counter
112   00    0        0        No. words in dim. list counter
113   0     0        0        No. call words counter
114   0     0        0        Temp.
115   0     0        0        Temp. + 1
116   0     0        0        Temp. + 2
117   0     20000    0

      CA    DZ120
```

Dimension No. 2 Translator.

This translator reads the Permanent Library Catalog from the master tape and the Catalog for the UNICODE Library from Uniservo 2 and inserts them in the Combination List. Since these Catalogs must follow the Dimension items (if any) in the Combination List, this Translator is entered from the Dimension 1 Translator if a Dimension sentence appears in the problem. Otherwise, it is entered from the Translation Set-up routine.

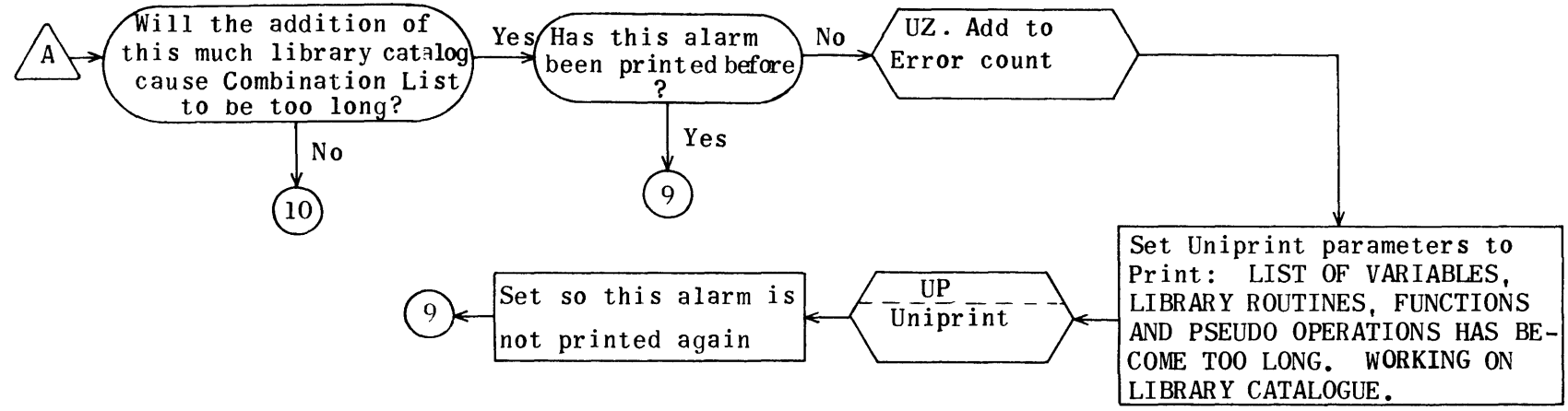
This routine makes appropriate checks on the labels of the Uniservo 2 Library tape to insure that, if present, it is positioned properly. If a Library tape is not required for the problem and is not mounted on Uniservo 2, compilation will continue after suitable indication is given.

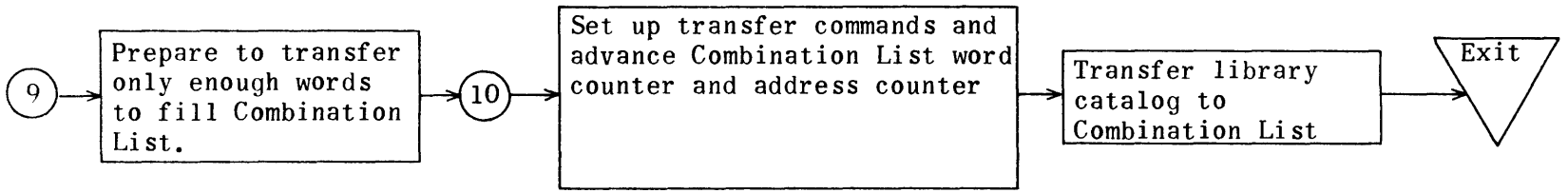

Entry from translator setup or from Dimension 1 translator
Set tape handler parameter to read fixed library catalog to buffer
Tape handler (read)
A
Add catalog to Combination List
1
Set tape handler parameter to read first block of library tape (servo 2)
Tape handler (read)
Are 21st and 22d words of block ΔΔLIB Δ ΔTAPE Δ ?
Yes
4
No
2
2
Set Uniprint parameter to print: MOUNT UNICODE LIBRARY TAPE ON SERVO 2. IF NO LIBRARY IS REQUIRED SET A NOT = ZERO. START
UP Uniprint
3
Set Tape Handler parameter to rewind Servo 2
Tape handler
Set accumulator equal zero
Stop
1
Yes
Library tape required?
No
7
5
Yes
4
Tape handler (read)
Are 1st and 2nd words of block LIBΔΔΔCATΔΔΔ ?
No
Set Uniprint parameter to print: LIBRARY CATALOG INCORRECT. MOUNT UNICODE LIBRARY ON SERVO 2. IF NO LIBRARY IS REQUIRED SET A NOT = ZERO. START
UP Uniprint
3


5
Is the number of words in catalog equal to zero?
No
6
Tape handler (read)
Is remaining catalog longer than one block?
No
A
Add catalog to Combination List
Yes
7
Yes
8
7
Exit to control to get next sentence
Yes
Has there been a Dimension sentence in this problem?
No
Exit to control to continue present sentence
8
Reduce catalog word count by 120
A
Add 120 words of catalog to Combination List
6

Dimension No. 2 Translator Page 3 of 3

Subroutine A Add Library Catalog to Combination List.





Dimension No. 2 Translator.

Regions

```
RE  CR5075
RE  CQ5131
RE  CV5151
RE  CW5177
RE  CX5214
RE  CY5270
RE  CZ5272
RE  CU46101        Numerical Constant

RE  UP421    )
RE  TE2663   |
RE  CB40101  |
RE  TH21     |
RE  WL3507   }     String-Out Subroutines Used
RE  CT714    |
RE  UZ3067   |
RE  TB2637   |
RE  FH50624  )
```

```
     IA   CR
 0   TP   CW6     TH3     Read fixed library catalog to Buffer CZ
 1   RJ   TH2     TH
 2   TP   CZ      CY      Length --> temp.
 3   TU   CR27    CV25    Set up Transfer
 4   RJ   CV24    CV      Add fixed library catalog to CB list
 5   TP   CW7     TH3     Read standard library to Buffer CZ
 6   RJ   TH2     TH
 7   SP   CZ24    0       Check for Δ Δ L I B Δ
10   EJ   CW10    CR12
11   MJ   0       CR14
12   SP   CZ25    0       Check for Δ T A P E Δ
13   EJ   CW11    CR23
14   TP   CX1     UP3     MOUNT LIBRARY, etc.
15   RJ   UP2     UP
16   TP   CW5     TH3     Rewind Uniservo 2.
17   RJ   TH2     TH
20   SP   CW      0       Set A = 0
21   MS   0       CR22
22   ZJ   CQ11    CR5     Library tape required?
23   RJ   TH2     TH      Read block of Library tape
24   TP   CZ      A       Check for L I B Δ Δ Δ
25   EJ   CW12    CR27
26   MJ   0       CR31
27   TP   CZ1     A       Check for C A T Δ Δ Δ
30   EJ   CW13    CQ
31   TP   CX      UP3     LIBRARY CATALOG INCORRECT, etc.
32   RJ   UP2     UP
33   MJ   0       CR16
     CA   CR34
```

```
     IA   CQ
 0   SP   CZ2     0       Third word of block --> A.
 1   ZJ   CQ2     CQ11    If no. of words = 0, exit
 2   TP   A       CY1
 3   TU   CR24    CV25    Set up transfer
 4   RJ   TH2     TH      Read block of Library tape
 5   SP   CW1     0
 6   TJ   CY1     CQ14    Is catalog longer than one block?
 7   TP   CY1     CY      No. words --> temp.
10   RJ   CV24    CV      Transfer to CB list
11   TP   WL2     A       Has there been a Dimension sentence?
12   EJ   CW14    CT      If yes, jump to get next sentence
13   MJ   0       CT13    If no, jump to continue present sentence.
14   RS   CY1     CW1     Reduce count by 170
15   TP   CW1     CY      Use 170 count
16   RJ   CV24    CV      Transfer to CB list
17   MJ   0       CQ4     Back for next block.
     CA   CQ20
```

```
    IA   CV
0   SP   CY      17
1   AT   CY      CY
2   SA   TE2     0        }
3   TJ   CW4     CV16     }   Will CB List be too long?
4   TP   TE3     Q        }
5   QJ   CV14    CV6      }   Has this alarm been printed before?
6   RJ   UZ      UZ1      }   No, so add to error count
7   TP   CX46    UP3      }   and print alarm
10  RJ   UP2     UP       }
11  TP   CX47    UP3      }
12  RJ   UP2     UP       }
13  TP   TB5     TE3          Set so this alarm is not printed again.
14  SP   CW3     0        }
15  ST   TE2     CY       }   Prepare to transfer partial block
16  TP   CW2     Q        }
17  QS   CY      CV24     }   Set n of RP
20  SP   CB      0        }
21  QA   CY      CB       }   Advance CB word counter
22  TV   TE2     CV25         Set up transfer
23  RA   TE2     CY           Advance CB list address counter
24  RP   30000   30000    }
25  TP   30000   30000    }   Transfer Library Catalog to
                               CB List.
    CA   CV26
```

```
    IA   CW
 0  0    0       0
 1  0    0       170
 2  0    7777    0
 3  0    CU      CU           1st word of List following CB List
 4  0    CU1     CU1
 5  10   2       0            Rewind Uniservo 2.
 6  50   101     CZ           Read one block of Uniservo 1.
 7  50   102     CZ           Read one block of Uniservo 2.
10  01   01463   42501        Δ Δ L I B Δ
11  01   66245   23001        Δ T A P E Δ
12  46   34250   10101        L I B Δ Δ Δ
13  26   24660   10101        C A T Δ Δ Δ
14  27   34473   05065        D I M E N S
    CA   CW15
```

```
     IA   CX
0    0    CX2     25
1    0    CX27    17
2    46   34255   42454      L I B R A R
3    73   01262   46624      Y Δ C A T A
4    46   51326   73001      L O G U E Δ
5    46   24253   04601      L A B E L Δ
6    34   50265   15454      I N C O R R
7    30   26662   20101      E C T . Δ Δ
10   47   51675   06601      M O U N T Δ
11   67   50342   65127      U N I C O D
12   30   01463   42554      E Δ L I B R
13   24   54730   15150      A R Y Δ O N
14   01   65305   47051      Δ S E R V O
15   01   05220   10134      Δ 2 . Δ Δ I
16   31   01505   10101      F Δ N O Δ Δ
17   01   01463   42554      Δ Δ L I B R
20   24   54730   13465      A R Y Δ I S
21   01   54305   36734      Δ R E Q U I
22   54   30270   16530      R E D Δ S E
23   66   01240   15051      T Δ A Δ N O
24   66   01760   10322      T Δ = Δ O .
25   01   01656   62454      Δ Δ S T A R
26   66   22010   10101      T . Δ Δ Δ Δ
27   47   51675   06601      M O U N T Δ
30   67   50342   65127      U N I C O D
31   30   01463   42554      E Δ L I B R
32   24   54730   15150      A R Y Δ O N
33   01   65305   47051      Δ S E R V O
34   01   05220   10134      Δ 2 . Δ Δ I
35   31   01505   10146      F Δ N O Δ L
36   34   25542   45473      I B R A R Y
37   01   34650   15430      Δ I S Δ R E
40   53   67345   43027      Q U I R E D
41   01   65306   60124      Δ S E T Δ A
42   01   50516   60176      Δ N O T Δ =
43   01   03220   10101      Δ O . Δ Δ Δ
44   01   01656   62454      Δ Δ S T A R
45   66   22010   10101      T . Δ Δ Δ Δ
46   0    FH23    21         Refers to Print-out in String-out Subs.
47   40   CX50    4
50   50   01463   42554      N Δ L I B R
51   24   54730   12624      A R Y Δ C A
52   66   24465   13267      T A L O G U
53   30   22777   77777      E .
     CA   CX54
```

Compute String-Out.

The string-out that this routine produces contains the call words of the alpha-numeric symbols of compute sentences. They appear in the same sequence as the symbols appear in the input sentence. Only subscripted variables are represented by more than their call words; these also have a modulus plus a number of subscripts and their multipliers in the string-out. A subscripted variable therefore occupies 2, 3, or 4 rows in the string-out, depending on whether it has 1, 2-3, or 4 subscripts. Parentheses, commas, periods and the like are not saved, except for the final space-period, which appears in excess-three representation and indicates the end of the string-out.

The compute sentence allows the combination of several terms within one sentence, such as: COMPUTE A(I) AND DOT (F,B (J), -20.4) AND W Δ. The string-out places a zero line between the terms within the sentence and after the last one, just preceding the space-period. This zero line is important for the compute-generate routine; it gives the indicator to place the RJ with the ten-line.

Variables that cannot be found in the Combination List or Dummy List are assigned call words by this routine and added to the Combination List. Only functions and subscripted variables are assumed to be in the lists, or an alarm occurs.

The Compute String-out routine checks for the type faults, wrongly chosen symbols, incorrect format of operands in functions or pseudo operations, and for the wrong number of subscripts of subscripted variables. Furthermore it checks for all sorts of errors induced by the combination of several symbols. The model for these error checks is given in the appendix to the flow chart.

When the first symbol of the sentence following the word "compute" is not alpha-numeric, the routine prints an alarm and exits without further checking of the sentence. This is the only case where the routine skips the rest of the sentence. In all other cases the sentence is handled up to the very end (if the number of alarms does not exceed the limit) and only contents of parentheses may be skipped, where a preceding alarm made the checking of the symbols inside the parentheses meaningless.

In cases where single-valued variables are used in place of subscripts, or vice versa, the routine gives an alarm but assumes that this format mistake was just incorrect writing and therefore adds the correct bit to the function or subscripted variable format-set-up in order to go on checking those expressions.

The routine handles negative floating point numbers inside functions and inside pseudo operations. In all other cases negative signs are rejected.

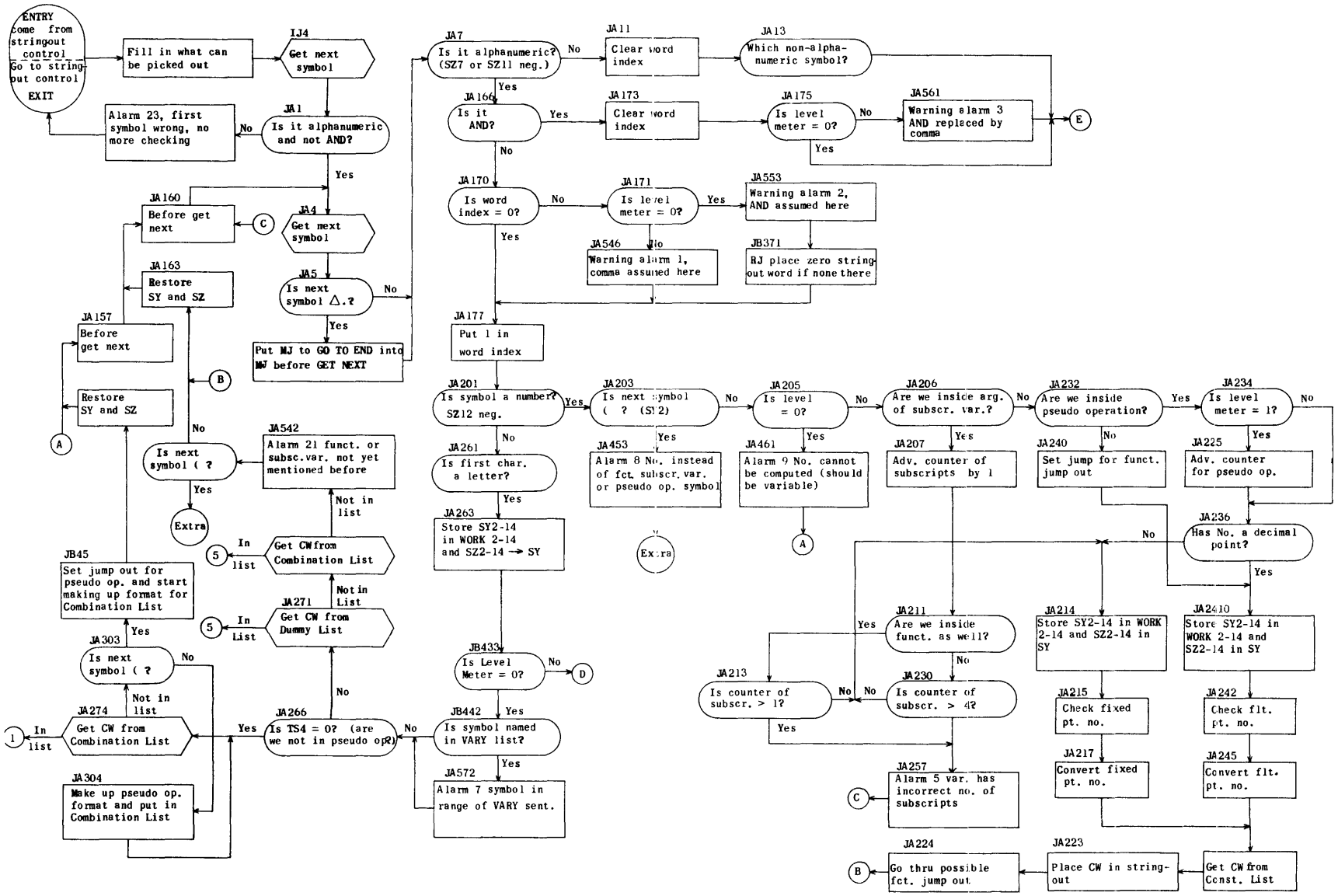
These are the 23 alarm print-outs and three warnings:

1) Symbol [ ] illegal in compute sentence.
2) Adjacent symbols [ ][ ] meaningless in compute sentence.
3) Pseudo operation [ ] has too many elements.
4) Parentheses not properly paired.
5) Variable [ ] has incorrect number of subscripts.
6) Arguments of function [ ] do not agree with previous usage.
7) [ ] cannot be computed here, since the sentence is in the range of a vary sentence.
8) Constant $\begin{bmatrix} \\ \end{bmatrix}$ is followed by open parenthesis.
9) Constant $\begin{bmatrix} \\ \end{bmatrix}$ appears as variable to be computed.
10) Arguments of function [ ] do not appear.
11) Arguments of function [ ] are superfluous.
12) Alpha-numeric symbol [ ] has digit as first character.
13) Symbol [ ] incorrectly used.
14) Variable [ ] is not defined by an equation.
15) Single valued variable [ ] is followed by an open parenthesis.
16) Subscripts of variable [ ] do not appear.
17) Arguments of dummy function [ ] do not appear.
18) Format of pseudo operation [ ] does not agree with previous usage.
19) Subscript [ ] of variable [ ] is not properly written.
20) Argument [ ] of function [ ] appears as subscript.
21) Function or subscripted variable [ ] has not been previously mentioned.
22) Subscript [ ] of variable [ ] exceeds allowed number of subscripts.
23) First symbol is wrong. Sentence not further checked.

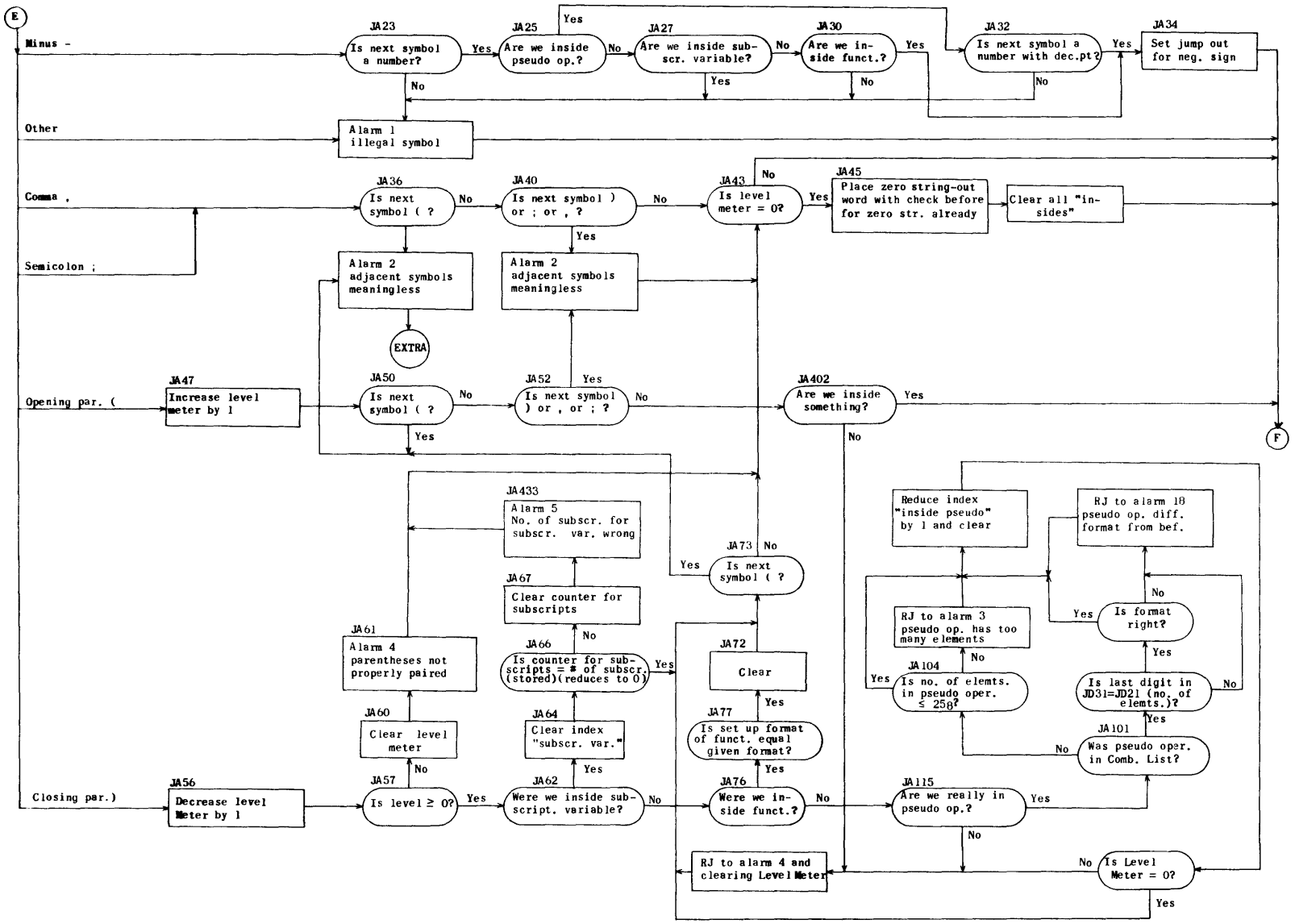
Warnings:

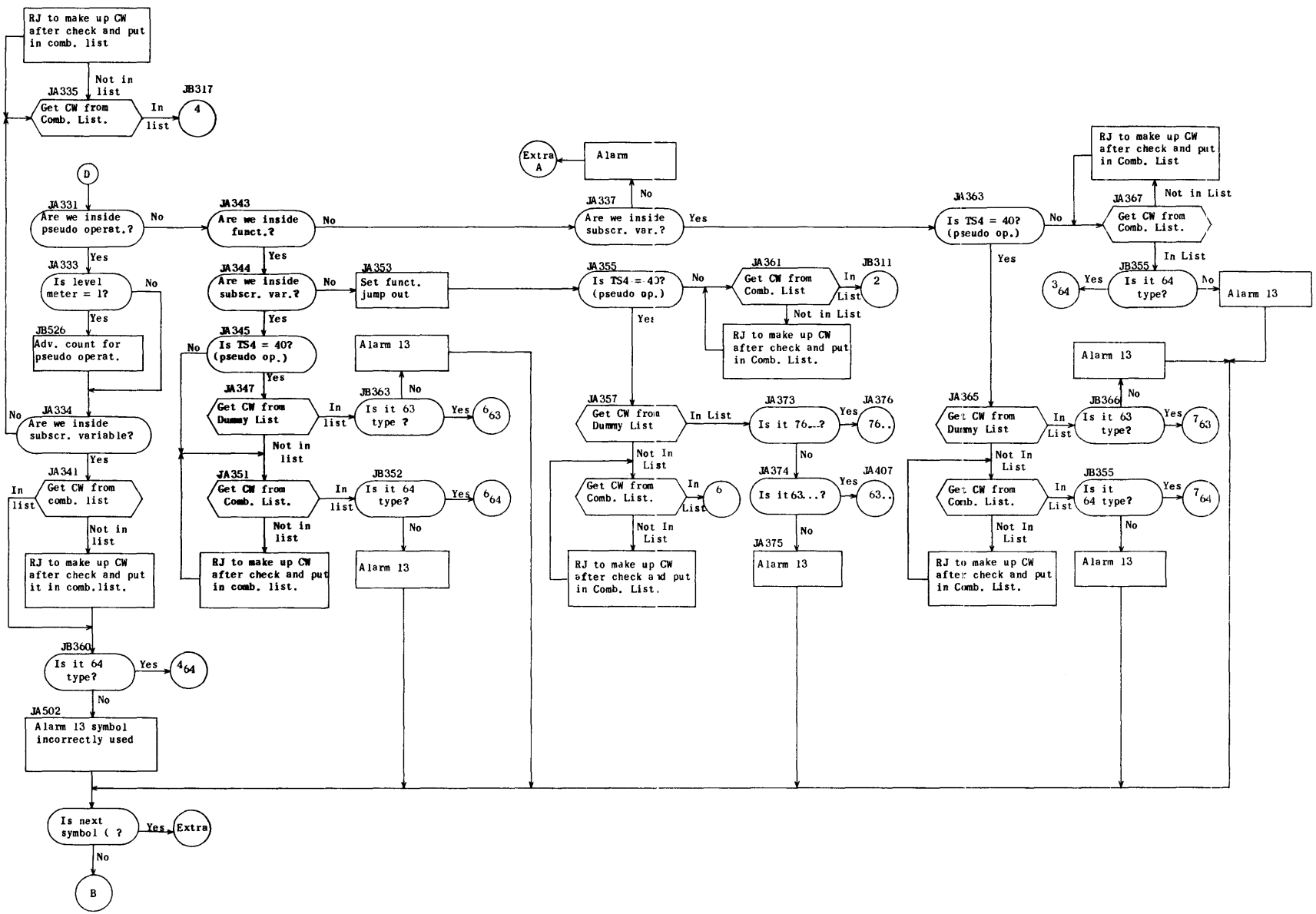
1) Comma assumed between $\begin{bmatrix} \\ \end{bmatrix}$ and preceding symbol.
2) "and" assumed between $\begin{bmatrix} \\ \end{bmatrix}$ and preceding symbol.
3) "and" inside parentheses replaced by comma.

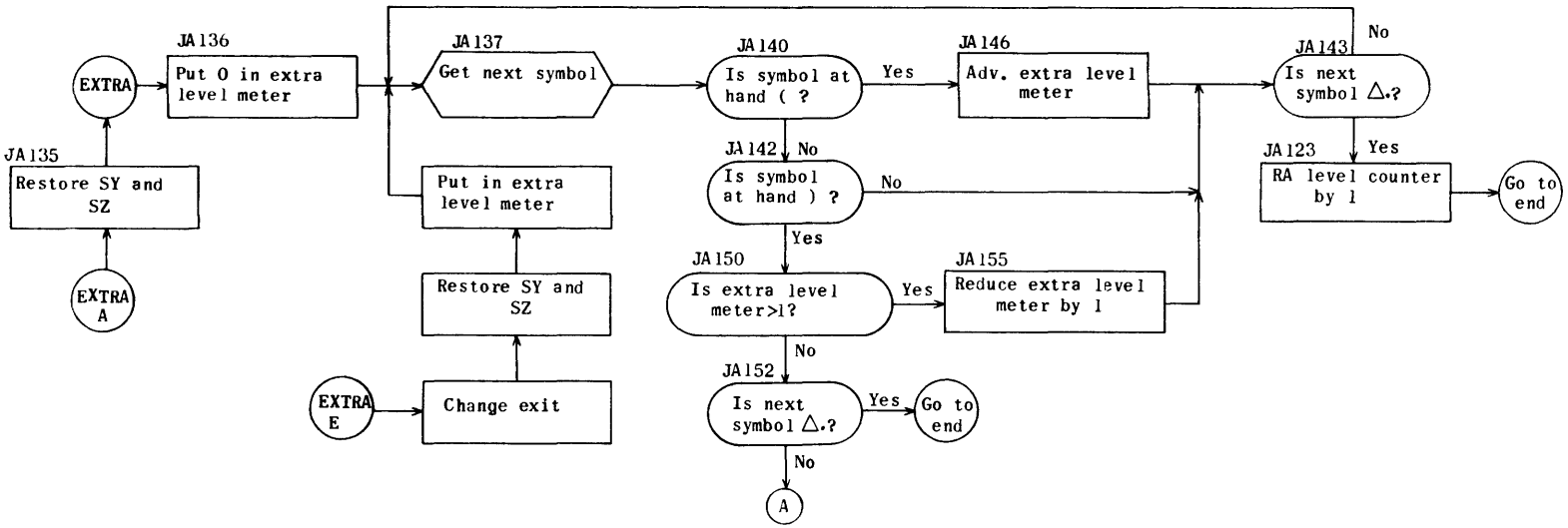
COMPUTE String-out

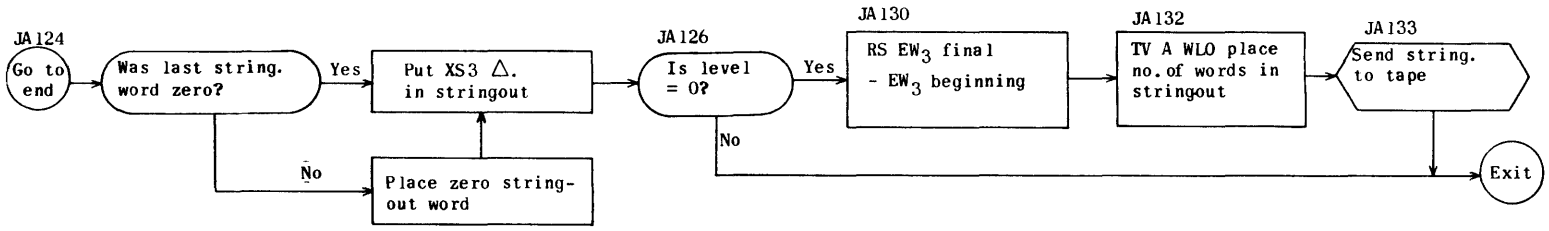


COMPUTE String-out (Cont.)

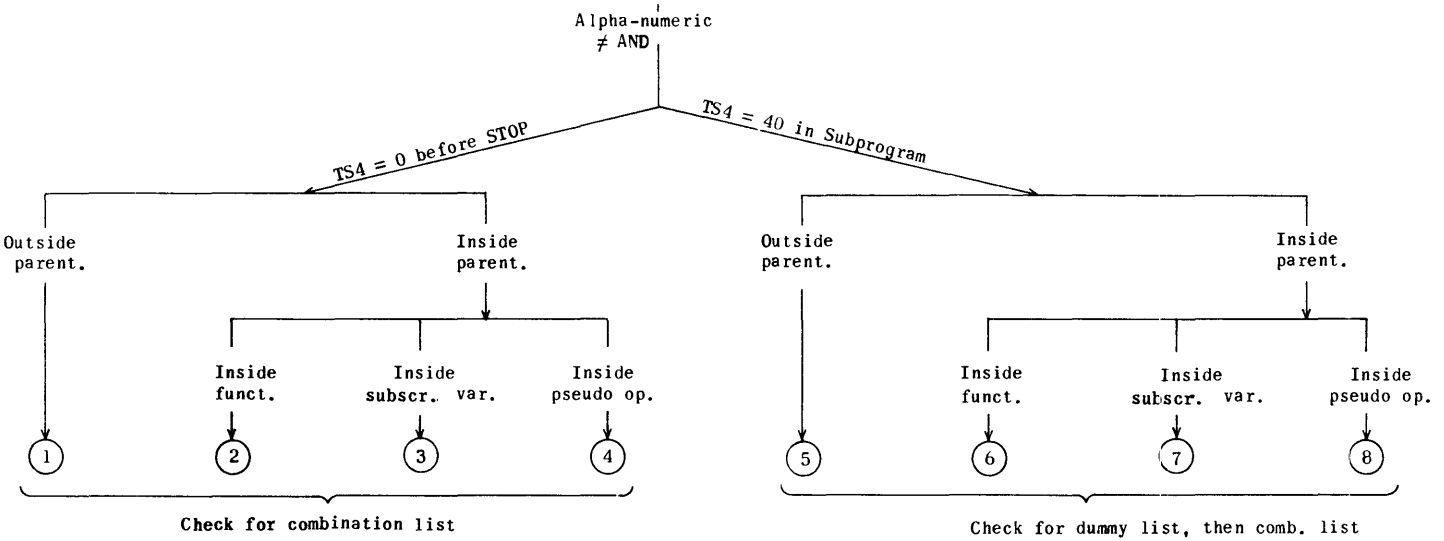


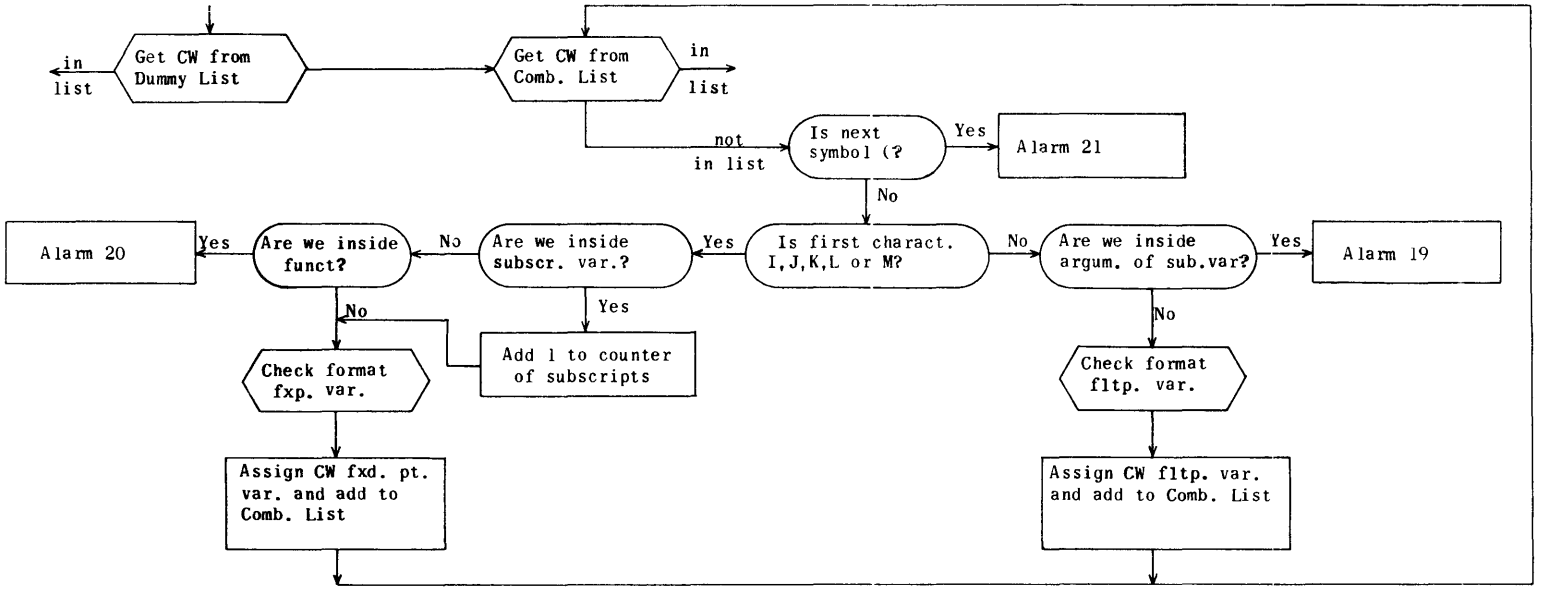

RJ to make up CW after check and put in comb. list
JA335
Get CW from Comb. List.
Not in list
In list
JB317
4
D
JA331
Are we inside pseudo operat.?
No
Yes
JA333
Is level meter = 1?
No
Yes
JB526
Adv. count for pseudo operat.
JA334
Are we inside subscr. variable?
No
Yes
JA341
Get CW from comb. list
In list
Not in list
RJ to make up CW after check and put it in comb.list.
JB360
Is it 64 type?
Yes
4_{64}
No
JA502
Alarm 13 symbol incorrectly used
Is next symbol (?
Yes
Extra
No
B
JA343
Are we inside funct.?
No
Yes
JA344
Are we inside subscr. var.?
No
Yes
JA345
Is TS4 = 40? (pseudo op.)
No
Yes
JA347
Get CW from Dummy List
In list
Not in list
JB363
Is it 63 type ?
No
Yes
6_{63}
JA351
Get CW from Comb. List.
In list
Not in list
JB352
Is it 64 type?
Yes
No
6_{64}
RJ to make up CW after check and put in comb. list.
JA353
Set funct. jump out
Alarm 13
Alarm 13
Extra A
Alarm
JA337
Are we inside subscr. var.?
No
Yes
JA355
Is TS4 = 40? (pseudo op.)
No
Yes
JA361
Get CW from Comb. List
In List
Not in List
JB311
2
RJ to make up CW after check and put in Comb. List.
JA357
Get CW from Dummy List
In List
Not In List
JA373
Is it 76...?
Yes
No
JA376
76..
Get CW from Comb. List.
In List
Not In List
6
JA374
Is it63...?
Yes
No
JA407
63..
JA375
Alarm 13
RJ to make up CW after check and put in Comb. List.
JA363
Is TS4 = 40? (pseudo op.)
No
Yes
JA367
Get CW from Comb. List.
Not in List
In List
RJ to make up CW after check and put in Comb. List
JB355
Is it 64 type?
Yes
No
3_{64}
Alarm 13
Alarm 13
JA365
Get CW from Dummy List
In List
Not In List
JB366
Is it 63 type?
No
Yes
7_{63}
Get CW from Comb. List.
In List
Not In List
JB355
Is it 64 type?
Yes
No
7_{64}
RJ to make up CW after check and put in Comb. List.
Alarm 13

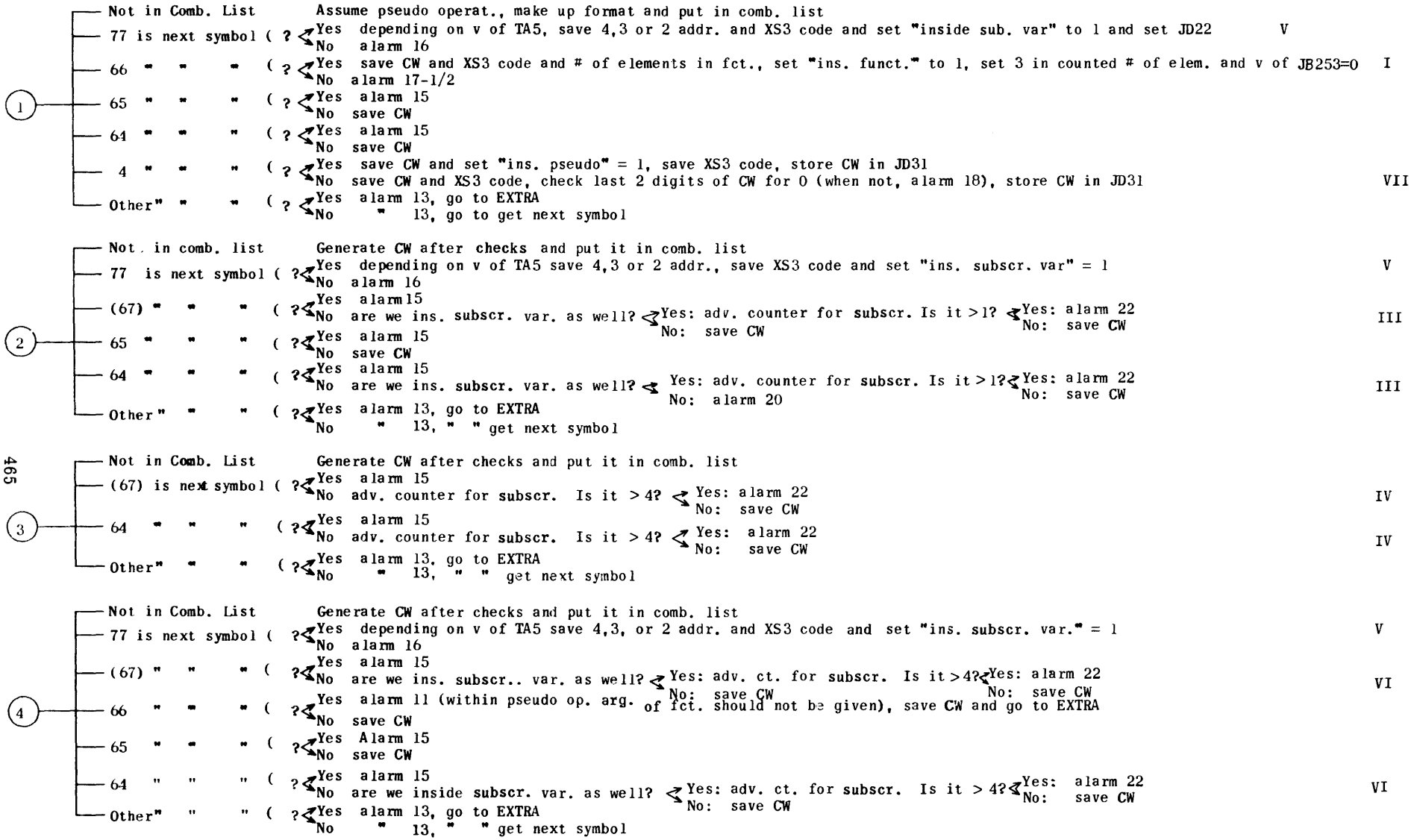

EXTRA
JA136
Put 0 in extra level meter
JA137
Get next symbol
JA140
Is symbol at hand (?
Yes
JA146
Adv. extra level meter
No
JA143
Is next symbol △.?
Yes
JA123
RA level counter by 1
Go to end
JA135
Restore SY and SZ
EXTRA A
Put in extra level meter
JA142
Is symbol at hand) ?
No
Yes
JA150
Is extra level meter>1?
Yes
JA155
Reduce extra level meter by 1
No
JA152
Is next symbol △.?
Yes
Go to end
No
A
Restore SY and SZ
EXTRA E
Change exit

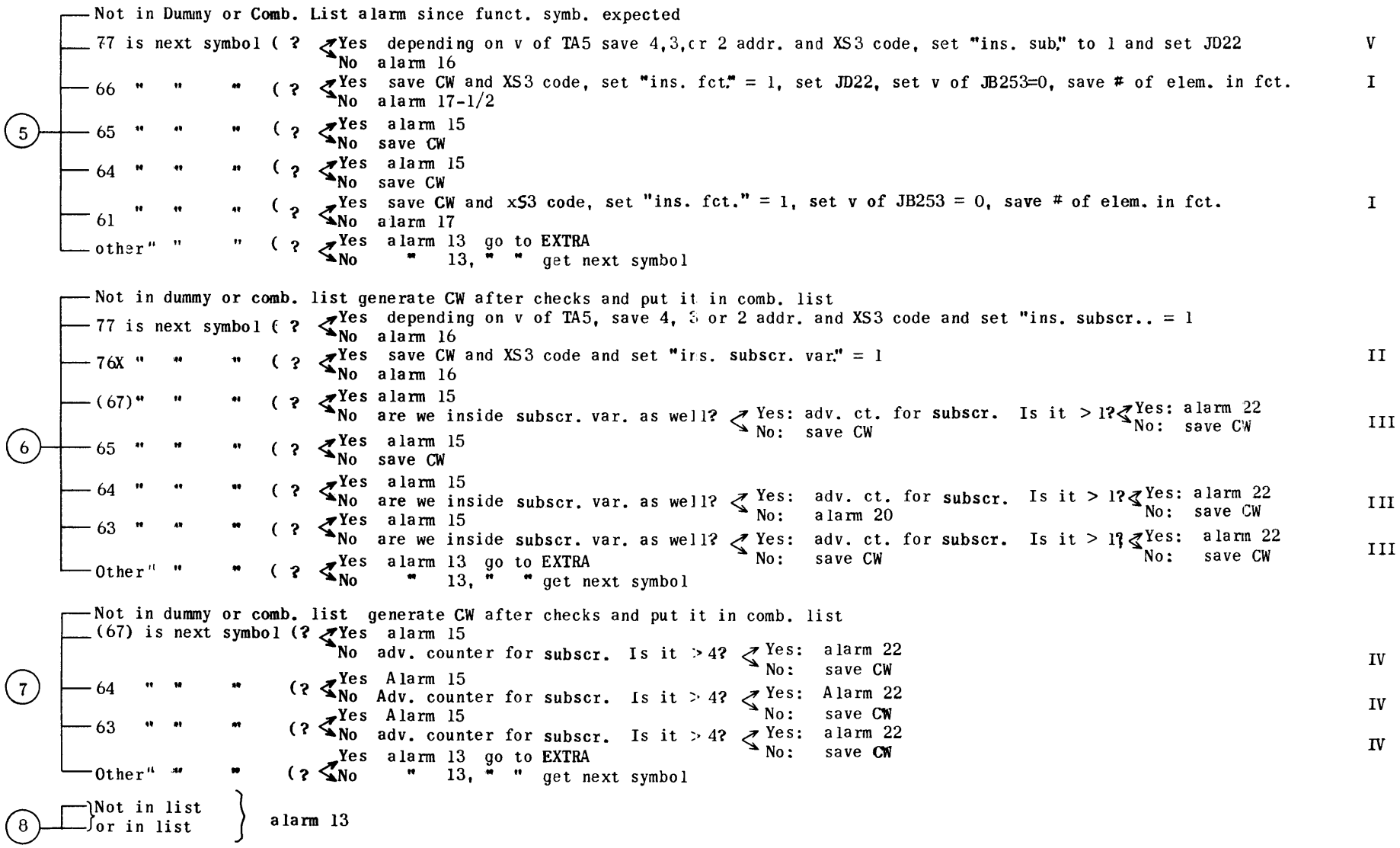

JA124
Go to end
Was last string. word zero?
Yes
Put XS3 △. in stringout
No
Place zero string-out word
JA126
Is level = 0?
Yes
JA130
RS EW3 final - EW3 beginning
JA132
TV A WLO place no. of words in stringout
JA133
Send string. to tape
No
Exit

COMPUTE STRINGOUT (Con't.)






1
Not in Comb. List — Assume pseudo operat., make up format and put in comb. list
77 is next symbol (? Yes depending on v of TA5, save 4,3 or 2 addr. and XS3 code and set "inside sub. var" to 1 and set JD22 V
No alarm 16
66 " " " (? Yes save CW and XS3 code and # of elements in fct., set "ins. funct." to 1, set 3 in counted # of elem. and v of JB253=0 I
No alarm 17-1/2
65 " " " (? Yes alarm 15
No save CW
64 " " " (? Yes alarm 15
No save CW
4 " " " (? Yes save CW and set "ins. pseudo" = 1, save XS3 code, store CW in JD31
No save CW and XS3 code, check last 2 digits of CW for 0 (when not, alarm 18), store CW in JD31 VII
Other" " " (? Yes alarm 13, go to EXTRA
No " 13, go to get next symbol
2
Not in comb. list — Generate CW after checks and put it in comb. list
77 is next symbol (? Yes depending on v of TA5 save 4,3 or 2 addr., save XS3 code and set "ins. subscr. var" = 1 V
No alarm 16
(67) " " " (? Yes alarm15
No are we ins. subscr. var. as well? Yes: adv. counter for subscr. Is it >1? Yes: alarm 22
No: save CW No: save CW III
65 " " " (? Yes alarm 15
No save CW
64 " " " (? Yes alarm 15
No are we ins. subscr. var. as well? Yes: adv. counter for subscr. Is it >1? Yes: alarm 22
No: alarm 20 No: save CW III
Other" " " (? Yes alarm 13, go to EXTRA
No " 13, " " get next symbol
3
Not in Comb. List — Generate CW after checks and put it in comb. list
(67) is next symbol (? Yes alarm 15
No adv. counter for subscr. Is it > 4? Yes: alarm 22
No: save CW IV
64 " " " (? Yes alarm 15
No adv. counter for subscr. Is it > 4? Yes: alarm 22
No: save CW IV
Other" " " (? Yes alarm 13, go to EXTRA
No " 13, " " get next symbol
4
Not in Comb. List — Generate CW after checks and put it in comb. list
77 is next symbol (? Yes depending on v of TA5 save 4,3, or 2 addr. and XS3 code and set "ins. subscr. var." = 1 V
No alarm 16
(67) " " " (? Yes alarm 15
No are we ins. subscr.. var. as well? Yes: adv. ct. for subscr. Is it > 4? Yes: alarm 22
No: save CW No: save CW VI
66 " " " (? Yes alarm 11 (within pseudo op. arg. of fct. should not be given), save CW and go to EXTRA
No save CW
65 " " " (? Yes Alarm 15
No save CW
64 " " " (? Yes alarm 15
No are we inside subscr. var. as well? Yes: adv. ct. for subscr. Is it > 4? Yes: alarm 22
No: save CW No: save CW VI
Other" " " (? Yes alarm 13, go to EXTRA
No " 13, " " get next symbol


Not in Dummy or Comb. List alarm since funct. symb. expected
77 is next symbol (? Yes depending on v of TA5 save 4,3,or 2 addr. and XS3 code, set "ins. sub." to 1 and set JD22 V
No alarm 16
66 " " " (? Yes save CW and XS3 code, set "ins. fct." = 1, set JD22, set v of JB253=0, save # of elem. in fct. I
No alarm 17-1/2
5
65 " " " (? Yes alarm 15
No save CW
64 " " " (? Yes alarm 15
No save CW
61 " " " (? Yes save CW and xS3 code, set "ins. fct." = 1, set v of JB253 = 0, save # of elem. in fct. I
No alarm 17
other " " (? Yes alarm 13 go to EXTRA
No " 13, " " get next symbol
Not in dummy or comb. list generate CW after checks and put it in comb. list
77 is next symbol (? Yes depending on v of TA5, save 4, 3 or 2 addr. and XS3 code and set "ins. subscr.. = 1
No alarm 16
76X " " " (? Yes save CW and XS3 code and set "ins. subscr. var." = 1 II
No alarm 16
(67)" " " (? Yes alarm 15
No are we inside subscr. var. as well? Yes: adv. ct. for subscr. Is it > 1? Yes: alarm 22
No: save CW No: save CW III
6
65 " " " (? Yes alarm 15
No save CW
64 " " " (? Yes alarm 15
No are we inside subscr. var. as well? Yes: adv. ct. for subscr. Is it > 1? Yes: alarm 22
No: alarm 20 No: save CW III
63 " " " (? Yes alarm 15
No are we inside subscr. var. as well? Yes: adv. ct. for subscr. Is it > 1? Yes: alarm 22
No: save CW No: save CW III
Other " " " (? Yes alarm 13 go to EXTRA
No " 13, " " get next symbol
Not in dummy or comb. list generate CW after checks and put it in comb. list
(67) is next symbol (? Yes alarm 15
No adv. counter for subscr. Is it > 4? Yes: alarm 22
No: save CW IV
7
64 " " " (? Yes Alarm 15
No Adv. counter for subscr. Is it > 4? Yes: Alarm 22
No: save CW IV
63 " " " (? Yes Alarm 15
No adv. counter for subscr. Is it > 4? Yes: alarm 22
No: save CW IV
Other " " " (? Yes alarm 13 go to EXTRA
No " 13, " " get next symbol
8
Not in list
or in list
alarm 13

Compute String-Out
Regions

```
    RE    IJ4400
    RE    JA4410
    RE    JB5234
    RE    JC5770
    RE    JI6114
    RE    JJ6126
    RE    JK6145
    RE    JL6157
    RE    JM6167
    RE    JO6202
    RE    JP6221
    RE    JQ6243
    RE    JR6257
    RE    JS6276
    RE    JT6310
    RE    JU6322
    RE    JV6340
    RE    JW6350
    RE    JX6363
    RE    JY6401
    RE    JZ6413
    RE    IA6426
    RE    IB6446
    RE    IC6466
    RE    ID6506
    RE    IE6527
    RE    IH6551
    RE    IF6565
    RE    IG6603
    RE    IK6617
    RE    II6631
    RE    TJ6643
    RE    JD6673
```

String-Out Subroutine regions are also needed to assemble this tape

```
    IA    IJ
0   MJ    0       CT          Exit
1   RP    10035   IJ3 }
2   TP    JC20    JD  }       Clear temporaries
3   TP    JC1     EW3         Set EW3
4   RJ    SY      SY1         Get first symbol
5   EJ    JC2     JA567       Is it AND? When yes, go to alarm 23.
6   MJ    0       JA          No, go to JA
    CA    IJ7
```

```
                  IA    JA
              0   TP    JC14    A       "40 Ø Ø" ⟶A
              1   EJ    SY7     JA4   } Is first symbol alpha-numeric?
              2   EJ    SY11    JA4   }
              3   MJ    0       JA567   No:  go to alarm 23
              4   RJ    SY      SY1     Get next symbol
              5   EJ    JC30    JA21    Is next symbol Δ .?
              6   TP    JC14    A       Put "40 Ø Ø"⟶A
              7   EJ    SZ7     JA165 } Is symbol at hand alpha-numeric?
             10   EJ    SZ11    JA165 }
             11   TP    JC20    JD      Clear word index
Not Alpha-   12   TP    SZ2     A       Put XS3 code of word at hand ⟶A
 numeric     13   EJ    JC34    JA36    ,
             14   EJ    JC35    JA36    ;
             15   EJ    JC13    JA47    (
             16   EJ    JC27    JA56    )
             17   RP    20002   JA413 } - (upper and lower case)
             20   EJ    JC104   JA23  }
Go to END    21   TV    JC61    JA160   Set exit for GO TO END
             22   MJ    0       JA6     Go to check symbol at hand
Neg. Sign    23   TP    JC14    A     } Has next symbol a letter?
             24   EJ    SZ12    JA612 }
             25   TP    JC20    A     } Are we inside pseudo op?
             26   TJ    JD17    JA32  }
             27   TJ    JD16    JA612   Are we inside subscr. var.?
             30   TJ    JD15    JA34    Are we inside funct.?
             31   MJ    0       JA612   None of them:  go to alarm 1
             32   TJ    SY6     JA34    Has next symbol decimal point?
             33   MJ    0       JA612   Go to alarm 1
             34   TV    JC5     JA246   Set for handl. neg. no.
             35   MJ    0       JA157   Go to get next symbol
Comma ,      36   TP    SY2     A     } Is next symbol (?
Semicolon ;  37   EJ    JC13    JA417 }
             40   EJ    JC27    JA425   Is next symbol )?
             41   EJ    JC34    JA425   Is next symbol ;?
             42   EJ    JC35    JA425   Is next symbol ,?
             43   TP    JC20    A     } Is level meter ≠ 0?
             44   TJ    JD1     JA157 }
             45   RJ    JB400   JB371   Place zero string-out word and clear
                                        counters
             46   MJ    0       JA157   Go to get next string-out word
Opening      47   RA    JD1     JC21    Adv. Level Meter
 Parent.     50   TP    SY2     A     } Is next symbol (?
             51   EJ    JC13    JA417 }
             52   EJ    JC27    JA425   Is next symbol ) ?
             53   EJ    JC34    JA425   Is next symbol ;?
             54   EJ    JC35    JA425   Is next symbol ,?
             55   MJ    0       JA401   Go to further checks
Closing      56   RS    JD1     JC21    Reduce Level Meter
 Parent.     57   SJ    JA60    JA62    Is Level Meter ≥ 0?
             60   TP    JC20    JD1     No Level Meter neg; clear Level Meter
             61   MJ    0       JA427   Go to alarm 4
```

```
                62   TP   JC20    A      } Are we not inside subscr. var.?
                63   EJ   JD16    JA76   }
                64   TP   JC20    JD16     We are inside; clear index "inside subscr."
                65   RS   JD22    JD20     Subtract counted no. of subscr. from
                                             given no. in storage.
                66   ZJ   JA67    JA72     Is no. of subscripts correct?
                67   TP   JC20    JD20   } No, clear counter of subscr. and stored
                70   TP   JC20    JD22   } no. of subscript.
                71   MJ   0       JA433    Go to alarm 5
                72   TP   JC20    JD20     Yes, no. of subscr. O.K.: clear counter
                                             of subscripts
                73   TP   SY2     A      } Is next symbol (?
                74   EJ   JC13    JA417  } Go to alarm 2
                75   MJ   0       JA43
                76   EJ   JD15    JA115    Are we not inside funct.?
                77   RS   JD32    JD33   } Is set-up format of funct. = given format?
               100   ZJ   JA120   JA121  }
Pseudo         101   EJ   JD24    JA113    Come from JA115. Was pseudo oper. not in
 Operation                                   Combination List?
               102   TP   JC67    Q      } Mask out last 2 digits
               103   QT   JD31    A      }
               104   EJ   JD21    JB505    Is no. of operands in pseudo op. correct?
               105   RJ   JB414   JB411    Clear Level Meter and counter of elements
                                             and inside pseudo
               106   MJ   0       JA440    Go to alarm 6-1/2
               107   RJ   JB414   JB412    Clear index inside pseudo and counter of
                                             elements.
               110   TP   JC20    A      } Is Level Meter = 0?
               111   EJ   JD1     JA72   }
               112   MJ   0       JA576    Go to alarm (4), emergency exit
               113   RS   JD21    JC31     Come from JA101 } check for no. of ele-
                                                            } ments ≤ 20_10
               114   SJ   JA107   JA445                    }
               115   TJ   JD17    JA101    Come from JA76 } Are we really in pseudo
                                                          } op.? When not, some-
               116   MJ   0       JA576                  } thing wrong, emergency
               117   0    0       0        Free            exit
               120   RJ   JB424   JA564    Alarm 5-1/2
               121   TP   JC20    JD15     Clear counter "inside funct."
               122   MJ   0       JA72
Before         123   RA   JD1     JC21
 GO TO END
GO TO END      124   RJ   JB461   JB453  } Put XS3 code of Δ . in string-out after
               125   RJ   EW      EW1    } checking whether zero word was already
                                             placed
               126   TP   JC20    A      } Is Level Meter = 0?
               127   TJ   JD1     JA451  }
               130   RS   EW3     JC62   }
               131   AT   JC21    A      } Set no. of words in string-out in v of WLO
               132   TV   A       WL     }
               133   RJ   WT      WT1      Bring string-out to tape
               134   MJ   0       IJ       Jump to EXIT
```

```
Before EXTRA   135   RJ   JB410   JB406    Restore SY and SZ
EXTRA          136   TP   JC20    JD30     Put 0 in extra Level Meter
               137   RJ   SY      SY1      Get next symbol
               140   TP   SZ2     A        } Is symbol at hand (?
               141   EJ   JC13    JA146    }
               142   EJ   JC27    JA150    Is symbol at hand )?
               143   TP   SY2     A        } Is next symbol Δ .?
               144   EJ   JC30    JA123    }
               145   MJ   0       JA137
               146   RA   JD30    JC21     Adv. extra Level Meter
               147   MJ   0       JA143    Go to check next symbol Δ .?
               150   TP   JC21    A        } Is extra Level Meter  >1?
               151   TJ   JD30    JA155    }
               152   TP   JC30    A        } Is next symbol  Δ .?
               153   EJ   SY2     JA124    }
               154   MJ   0       JA157    Restore and go to get next symbol
               155   RS   JD30    JC21     Reduce extra Level Meter
               156   MJ   0       JA143    Jump to check next symbol Δ .?
               157   MJ   0       JA160    Before "go to get next symbol"
               160   MJ   0       JA4      Go to get next symbol
               161   RJ   JB410   JB406    Restore SY and SZ
               162   MJ   0       JA160    Go directly to get next symbol
               163   RJ   JB410   JB406    Restore SY and SZ
               164   MJ   0       JA157    Go to "before go to get next symbol"
                                            (used as const. for JB515)
               165   TP   SZ2     A        } Is symbol at hand AND?
               166   EJ   JC2     JA173    }
               167   TP   JC20    A        } Is word index = 0?
               170   EJ   JD      JA177    }
               171   EJ   JD1     JA553    Is Level Meter = 0?  Yes; warning alarm
                                            2
               172   MJ   0       JA546    Go to warning alarm 1
               173   TP   JC20    JD       Clear word index
               174   TP   JC20    A        } Is Level Meter = 0?
               175   EJ   JD1     JA36     }
               176   MJ   0       JA561    Go to warning alarm 3
               177   TP   JC21    JD       Put 1 in word index
               200   TP   JC14    A        } Has symbol at hand a letter?
               201   EJ   SZ12    JA261    }
               202   TP   SY2     A        } Is next symbol (?
               203   EJ   JC13    JA453    }
               204   TP   JC20    A        } Is level = 0?
               205   EJ   JD1     JA461    }
               206   EJ   JD16    JA232    Are we not inside subscr. var.?
               207   RA   JD20    JC21     Adv. counter for subscripts by 1
               210   TP   JC20    A        } Are we not inside function?
               211   EJ   JD15    JA227    }
               212   TP   JC21    A        } Is counter of subscripts  > 1?
               213   TJ   JD20    JA257    }
Fixed Point No. 214  RJ   JB405   JB401    Save SY2-14 in JD2-14 and put SZ2-14
                                            ⟶ SY2-14
               215   RJ   RD      RD1      Check fixed pt. const. format
```

	216	TP	SZ2	RS4	Convert to octal (fixed point)
	217	RJ	RS2	RS	
	220	TP	RS3	A	Get CW for const. from Const. List
	221	RJ	GW	GW1	
	222	TP	Q	EW2	Bring CW to string-out
	223	RJ	EW	EW1	
	224	MJ	0	JB222	Jump to possible funct. jump out
	225	RA	JD21	JC21	Adv. counter of elements in pseudo
	226	MJ	0	JA235	
	227	TP	JC24	A	Is counter of subscripts > 4?
	230	TJ	JD20	JA257	
	231	MJ	0	JA214	
	232	EJ	JD17	JA240	Are we not inside pseudo?
	233	TP	JC21	A	Is Level Meter = 1?
	234	EJ	JD1	JA225	
	235	TP	JC20	A	Has no. a decimal point?
	236	TJ	SZ6	JA241	
	237	MJ	0	JA214	
	240	TV	JC101	JB222	Set jump for exit out of function
	241	RJ	JB405	JB401	Store SY, SZ
	242	RJ	RB	RB1	Check format floating point constant
	243	TP	SY2	GG4	
	244	TP	SY3	GG5	Convert to floating point octal
	245	RJ	GG2	GG	
	246	RJ	JA246	JA247	Inserted for eventual TN GG3 GG3 (?)
	247	TP	GG3	A	Go to assign CW and store it
	250	MJ	0	JA221	
	251	TN	GG3	A	Case of neg. floating point no.
	252	MJ	0	JA221	
Exit When	253	RJ	JB525	JB523	Mask out CW code of CW found in Dummy
Case ⑤ in	254	QT	JC12	A	List
Dummy List	255	EJ	JC43	JA317	Is it 61? Case ⑤ in Dummy List
	256	MJ	0	JA502	No, go to alarm 13
	257	RJ	JA436	JA433	Case fixed pt. const. no. of subscr. $\begin{matrix}>4\\>1\end{matrix}$, alarm 5 in RJ
	260	MJ	0	JA160	(JA160 OK since coming from JA230, there was SY2-14 not yet changed)
	261	EJ	SZ7	JA263	Is first char. a letter?
	262	MJ	0	JA476	
	263	RJ	JB405	JB401	Store SY → JD and SZ → SY.
	264	TP	JC20	A	Is Level Meter ≠ 0?
	265	MJ	0	JB433	Jump to subroutine to check whether VARY OK
	266	EJ	TS4	JA274	Are we not in subprogram? (TS4 = 0?)
	267	TV	JC64	JB4	Set jump to alarm 21 in JB4 (JA312)
	270	TV	JC73	JB2	Set exit in case it is in Dummy List
	271	RJ	JB5	JB	Get CW from Dummy List or Comb. List
	272	TV	JC63	JB4	Restore address JB4
	273	MJ	0	JB325	Case ⑤ in Comb. List
	274	RJ	TA	TA1	Get file from Comb. List
Not in List	275	MJ	0	JA302	Pseudo oper. not in list; go to put it in
In list	276	TP	JC44	Q	Mask out first digit of CW → A
	277	QT	TA4	A	

	300	EJ	JC45	JA314	Is CW 4---- type (pseudo)?
	301	MJ	0	JB303	Case (1) in list
	302	TP	JC13	A	Is next symbol (?
	303	EJ	JD2	JB45	Go to put pseudo op. in Comb. List
	304	TP	SZ2	TF1	Shortcut: Pseudo op. without following parent.
	305	TP	JC111	TF	
	306	TP	VB	A	Place CW
	307	AT	JC36	TF2	
	310	RJ	TE	TE1	Bring list in Comb. List
	311	MJ	0	JA274	Go again to get list from Comb. List
	312	TV	JC63	JB4	Restore addr. JB4
	313	MJ	0	JA542	Jump to alarm 21
Case CW	314	TP	JC13	A	Case pseudo in list is next symbol (?
4__ __ __	315	EJ	JD2	JB470	
Pseudo Op.	316	MJ	0	JB262	Jump to handle pseudo without following parent.
Case CW	317	TP	JC13	A	Case CW 61 in Dummy List is next
61__ __ __	320	EJ	JD2	JA322	symbol (?
	321	MJ	0	JA524	No, go to alarm 17
	322	TP	JC23	JD32	Set counter of elements in funct. to 3
	323	RJ	JB153	JB145	Save format of given funct.
	324	TV	JC20	JB253	Clear shift count
	325	TP	JC21	JD15	Set index "inside funct." to 1
	326	TP	TS2	JD26	Save XS3 code
	327	TP	TS3	EW2	Place CW in string-out and go to get
	330	MJ	0	JB221	next
	331	EJ	JD17	JA343	Are we not inside pseudo op?
	332	TP	JC21	A	Is Level Meter = 1? (only inside
	333	EJ	JD1	JB526	pseudo?)
	334	EJ	JD16	JA341	Are we inside subscr. variable?
	335	RJ	JB5	JB3	Get file from Comb. List or put it in when needed
	336	MJ	0	JB317	Case (4) in list
	337	EJ	JD16	JA610	Are we not inside subscr. var.? ⟶ alarm
	340	MJ	0	JA363	Go on to check for TS4
	341	RJ	JB5	JB3	Get file from Comb. List
	342	MJ	0	JB360	Case (64) in list
	343	EJ	JD15	JA337	Are we not inside funct.?
	344	EJ	JD16	JA353	Are we not inside subscr. var.?
	345	EJ	TS4	JA351	Is TS4 = 0?
	346	TV	JC65	JB2	Set exit for case it is in Dummy List (63)
	347	RJ	JB5	JB	Get file from dummy or Comb. List
	350	MJ	0	JB352	Case (64) in list (= 2_{64} case)
	351	RJ	JB5	JB3	Get CW from Comb. List
	352	MJ	0	JA350	
	353	TV	JC101	JB222	Set jump-out for funct. (to JB240)
	354	TP	JC20	A	Put zero ⟶ A
	355	EJ	TS4	JA361	Is TS4 = 0?
	356	TV	JC74	JB2	Set exit in case it is in Dummy List
	357	RJ	JB5	JB	Get CW from Dummy List or Comb. List

```
                360   MJ   0       JB333    Case (6) in Comb. List
                361   RJ   JB5     JB3      Get CW from Comb. List
                362   MJ   0       JB311    Case (2) list
                363   EJ   TS4     JA367    Is TS4 = 0 ?
                364   TV   JC66    JB2      Set exit to (63) when in Dummy List
                365   RJ   JB5     JB       Get CW from Dummy List or Comb. List
                366   MJ   0       JB355    Case (64) in list
                367   RJ   JB5     JB3      Get CW from Comb. List
                370   MJ   0       JB355    Case (64) in list
Exit when       371   TP   TS3     Q        Mask out CW code-case (6) in Dummy List
 case (6) in    372   QT   JC12    A
 Dummy List     373   EJ   JC42    JA376    Is it 76 --- ?
                374   EJ   JC11    JA407    Is it 63 --- ?
                375   MJ   0       JA502    Go to alarm 13
                376   TP   JC13    A        Is next symbol ( ?
                377   EJ   JD2     JB162
                400   MJ   0       JA520    Jump to alarm 16
                401   TP   JC20    A        Come from JA55, zero  > A
                402   TJ   JD15    JA157    When open parenthesis found:
                403   TJ   JD16    JA157    Am I really inside something?
                404   TJ   JD17    JA157
                405   MJ   0       JA576    No, jump to alarm 4
                406   0    0       0        Free
                407   TP   JC13    A        Is next symbol ( ?
                410   EJ   JD2     JA514
                411   TP   TS3     TA4      Put CW from dummy result to Comb. result
                412   MJ   0       JB531
Alarm           413   TP   SZ2     JI4
 Entries        414   TU   JI      JB420
                415   RJ   JB424   JB415    Alarm 1
                416   MJ   0       JA160
                417   TP   SZ2     JJ5
                420   TP   SY2     JJ7
                421   TU   JJ      JB420
                422   RJ   JB424   JB415    Alarm 2
                423   RJ   JA423   JA424
                424   MJ   0       JA157
                425   RJ   JA423   JA417
                426   MJ   0       JA43     Alarm 2.5 or 3
                427   TU   JL      JB420
                430   RJ   JB424   JB415
                431   RJ   JA431   JA432    Alarm 4
                432   MJ   0       JA43
                433   TP   JD27    JM4
                434   TU   JM      JB420
                435   RJ   JB424   JB415    Alarm 5
                436   RJ   JA436   JA437
                437   MJ   0       JA43
                440   TP   JD25    IA7
                441   TU   IA      JB420
                442   RJ   JB424   JB415    Alarm 6-1/2 or 18.5
                443   RJ   JA443   JA444
                444   MJ   0       JA107
```

445	TP	TJ1	JK5	
446	TU	JK	JB420	Alarm 6-3/4
447	RJ	JB424	JB415	
450	MJ	0	JA107	
451	RJ	JA431	JA427	Alarm 4.5 or 7
452	MJ	0	JA134	
453	TP	SZ2	JQ4	
454	TP	SZ3	JQ5	
455	TU	JQ	JB420	Alarm 8
456	RJ	JB424	JB415	
457	TV	JC102	JA246	
460	MJ	0	JA136	
461	TP	SZ2	JR4	
462	TP	SZ3	JR5	
463	TU	JR	JB420	Alarm 9
464	RJ	JB424	JB415	
465	TV	JC102	JA246	
466	MJ	0	JA157	
467	RJ	JA436	JA433	
470	TV	JC102	JA246	Alarm 5.5 or 10
471	MJ	0	JA157	
472	TP	SZ2	JT6	
473	TU	JT	JB420	Alarm 11
474	RJ	JB424	JB415	
475	MJ	0	30000	
476	TP	SZ2	JU6	
477	TU	JU	JB420	Alarm 12
500	RJ	JB424	JB415	
501	MJ	0	JB450	
502	TP	SZ2	JV4	
503	TU	JV	JB420	
504	RJ	JB424	JB415	Alarm 13
505	RJ	JA505	JA506	
506	MJ	0	JB445	
507	TP	SZ2	JW4	
510	TU	JW	JB420	
511	RJ	JB424	JB415	Alarm 14
512	RJ	JA512	JA513	
513	MJ	0	JB445	
514	TP	SZ2	JX6	
515	TU	JX	JB420	Alarm 15
516	RJ	JB424	JB415	
517	MJ	0	JA135	
520	TP	SZ2	JY6	
521	TU	JY	JB420	Alarm 16
522	RJ	JB424	JB415	
523	MJ	0	JA163	
524	TP	SZ2	JZ7	
525	TU	JZ	JB420	Alarm 17
526	RJ	JB424	JB415	
527	MJ	0	JA163	
530	TP	SZ2	IA7	Alarm 18
531	MJ	0	JA441	

```
532  TP   SZ2     IB4
533  TP   JD27    IB10     Alarm 19
534  TU   IB      JB420
535  MJ   0       JB415    Alarm 19-1/2 Set RJ exit in JB424
536  TP   SZ2     IC4
537  TP   JD26    IC10     Alarm 20
540  TU   IC      JB420
541  MJ   0       JB415    Alarm 20-1/2 Set RJ exit in JB422
542  TP   SZ2     ID10
543  TU   ID      JB420    Alarm 21
544  RJ   JB424   JB415
545  MJ   0       JB445
546  TP   SZ2     IF6
547  TP   SZ3     IF7
550  TU   IF      JB420    Warning alarm 1
551  RJ   JB424   JB425
552  MJ   0       JA177
553  TP   SZ2     IG6
554  TP   SZ3     IG7
555  TU   IG      JB420    Warning alarm 2
556  RJ   JB424   JB425
557  RJ   JB400   JB371
560  MJ   0       JA177
561  TU   IK      JB420
562  RJ   JB424   JB425    Warning alarm 3
563  MJ   0       JA36
564  TP   JD26    J06
565  TU   J0      JB420    Alarm 5-1/2
566  MJ   0       JB415
567  TU   IH      JB420
570  RJ   JB424   JB415    Alarm 23
571  MJ   0       JA134
572  TP   SZ2     JP2
573  TU   JP      JB420    Alarm 7 or 12-1/2
574  RJ   JB424   JB415
575  MJ   0       JB443
576  RJ   JA431   JA60     Alarm 6 or 4.55
577  MJ   0       JA72
600  TP   SZ2     JS6
601  TU   JS      JB420    Alarm 10 or 17-1/2
602  RJ   JB424   JB415
603  MJ   0       JA163
604  TP   SZ2     IE4
605  TP   JD27    IE10     Alarm 22
606  TU   IE      JB420
607  MJ   0       JB415
610  RJ   JA431   JA60     Alarm 4-1/10
611  MJ   0       JA135
612  TP   JC1     JI4
613  TP   JC123   Q
614  SP   SY2     102      Alarm entr. 1 for neg. sign
615  QS   A       JI4
616  MJ   0       JA414
```

617	TV	JB231	JB223	Alarm 11 1/2
620	TP	SZ2	JT6	
621	TU	JT	JB420	
622	RJ	JB424	JB415	
623	MJ	0	JA135	
	CA	JA624		

```
     IA    JB
 0   RJ    TS      TS1       Get file from Dummy List     Get file from
 1   MJ    0       JB3       not in Dummy List            Comb. or Dummy
                                                          List
 2   MJ    0       30000     RJ-come-back when in Dummy List
 3   RJ    TA      TA1       Get file from Comb. List
 4   MJ    0       JB6       Not in Comb. List:  have CW made
 5   MJ    0       30000     RJ-come-back when in Comb. List
 6   TP    JC15    Q         Set mask in Q                Add symbol to
                                                          Comb. List
 7   TP    JC13    A         }
10   EJ    JD2     JA542     }  Is next symbol (?
11   QT    SZ2     A         }
12   TP    A       A         }  Mask out XS3 code of first digit
13   RP    20005   JB15      }
14   EJ    JC116   JB24      }  Is it I, J, K, L, M?
15   TP    JC20    A         }
16   EJ    JD16    JB22      }  Are we not inside subscr. var.?
17   RJ    JB424   JA532        Print alarm (subscript not starting with
                                 I, J, K, L, M.
20   RA    JD20    JC21         Adv. No. of subscr. (in order to go on
                                 checking format of subscr. var.)
21   MJ    0       JA161        Go and get next symbol (avoid funct. check
                                 because subscr. not interesting anyway)
22   TU    JC6     JB42         Setting for floating point variable.
23   MJ    0       JB35
24   TP    JC20    A         }  Var. start with I, J, K, L, or M
25   EJ    JD16    JB30      }  are we not inside subscr. var.?
26   MJ    0       JB34         We are inside subscr. var.; jump to further
                                 handling.
27   MJ    0       JB34         Free (instr. unused)
30   EJ    JD15    JB34         We are not inside sub.:  are we not inside
                                 funct.?
31   RJ    JB422   JA536        Print alarm
32   TP    JC26    TA4       }  Put "65000" in TA4 and save it as CW in
33   MJ    0       JB220     }  string-out (assuming floating point var. was
                                 meant, in order to go on checking the format
                                 of the function.)
34   TU    JC5     JB42         We are not inside funct.:  make fixed pt. var.
35   TP    JC111   TF           Put 3 in u of TF
36   TP    SZ2     TF1          Put XS3 code in TF1
37   TP    JC20    TF3          Clear TF3
40   RJ    RH      RH1          Check for format
41   RJ    TK      TK1       }  Get CW
42   AT    30000   TF2       }  Add 64 --- or 65 --- (in u 200 or 216)→TF2
43   RJ    TE      TE1          Add new floating point or fixed point vari-
                                 able to Comb. List
44   MJ    0       JB3          Jump to get symbol from Comb. List
45   TP    SZ2     TJ1          Save XS3 repr. of pseudo op.  Pseudo op not
                                 in list.
46   RA    VB      JC36      }  Make up CW
47   TP    A       TJ2       }  (This TJ2 used for updating last TF address,
                                 later added to last bit for no. of elements,
                                 JB77)
```

```
 50   TP   JC110    TJ          Put 2 in u and v of TF0
 51   TV   JC32     JA157       Change exit 19
 52   RJ   JB400    JB376       Place zero string-out word for filling later
 53   TV   EW3      JB117       Save addr. of string-out word for later
                                 placing it
 54   TP   JC21     JD17        Set index "inside pseudo"
 55   TP   JC20     JD24        Set index "pseudo was in list" to zero
                                 since it was not
 56   MJ   0        JA161
 57   RJ   JB57     JA160       Only first time:  get symbol after parent.
 60   TP   JD1      A           Put level Meter --> A
 61   EJ   JC21     JB64        Level = 1 ?
 62   EJ   JC20     JB462       Level = 0 ?
 63   MJ   0        JA160       Level  > 1, go to get next
 64   TP   JC14     A           Level is 1:  put "40 Ø  Ø" --> A
 65   EJ   SZ7      JB104       Is symbol at hand alpha-numeric?
 66   EJ   SZ11     JB70        Is symbol at hand a number?
 67   MJ   0        JA160       For all other symbols go to get next symbol
 70   TP   JC20     A           Has number decimal point?
 71   TJ   SZ6      JB101
 72   TU   JC33     JB75        No, set addr. for format code 2 in JB75
 73   RA   JB47     JC21        Adv. for next TJ word
 74   TV   JB47     JB75        Set next TJ word address in JB75
 75   TP   30000    30000       Place next TJ word
 76   RA   TJ       JC107  }    Adv. counters in TJ region in u
                               } (adv. count of elements) in v
 77   RA   TJ2      JC21   }
100   MJ   0        JA160
101   TU   JC32     JB75        No. with dec. pt; set address for format
                                 code in JB75
102   MJ   0        JB73
103   RJ   TA       TA1         Is this used?
104   TP   JC12     Q           Mask out CW
105   QT   TA4      A
106   EJ   JC25     JB112       Is CW 66?
107   EJ   JC26     JB101       Is CW 65?
110   EJ   JC10     JB72        Is CW 64?
111   MJ   0        JB114       CW 77 left
112   TU   JC37     JB75        Case funct. (66)
113   MJ   0        JB73        Bring TA5 in next TF address.
114   SP   TA5      36     }    Case subscr. var. (77) set up "No. of
115   AT   JC24     TA5    }    subscr.  Ø 4" and store in TA5 (TA5 used
                                 for temp. storage)
116   MJ   0        JB112
117   TP   TF2      30000       Come from JB462/463 (= patch to JB62)
                                 Put CW in saved str. address (set by JB53)
120   RJ   TE       TE1         Finish up bringing pseudo op in Comb. List
121   TV   JC6      JB57        Restore 120 = JB57 for next pseudo not in
                                 list
122   TV   JC6      JA157       Restore exit in "bef. bef. next"
123   TV   JC72     JB47        Restore JB47 to TF2 in v
124   MJ   0        JA160       (Could go to 19 or 20 since exit out of 19
                                 is restored)
```

```
        125  TP   JC13    A       Next symbol ( ?          Subroutine for equal-
        126  EJ   JD2     JB136                            ity jump series
        127  TP   TA4     Q       Mask out CW ind.
        130  QT   JC12    A
        131  EJ   JC25    30000   66?
        132  EJ   JC26    30000   65?
        133  EJ   JC10    30000   64?                      No ( following
        134  EJ   JC12    JA520   77?
        135  MJ   0       JA502   Others?
        136  TP   TA4     Q       Mask out CW ind.
        137  QT   JC12    A
        140  EJ   JC12    30000   77?
        141  EJ   JC25    30000   66?
        142  EJ   JC26    JA514   65?                      With ( following
        143  EJ   JC10    JA514   64?
        144  MJ   0       JA502   Others?
        145  TU   TS17    JB152   Entry for 61 funct.
        146  RA   JB152    JC3
        147  MJ   0       JB151
        150  TU   JC37    JB152   Entry for 66 funct.      Store given
        151  TP   JC20    JD33    Clear JD33               format of
        152  TV   30000   JD33    Store format of funct.   function
        153  MJ   0       30000   Exit for RJ use
I       154  TP   TA3     JD26    Save XS3 code in funct.  XS3 code-storage
                                  Handling 66
        155  TP   JC23    JD32    Put "3" in format of elements (for format
                                  "funct.")
        156  RJ   JB153   JB150   Go to store given funct. format.
        157  TV   JC20    JB253   Clear shift count for next counted element
        160  TP   JC21    JD15    Set "inside funct." to 1
        161  MJ   0       JB220   Go to save CW             Handling 76X
II      162  TP   JC21    JD16    Set inside subscr. to 1.
        163  TP   TS2     JD27    Save XS3 code in subscr. var. XS3 code
        164  TP   JC75    Q       storage
        165  QT   TS3     JD22    Save no. of elements in v of JD22
        166  LQ   JD22    36
        167  MJ   0       JA327   Go to save CW
III     170  TP   JC20    A       Are we not inside subscript var.?  Handl.
        171  EJ   JD16    JB521   subscr.
        172  RA   JD20    JC21    Adv. counter of subscr. by 1
        173  TJ   JC22    JB220   Is 2 > no. of subscripts?  (maximal 1)
        174  RJ   JB424   JA604   Jump to alarm 22 (exit not via possible
        175  MJ   0       JA161   funct. jump out, since inside subscripted
                                  var. not of interest for funct. format)
IV      176  RA   JD20    JC21    Adv. counter of subscr. by 1
        177  TJ   JC7     JB220   Is 5 > no. of subscripts (max. 4) ?
        200  MJ   0       JB174   No, jump to alarm 22 via JB174
V       201  TP   TA3     JD27    Save XS3 code
        202  TP   JC21    JD16    Set "inside subscr." to 1
        203  TP   JC40    Q       Store no. of subscripts
        204  QT   TA5     JD22
```

```
     205   EJ   JC21    JB224      Is No. of subscr. = 1 ?
     206   TJ   JC24    JB232      Is 4 > no. of subscr.?  (case 2 or 3
                                   subscr.)
     207   MJ   0       JB235      Case 4 subscripts
VII  210   TP   JC21    JD17       Set "inside pseudo" to 1
     211   TP   TA3     JD25       Save XS3 code in pseudo op XS3 code
                                   storage
     212   TP   TA4     JD31       Save CW code in pseudo op CW code storage
     213   TP   JC21    JD24       Set index "pseudo was in list"
     214   MJ   0       JB220      Go to save CW
VI   215   TP   JC20    A          Clear A
     216   EJ   JD16    JB220      Are we not inside subscr. var.?
     217   MJ   0       JB176      Jump to IV
Save 220   TP   TA4     EW2  ┐     Save 1 CW
1 CW 221   RJ   EW      EW1  │     RJ for exit with funct. (jumps to JB240)
     222   RJ   JB222   JB223 │     and used by pseudo JB470 as exit, v rest
     223   MJ   0       JA163┘     by JA617
Save 224   RJ   JB223   JB220 ┐
2    225   RA   JB220   JC    │
CW's 226   RJ   JB223   JB220 │     Save 2 CW's
     227   TU   JC41    JB220 │     Restore JB220
     230   TV   JC41    JB223 │     Restore JB223
     231   MJ   0       JA163┘
Save 232   RJ   JB223   JB220 ┐
3    233   RA   JB220   JC    │     Save 3 CW's
CW's 234   MJ   0       JB224┘
Save 235   RJ   JB223   JB220 ┐     Save 4 CW's          Form funct. format for
4    236   RA   JB220   JC    │                          comp. with given one.
CW's 237   MJ   0       JB232┘
     240   TU   EW3     JB242 ┐     Come from JB222 (when funct. jump out
                               │     was set)
     241   TP   JC12    Q     │     Mask out CW code of CW just placed in
     242   QT   30000   A     ┘     string-out
     243   EJ   JC13    JB256      77
     244   EJ   JC42    JB256      76
     245   EJ   JC77    JB260      67
     246   EJ   JC26    JB260      65
     247   EJ   JC1     JB260      63
     250   MJ   0       JB223      Others (ignore and go on)
     251   RA   JB253   JC23       Come from JB256 or JB260.  Adv. shift count
     252   TP   30000   Q          Put 1 or 4 in Q
     253   LQ   Q       30000      Shift (set to 0 when funct. symbol found)
     254   RA   JD32    Q          Add code no. to "counter" of element in funct.
     255   MJ   0       JB223      Go back to exit of "save CW"
     256   TU   JC72    JB252 ┐    Cases 77, 76, (4)
     257   MJ   0       JB251┘
     260   TU   JC73    JB252 ┐    Cases 67, 65, 63 (1) (when 63 and inside
     261   MJ   0       JB251 │    sub. we skip the funct. jump out)
     262   TP   JC67    Q     ┐    Come from JA316 (entr. set from v in
     263   QT   TA4     A     ┘    JC70)  Mask out last 2 digits for pseudo
                                   (to check whether with operands or not)
     264   ZJ   JA530   JB220      When last 2 bits = 0, go to put CW in
                                   string-out
```

```
                265   TP   JC12    Q       } Come from JB307 or JB331
                266   QT   TA4     A       } Mask out first 2 bits of symbol at hand
                                             (after it came out of Comb. List)
                267   EJ   JC12    JB274     77
                270   EJ   JC25    JB274     66
                271   EJ   JC26    JB274     65
                272   EJ   JC10    JB274     64
                273   MJ   0       30000
                274   TP   JC71    Q       }
                275   QT   TA5     A       } Is symbol defined by special equation?
                276   ZJ   JB273   JA507   }
                277   RJ   JB223   JB220     Come from JB141, save CW
                300   RJ   JA154   JA617     Give alarm concerning skipping inside
                                             parent.
                301   TV   JB302   JA154     Restore exit of alarm
                302   MJ   0       JA157     Go to EXTRA. Checks for special equations
(1)             303   RJ   JB467   JB464     66, 65, 64  }
                304   MJ   0       JB305     Free        } Not followed by (
                305   TV   JC52    JB140     77          }
                306   TV   JC46    JB141     66          } Followed by (
                307   RJ   JB273   JB265   Check whether symbol has special equat.
                310   MJ   0       JB125
(2)             311   TV   JC55    JB131     66      }
                312   TV   JC54    JB132     65      } Not followed by (
                313   TV   JC50    JB133     64      }
                314   TV   JC52    JB140     77      }
                315   TV   JC55    JB141     66      } Followed by (
                316   MJ   0       JB125
(4)             317   TV   JC54    JB131     66      }
                320   TV   JC54    JB132     65      } Not followed by (
                321   TV   JC53    JB133     64      }
                322   TV   JC52    JB140     77      }
                323   TV   JC60    JB141     66      } Followed by (
                324   MJ   0       JB125
(5)             325   RJ   JB467   JB464     66, 65, 64  }
                326   MJ   0       JB327     Free        } Not followed by (
                327   TV   JC52    JB140     77          }
                330   TV   JC46    JB141     66          } Followed by (
                331   RJ   JB273   JB265   Check whether symbol has special equat.
                332   MJ   0       JB125
(6)             333   TV   JC55    JB131     66      }
                334   TV   JC54    JB132     65      } Not followed by (
                335   TV   JC50    JB133     64      }
                336   TV   JC52    JB140     77      }
                337   TV   JC55    JB141     66      } Followed by (
                340   MJ   0       JB125             Entries for EJ series
Case 63         341   TP   TS3     Q       }
                342   MJ   0       JB344   } Mask out (dummy) CW code
Case 64         343   TP   TA4     Q       }
                344   QT   JC12    A       } Mask out CW code
                345   EJ   30000   JB347     63--- or 64--- (has been set before
                                             entering here)
```

```
346  MJ   0       JA502     Jump to alarm 13
347  TP   JC13    A         Is next symbol ( ?
350  EJ   JD2     JA514     Jump to alarm 15
351  MJ   0       30000
352  TU   JC46    JB345
353  TV   JC50    JB351     2 64 or 6 64
354  MJ   0       JB343
355  TU   JC46    JB345
356  TV   JC51    JB351     3 64 or 7 64
357  MJ   0       JB343
360  TU   JC46    JB345
361  TV   JC53    JB351     4 64
362  MJ   0       JB343
363  TU   JC47    JB345     6 63
364  TV   JC50    JB351
365  MJ   0       JB503     Jump first to place TS3 ⟹ TA4, then
366  TU   JC47    JB345     go to further handling.
367  TV   JC51    JB351     7 63
370  MJ   0       JB503                          Place zero string-
371  TU   EW3     JB373     Come from JA45         out word
372  TP   JC20    A         Was last string-out word a zero?
373  EJ   30000   JB400
374  RP   10017   JB376
375  TP   JC20    JD15      Clear counters and indices and storages
376  TP   JC20    EW2
377  RJ   EW      EW1       Place zero string-out word
400  MJ   0       30000     Only used in RJ when found, or AND and
                            Level Meter = 0.
401  RP   30013   JB403
402  TP   SY2     JD2
403  RP   30013   JB405     Save SY, SZ
404  TP   SZ2     SY2
405  MJ   0       30000
406  RP   30013   JB410
407  TP   JD2     SY2       Restore SY, SZ
410  MJ   0       30000
411  TP   JC20    JD1       Clear level meter
412  TP   JC20    JD21      Clear counter of elements inside pseudo
413  TP   JC20    JD17      Clear inside pseudo
414  MJ   0       30000
415  MJ   0       JB427     Alarm Print
416  TP   WL1     II4
417  RJ   UP2     UP
420  TP   30000   UP3
421  RJ   UP2     UP        Exit both when funct. jump out should
422  RJ   JB422   JB423     not be restored, used by Alarm 20
423  TV   JC100   JB222     Restore funct. jump out (in case it was
                            set)
424  MJ   0       30000     Exit both (restoring of funct. jump out
                            does not hurt warning alarms, since they
                            occur never for alpha-numeric symbols)
425  TP   II1     UP3
426  MJ   0       JB416     Entrance alarm print warning
```

```
427   RJ   UZ       UZ1
430   TP   II       UP3      Entrance alarm print error
431   MJ   0        JB416
432   0    0        0        Free completely     VARY check
433   TJ   JD1      JA331    Is Level Meter = 0 ?
434   TP   JC20     JB441    Set RP 2(n)  JB443
435   TU   VL       JB441
436   RS   JB441    JC56
437   AT   JC103    JB441
440   TP   SZ2      A        Put XS3 symbol at hand --> A
441   0    0        0        Is it equal one in given list?  (Concern-
                             ing VARY
442   EJ   VL1      JA572    Yes, go to alarm 12-1/2 = 7
443   TP   JC20     A        All OK, restore A to 0
444   MJ   0        JA266    Go back to routine
445   TP   JD2      A        Alarm exit when SY stored otherwise
446   EJ   JC13     JA135    Is next symbol ( ?  If yes, go to skip
447   MJ   0        JA163    contents of parent.
450   TP   SY2      A        Alarm exit when SY not stored otherwise
451   EJ   JC13     JA136    Is next symbol ( ?  If yes, go to skip
                             contents of parent.
452   MJ   0        JA157
453   TU   EW3      JB455    Place address of last  string-out word in
                             JB455
454   TP   JC20     A        Clear A
455   EJ   30000    JB460    Was last string-out word = zero?  If yes,
                             skip placing another
456   TP   JC20     EW2      Place  zero-string-out word
457   RJ   EW       EW1
460   TP   JC30     EW2      Put XS3  code of Δ . in EW2
461   MJ   0        30000    Jump back to routine
462   RP   30030    JB117    Place TJ region in TF region after
463   TP   TJ       TF       pseudo was built up
464   TV   JC106    JB131    Patch, come from JB303 or JB325
465   TV   JC54     JB132    Setting 66, 65, 64 with no (
466   TV   JC54     JB133
467   MJ   0        30000
470   RJ   JB222    JB210    Come from JA315 jump to do normal han-
                             dling of pseudo
471   TV   JC32     JA157    Set pseudo jump-out in JA157
472   RP   30025    JB474    Store format of all possible CW + 24
                             operands
473   TP   TA4      JD35     Save TA4
474   RA   JB462    JC22     Set jump over "bring list in Comb. List"
475   TP   JC115    Q        Put mask 77700 --> Q
476   QT   TA4      TJ2      Set first 3 digits of CW in TJ2 (set up
                             for format)
477   RJ   JB124    JB56     Go back to routine
500   TV   JC6      JB124    Rest JB124 (RJ exit)
501   RS   JB462    JC22     Reset jump to bring list in Comb. List
502   MJ   0        JA160    Go to get next symbol
```

```
503   TP   TS3      TA4     }  Set dummy CW's aside for storing in
504   MJ   0        JB341   }   string-out(when found in Dummy List)
505   TP   JC114    JB510      Come from JA104; preset JB510
506   TP   JC67     Q       }  Set index for no. of operands
507   QT   TJ2      JD34    }  (mask Ø Ø 00077)
510   RS   JD35     TJ2        Preset by JB505
511   ZJ   JA440    JB512      Are the values equal?          } Compare set
512   RA   JB510    JC107      Adv. to next val.              } up pseudo op
513   IJ   JD34     JB510      Are all val. handled?          } in TJ2-25
514   MJ   0        JA107      Test OK, go on in routine      } with the
                                                              stored given
                                                              values
515   TP   JC21     JD30       Set extra Level Meter = 1      }
516   RJ   JB410    JB406      Restore SY, SZ                 } Special en-
517   RS   JD1      JC21       Restore Level Meter            } trance EXTRA
520   MJ   0        JA137      Jump to extra                  }
521   RJ   JB422    JA536      Jump to alarm 20 for subscr. in funct.
                                and don't restore funct. jump out
522   MJ   0        JB32       Go to set dummy 65000 in TA4
523   TV   JC63     JB4     }  Patch for JA253; restore exit in JB4
524   TP   TS3      Q       }  Prepare masking of CW code
525   MJ   0        30000   }  Go back
526   RA   JD21     JC21    }  Patch for JA333 (to restore accumulator
527   TP   JC21     A       }  after RA) Adv. no. of elements in pseudo
                             }  Put 1    accumulator
530   MJ   0        JA334   }  Go back to routine
531   TP   JC20     A       }  Patch for JB351
532   EJ   JD16     JB220   }  Clear A
                             }  Are we not in subscr. var.? Then save
                             }  CW, case 63 for single valued floating
                             }  point var.
533   MJ   0        JB172   }  We are in subscr. var.: handle case 63
                                for subscr.
      CA   JB534
```

```
      IA    JC
 0    0     1       0
 1    0     WL3     WL3
 2    24    50277   77777        XS3 code AND
 3    0     2       0
 4    0     3       0
 5    0     JC10    JA251
 6    0     JC26    JA160
 7    0     0       5
10    0     0       64000
11    0     0       63000
12    0     0       77000
13    17    77777   77777        XS3 (
14    40    0       0
15    77    0       0
16    34    0       0
17    50    0       0
20    0     0       0            Zero
21    0     0       1            One
22    0     0       2
23    0     0       3
24    0     0       4
25    0     0       66000
26    0     0       65000
27    43    77777   77777        XS3 )
30    01    22777   77777        XS3 Δ .
31    0     0       25           Used by JA113
32    0     JC21    JB57
33    0     JC22    JB135
34    21    77777   77777        XS3 ,
35    23    77777   77777        XS3 ;
36    0     0       00100
37    0     TA5     0
40    0     0       77777
41    0     TA4     JA163
42    0     0       76000
43    0     0       61000
44    0     0       70000
45    0     0       40000
46    0     JC10    JB154        I
47    0     JC11    JB162        II
50    0     0       JB170        III
51    0     0       JB176        IV
52    0     0       JB201        V
53    0     0       JB215        VI
54    0     0       JB220        Save 1 CW
55    0     0       JA502
56    0     VL1     0            For checking VARY range
57    0     0       JB210
60    0     0       JB277
61    0     0       JA124
62    0     WL      WL
63    0     0       JB6
```

```
 64     0     0       JA312
 65     0     0       JB363
 66     0     0       JB366
 67     0     0       00077
 70     0     0       JB262
 71     04    0       0
 72     0     JC24    TJ2
 73     0     JC21    JA253
 74     0     0       JA371
 75     0     0       00700
 76     0     70000   0
 77     0     0       67000
100     0     0       JB223
101     0     0       JB240
102     0     0       JA247
103     RP    20000   JB443
104     02    77777   77777      Lower case
105     0     77777   77777      Upper case
106     0     0       JA600
107     0     1       1
110     0     2       2
111     0     3       3
112     0     JC20    TE1
113     0     JC21    0
114     RS    JD35    TJ2
115     0     0       77700      Mask
116     34    0       0          }
117     44    0       0          }  Comparisons
120     45    0       0          }  for I,J,K,L,M
121     46    0       0          }
122     47    0       0          }
123     0     77777   77777      For neg. sign alarm 1
        CA    JC124
```

Alarm 1

```
      IA    JI
 0    0     JI1     0
 1    40    JI2     10
 2    65    73472   55146
 3    01    77777   77777        Symbol [ ] illegal
 4    0     0       0   SZ2           in compute sentence
 5    01    34464   63032
 6    24    46013   45001
 7    26    51475   26766
10    30    01653   05066
11    30    50263   02201
      CA    JI12
```

Alarm 2

```
      IA    JJ
 0    0     JJ1     0
 1    40    JJ2     15
 2    24    27442   42630        Adjacent symbols [ ] [ ]
                                 meaningless in compute sentence
 3    50    66016   57347
 4    25    51466   50177
 5    0     0       0            SZ2
 6    01    77777   77777
 7    0     0       0            SY2
10    01    47302   45034
11    50    32463   06565
12    01    34500   12651
13    47    52676   63001
14    01    01010   10101
15    65    30506   63050
16    26    30220   17777
      CA    JJ17
```

Alarm 3 = 6-3/4

	IA	JK		
0	0	JK1	0	
1	40	JK2	10	
2	52	65306	72751	Pseudo operation [ ] has too many elements.
3	01	51523	05424	
4	66	34515	00177	
5	0	0	0	TJ 1
6	01	33246	50166	
7	51	51014	72450	
10	73	01304	63047	
11	30	50666	52277	
	CA	JK12		

Alarm 4

	IA	JL		
0	0	JL1	0	
1	40	JL2	6	Parentheses not properly paired
2	52	24543	05066	
3	33	30653	06501	
4	50	51660	15254	
5	51	52305	44673	
6	01	52243	45430	
7	27	22017	77777	
	CA	JL10		

Alarm 5

```
        IA      JM
 0      0       JM1     0
 1      40      JM2     11          Variable[ ] has incorrect
 2      70      24543   42425       number of subscripts
 3      46      30017   77777
 4      0       0       0           JD27
 5      01      33246   50134
 6      50      26515   45430
 7      26      66015   06747
10      25      30540   15131
11      01      65672   56526
12      54      34526   66522
        CA      JM13
```

Alarm 6 = 5-1/2

```
        IA      J0
 0      0       J01     0
 1      40      J02     15          Arguments of function [ ]
 2      24      54326   74730       do not agree with previous usage
 3      50      66650   15131
 4      01      31675   02666
 5      34      51500   17777
 6      0       0       0           JD26
 7      01      27510   15051
10      66      01243   25430
11      30      01713   46633
12      01      01010   10101
13      01      01010   10101
14      52      54307   03451
15      67      65016   76524
16      32      30220   17777
        CA      J017
```

Alarm 7 = 12-1/2

	IA	JP	
0	0	JP1	0
1	40	JP2	20
2	0	0	0
3	01	26245	05051
4	66	01253	00126
5	51	47526	76630
6	27	01333	05430
7	21	01653	45026
10	30	01663	33001
11	65	30506	63050
12	26	30013	46501
13	01	01010	10101
14	34	50016	63330
15	01	54245	03230
16	01	51310	12401
17	70	24547	30165
20	66	24663	04730
21	50	66227	77777
	CA	JP22	

SZ2 [ ] Cannot be computed here, since the sentence is in the range of a vary sentence.

Alarm 8

	IA	JQ		
0	0	JQ1	0	
1	40	JQ2	12	
2	26	51506	56624	Constant [ ] is followed by open parenthesis.
3	50	66017	77777	
4	0	0	0	SZ2
5	0	0	0	SZ3
6	01	34650	13151	
7	46	46517	13027	
10	01	25730	15152	
11	30	50015	22454	
12	30	50663	33065	
13	34	65227	77777	
	CA	JQ14		

Alarm 9

	IA	JR		
0	0	JR1	0	
1	40	JR2	15	Constant [ ] appears as variable to be computed
2	26	51506	56624	
3	50	66017	77777	
4	0	0	0	SZ2
5	0	0	0	SZ3
6	01	24525	23024	
7	54	65012	46501	
10	70	24543	42425	
11	46	30016	65101	
12	25	30010	10101	
13	01	01010	10101	
14	01	01010	10101	
15	26	51475	26766	
16	30	27220	17777	
	CA	JR17		

Alarm 10 = 17-1/2

```
      IA    JS
0     0     JS1     0
1     40    JS2     10
2     24    54326   74730
3     50    66650   15131
4     01    31675   02666
5     34    51500   17777
6     0     0       0
7     01    27510   15051
10    66    01245   25230
11    24    54227   77777
      CA    JS12
```

Arguments of function [ ] do not appear.

Alarm 11

```
      IA    JT
0     0     JT1     0
1     40    JT2     10
2     24    54326   74730
3     50    66650   15131
4     01    31675   02666
5     34    51500   17777
6     0     0       0
7     01    24543   00165
10    67    52305   43146
11    67    51676   52201
      CA    JT12
```

Arguments of function [ ] are superfluous

Alarm 12

	IA	JU		
0	0	JU1	0	
1	40	JU2	14	Alphanumeric symbol [ ]
2	24	46523	32450	has digit as first character
3	67	47305	43426	
4	01	65734	72551	
5	46	01777	77777	
6	0	0	0	SZ2
7	01	33246	50127	
10	34	32346	60124	
11	65	01313	45465	
12	66	01010	10101	
13	01	01010	10101	
14	26	33245	42426	
15	66	30542	27777	
	CA	JU16		

Alarm 13

	IA	JV		
0	0	JV1	0	
1	40	JV2	6	Symbol[ ] incorrectly used
2	65	73472	55146	
3	01	77777	77777	
4	0	0	0	SZ2
5	01	34502	65154	
6	54	30266	64673	
7	01	67653	02722	
	CA	JV10		

Alarm 14

```
        IA    JW
 0      0     JW1      0
 1      40    JW2      11          Variable [  ] is not defined
 2      70    24543    42425       by an equation
 3      46    30017    77777
 4      0     0        0           SZ2
 5      01    34650    15051
 6      66    01273    03134
 7      50    30270    12573
10      01    24500    13053
11      67    24663    45150
12      22    77777    77777
        CA    JW13
```

Alarm 15

```
        IA    JX
 0      0     JX1      0
 1      40    JX2      14          Single valued variable [  ]
 2      65    34503    24630       is followed by an open parenthesis
 3      01    70244    66730
 4      27    01702    45434
 5      24    25463    00177
 6      0     0        0           SZ2
 7      01    34650    13151
10      46    46517    13027
11      01    25730    12450
12      01    51523    05001
13      01    01010    10101
14      52    24543    05066
15      33    30653    46522
        CA    JX16
```

Alarm 16

	IA	JY		
0	0	JY1	0	
1	40	JY2	10	
2	65	67256	52654	Subscripts of variable [ ]
3	34	52666	50151	do not appear
4	31	01702	45434	
5	24	25463	00177	
6	0	0	0	SZ2
7	01	27510	15051	
10	66	01245	25230	
11	24	54227	77777	
	CA	JY12		

Alarm 17

	IA	JZ		
0	0	JZ1	0	
1	40	JZ2	11	Arguments of dummy
2	24	54326	74730	function [ ] do not
3	50	66650	15131	appear
4	01	27674	74773	
5	01	31675	02666	
6	34	51500	17777	
7	0	0	0	SZ2
10	01	27510	15051	
11	66	01245	25230	
12	24	54227	77777	
	CA	JZ13		

Alarm 18

```
        IA    IA
 0      0     IA1      0
 1      40    IA2      16          Format of pseudo operation[ ]
 2      31    51544    72466       does not agree with previous usage
 3      01    51310    15265
 4      30    67275    10151
 5      52    30542    46634
 6      51    50017    77777
 7      0     0        0           SZ2 for 18; JD25 for 6-1/2
10      01    27513    06501
11      50    51660    12432
12      54    30300    10101
13      01    01010    10101
14      01    71346    63301
15      52    54307    03451
16      67    65016    76524
17      32    30220    17777
        CA    IA20
```

Alarm 19

```
        IA    IB
 0      0     IB1      0
 1      40    IB2      16          Subscript[ ]of variable[ ] is
 2      65    67256    52654       not properly written
 3      34    52660    17777
 4      0     0        0           SZ2
 5      01    51310    17024
 6      54    34242    54630
 7      01    77777    77777
10      0     0        0           JD27
11      01    34650    15051
12      66    01525    45152
13      30    54467    30101
14      01    01010    10101
15      01    01010    10101
16      71    54346    66630
17      50    22017    77777
        CA    IB20
```

Alarm 20

	IA	IC		
0	0	IC1	0	
1	40	IC2	16	
2	24	54326	74730	Argument[ ] of function[ ]
3	50	66017	77777	appears as subscript
4	0	0	0	SZ2
5	01	51310	13167	
6	50	26663	45150	
7	01	77777	77777	
10	0	0	0	JD26
11	01	24525	23024	
12	54	65012	46501	
13	01	01010	10101	
14	01	01010	10101	
15	01	01010	10101	
16	65	67256	52654	
17	34	52662	27777	
	CA	IC20		

Alarm 21

	IA	ID		
0	0	ID1	0	
1	40	ID2	17	Function or subscripted
2	31	67502	66634	variable[ ]has not been
3	51	50015	15401	previously mentioned
4	65	67256	52654	
5	34	52663	02701	
6	70	24543	42425	
7	46	30017	77777	
10	0	0	0	SZ2
11	01	33246	50150	
12	51	66010	10101	
13	01	01010	10101	
14	25	30305	00152	
15	54	30703	45167	
16	65	46730	14730	
17	50	66345	15030	
20	27	22017	77777	
	CA	ID21		

Alarm 22

```
        IA    IE
 0      0     IE1      0
 1      40    IE2      20          Subscript[  ] of variable[  ]
 2      65    67256    52654       exceeds allowed number
 3      34    52660    17777       of subscripts
 4      0     0        0           SZ2
 5      01    51310    17024
 6      54    34242    54630
 7      01    77777    77777
10      0     0        0           JD27
11      01    30722    63030
12      27    65012    44646
13      51    71302    70101
14      01    01010    10101
15      01    01010    10101
16      50    67472    53054
17      01    51310    16567
20      25    65265    43452
21      66    65220    17777
        CA    IE22
```

Alarm 23

	IA	IH		
0	0	IH1	0	
1	40	IH2	12	
2	31	34546	56601	
3	65	73472	55146	First symbol is wrong.
4	01	34650	17154	Sentence is not checked
5	51	50322	20165	further
6	30	50663	05026	
7	30	01505	16601	
10	31	67546	63330	
11	54	01010	10101	
12	01	01010	12633	
13	30	26453	02722	
	CA	IH14		

Warning Alarm 1

	IA	IF		
0	0	IF1	0	
1	40	IF2	14	
2	26	51474	72401	Comma assumed between
3	24	65656	74730	[ ] and preceding symbol
4	27	01253	06671	[ ]
5	30	30500	17777	
6	0	0	0	SZ2
7	0	0	0	SZ3
10	01	24502	70152	
11	54	30263	02734	
12	50	32010	10101	
13	01	01010	10101	
14	01	01010	10165	
15	73	47255	14622	
	CA	IF16		

Warning Alarm 2

```
        IA    IG
 0      0     IG1      0
 1      40    IG2      12
 2      41    24502    74101        "And" assumed between
 3      24    65656    74730         [ ] and preceding symbol
 4      27    01253    06671         [ ]
 5      30    30500    17777
 6      0     0        0            SZ2
 7      0     0        0            SZ3
10      01    24502    70152
11      54    30263    02734
12      50    32016    57347
13      25    51462    20177
        CA    IG14
```

Warning Alarm 3

```
        IA    IK1
 0      0     IK1      0
 1      40    IK2      10
 2      41    24502    74101        "And" inside parentheses
 3      34    50653    42730        replaced by comma
 4      01    52245    43050
 5      66    33306    53065
 6      01    54305    24624
 7      26    30270    12573
10      01    01010    10101
11      26    51474    72422
        CA    IK12
```

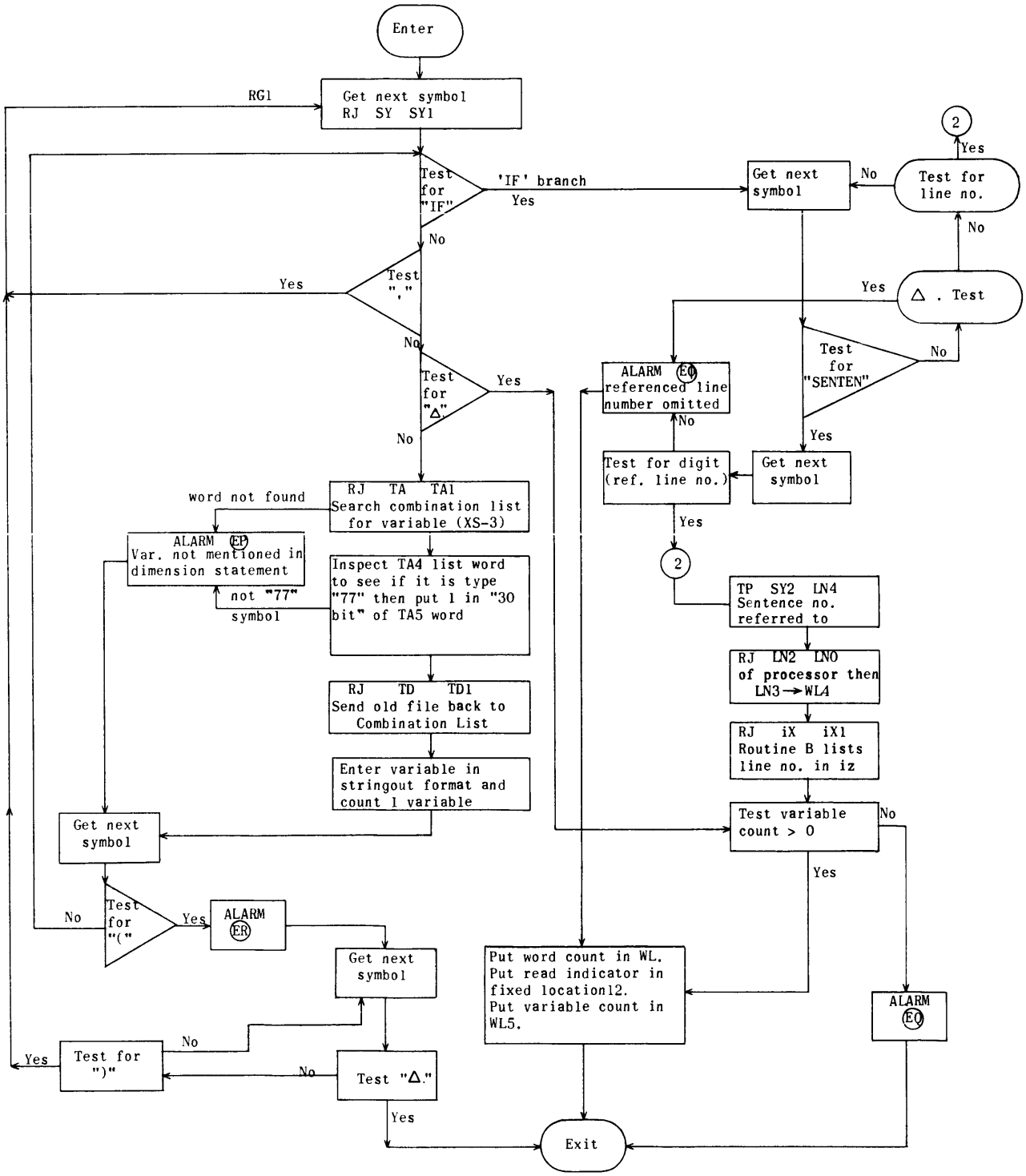
Alarm Heading

```
        IA    II
 0      0     II2      6            (real alarm)
 1      0     II2      10           (warning)
 2      65    30506    63050
 3      26    30010    17777        Sentence Δ Δ [ ] ΔΔ Δ      (compute)
 4      0     0        0             ΔΔ    warning, ΔΔ
 5      01    01011    72651
 6      47    52676    63043
 7      01    01777    77777
10      71    24545    03450
11      32    21010    17777
        CA    II12
```

Temporaries Compute String-out

JD	0	Word index
	1	Level Meter
	2-14	Storage for SY
	15	Index "inside function"
	16	Index "inside subscr. var."
	17	Index inside pseudo oper.
	20	Counter of subscripts
	21	Counter of elements in pseudo operation
	22	Stored number of subscripts
	23	Free (?)
	24	Index "pseudo was in list"
	25	XS3 represent. of pseudo operation
	26	XS3 represent. of function
	27	XS3 represent. of subscripted variable
	30	Extra level meter
	31	CW pseudo operat. (for no. of elements)
	32	Built up format of function
	33	Stored format of function
	34	Index for pseudo operation
	35-61	Stored CW + operands of pseudo operation
TJ	0-27	Space to build up pseudo operation for Comb. List

Flow Chart for Read String-Out



```
      RE    RG4400
      RE    E04474
      RE    EP4520
      RE    EQ4543
      RE    ER4564
```

String-Out subroutine regions are needed for assembly of this tape.

```
      IA    RG
 0    MJ    0       CT          Exit to String-Out Control
 1    RJ    SY      SY1         Get next symbol
 2    TP    EP10    WL4         Clearing 5th line of output
 3    EJ    RG57    RG33        Is symbol an IF?
 4    EJ    RG60    RG1         Comma test
 5    EJ    RG61    RG46        Space period test.
 6    RJ    TA      TA1         Get file from Combination List.
 7    MJ    0       EP1         Error return from TA
10    MJ    0       RG20
11    EJ    RG61    E01         Is symbol a space period?
12    TP    SY11    Q           } Is symbol a constant?
13    QJ    RG42    RG33        }
14    RJ    SY      SY1
15    EJ    RG61    RG          Δ . test                          } Deleting loop
16    EJ    RG63    RG1         Closed parenthesis test           }
17    MJ    0       RG14
20    TP    TA4     Q           Call word to q
21    QT    RG64    A           Mask (0     0     XX000)
22    TJ    RG65    EP1         Test for "77 ---" Type
23    TP    RG66    Q           } Put 1 in 30 bit.
24    QS    RG66    TA5         }
25    RJ    TD      TD1         Send file back to Combination List
26    TP    TA3     [WL6]
27    RA    RG26    RG67
30    RJ    SY      SY1
31    EJ    RG62    ER1         Open parenthesis detection gives alarm.
32    MJ    0       RG2
33    RJ    SY      SY1         Begin IF rout.
34    EJ    RG70    RG36        Is symbol "SENTEN"?
35    MJ    0       RG11
36    RJ    SY      SY1
37    TP    SY11    Q           } Is symbol a constant?
40    QJ    RG42    RG41        }
41    MJ    0       E01
42    TP    SY2     LN4         }
43    RJ    LN2     LN          } Line number put in standard form and put
                                  in output.
44    TP    LN3     WL4         }
```

```
45      RJ    IX      IX1         Line number put in referenced-line number
                                  list.
46      TP    RG26    A           TP TA3   [WL6]   to A
47      SS    RG71    0           - TP     TA3     WL6
50      ZJ    RG51    EQ1
51      TP    A       WL5         Variable count
52      AT    RG72    WL
53      TP    RG73    Q  }
54      QS    Q       12 }        Put read indicator into 12
55      RJ    SS      SS1         Exit procedure
56      MJ    0       RG
57      34    31777   77777       IF
60      21    77777   77777       ,
61      01    22777   77777       Δ   .
62      17    77777   77777       (
63      43    77777   77777       )
64      0     0       77000
65      0     0       76777
66      1     0       0           1 in 30 bit
67      0     0       1
70      65    30506   63050
71      TP    TA3     WL6
72      0     0       6
73      0     2       0
        CA    RG74
```

Read Error Print-Out Subroutines

```
        IA    EP
0       MJ    0       RG30
1       TP    SY2     EP10        Set-ups variable for print-out
2       RJ    WA      WA1         Print-out:  Sentence_____(Read)
3       TP    EP22    UP3 }
4       RJ    UP2     UP  }       Print-Out shown below.
5       MJ    0       EP
6       70    24543   42425       V  A  R  I  A  B
7       46    30017   77777       L  E  Δ
10      0     0       0
11      01    50516   60147       Δ  N  0  T  Δ  M
12      30    50663   45150       E  N  T  I  0  N
13      30    27013   45001       E  D  Δ  I  N  Δ
14      66    33300   11727       T  H  E  Δ  (  D
15      34    47305   06534       I  M  E  N  S  I
16      51    50430   10101       0  N  )  Δ  Δ  Δ
17      01    01010   10101       Δ  Δ  Δ  Δ  Δ  Δ
20      01    65662   46630       Δ  S  T  A  T  E
21      47    30506   62277       M  E  N  T  .  Δ
22      40    EP6     14
        CA    EP23

        IA    EQ
0       MJ    0       RG55
```

```
 1      RJ    WA       WA1
 2      TP    EQ20     UP3
 3      RJ    UP2      UP
 4      MJ    0        EQ
 5      50    51016    56725      N  O  Δ  S  U  B
 6      65    26543    45266      S  C  R  I  P  T
 7      30    27017    02454      E  D  Δ  V  A  R
10      34    24254    63001      I  A  B  L  E  Δ
11      47    30506    63451      M  E  N  T  I  O
12      50    30270    13450      N  E  D  Δ  I  N
13      01    66333    00117      Δ  T  H  E  Δ  (
14      54    30242    74301      R  E  A  D  )  Δ
15      01    01010    10165      Δ  Δ  Δ  Δ  Δ  S
16      66    24663    04730      T  A  T  E  M  E
17      50    66227    77777      N  T  .
20      40    EQ5      13
        CA    EQ21

        IA    ER
 0      MJ    0        RG14
 1      RJ    WA       WA1
 2      TP    ER27     UP3
 3      RJ    UP2      UP
 4      MJ    0        ER
 5      65    67256    52654      S  U  B  S  C  R
 6      34    52663    02701      I  P  T  E  D  Δ
 7      70    24543    42425      V  A  R  I  A  B
10      46    30650    12454      L  E  S  Δ  A  R
11      30    01505    16601      E  Δ  N  O  T  Δ
12      24    46465    17130      A  L  L  O  W  E
13      27    01665    10133      D  Δ  T  O  Δ  H
14      24    70300    10101      A  V  E  Δ  Δ  Δ
15      01    01010    10152      Δ  Δ  Δ  Δ  Δ  P
16      24    54305    06633      A  R  E  N  T  H
17      30    66342    62446      E  T  I  C  A  L
20      01    30725    25430      Δ  E  X  P  R  E
21      65    65345    15065      S  S  I  O  N  S
22      01    34500    16633      Δ  I  N  Δ  T  H
23      30    01175    43024      E  Δ  (  R  E  A
24      27    43016    56624      D  )  Δ  S  T  A
25      66    30473    05066      T  E  M  E  N  T
26      22    77777    77777      .
27      40    ER5      22
        CA    ER30

        IA    E0
 0      MJ    0        RG51
 1      RJ    WA       WA1
 2      TP    E023     UP3
 3      RJ    UP2      UP
 4      MJ    0        E0
 5      67    65345    03201      U  S  I  N  G  Δ
```

6	66	33300	17151	T H E Δ W O
7	54	27013	43101	R D Δ I F Δ
10	34	47524	63430	I M P L I E
11	65	01543	03130	S Δ R E F E
12	54	30502	63450	R E N C I N
13	32	01240	14634	G Δ A Δ L I
14	50	30015	06747	N E Δ N U M
15	25	30542	10125	B E R , Δ B
16	67	66016	63330	U T Δ T H E
17	01	50674	72530	Δ N U M B E
20	54	01712	46501	R Δ W A S Δ
21	51	47346	66630	O M I T T E
22	27	22777	77777	D .
23	40	E05	16	
	CA	E024		

Type String-Out.

The type string-out saves a call word and an excess-three representation for every symbol of a type sentence even for parentheses, commas and the like. Space period in excess-three form represents the end of the string-out.

The routine checks for type faults, wrongly chosen symbols and incorrect number of subscripts of a subscripted variable. Furthermore it checks for all sorts of errors induced by the combination of several symbols.

When the first symbol of the sentence following the word "type" is space period, the routine prints an alarm and exits without further checking of the sentence. This is the only case where the routine skips the rest of the sentence. In all other cases the sentence is handled up to the very end (if the number of alarms does not exceed the limit) and only contents of parentheses may be skipped, where a preceding alarm made the checking of the symbols inside the parentheses meaningless.

The alarm print-outs are as follows:

1) Symbol [ ] illegal in type sentence.
2) First symbol space point. Sentence not further checked.
3) Adjacent symbols [ ] [ ] meaningless in type sentence.
4) Parentheses not correctly placed.
5) Subscripted variable [ ] has incorrect number of subscripts.
6) Alphanumeric symbol [ ] has digit as first character.
7) [ ] appears as function or subscripted variable, but is not previously mentioned in problem.
8) Floating point variable [ ] used as subscript.
9) Pseudo operation symbol [ ] illegal in type sentence.
10) Subscripts of subscripted variable [ ] are missing.
11) Library routine symbol [ ] illegal in type sentence.
12) Floating point variable [ ] used as function symbol or as subscripted variable.
13) Subscript [ ] appears as function or subscripted variable.
14) Subscript [ ] used as argument of a function.
15) Function symbol [ ] appears as subscript.

16) Constant [ ] appears as function or subscripted variable.

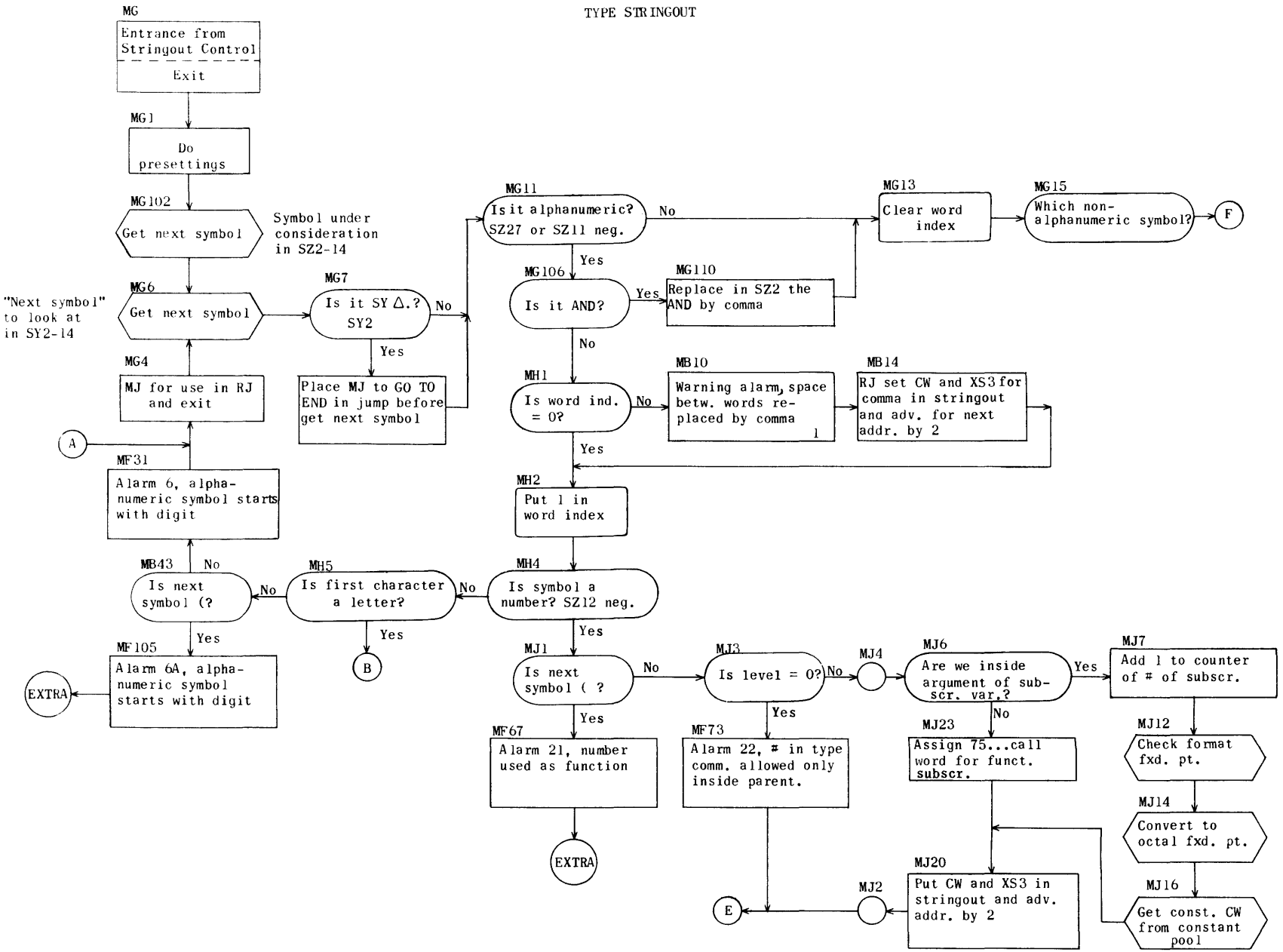
17) Constant [ ] appears as variable to be typed.

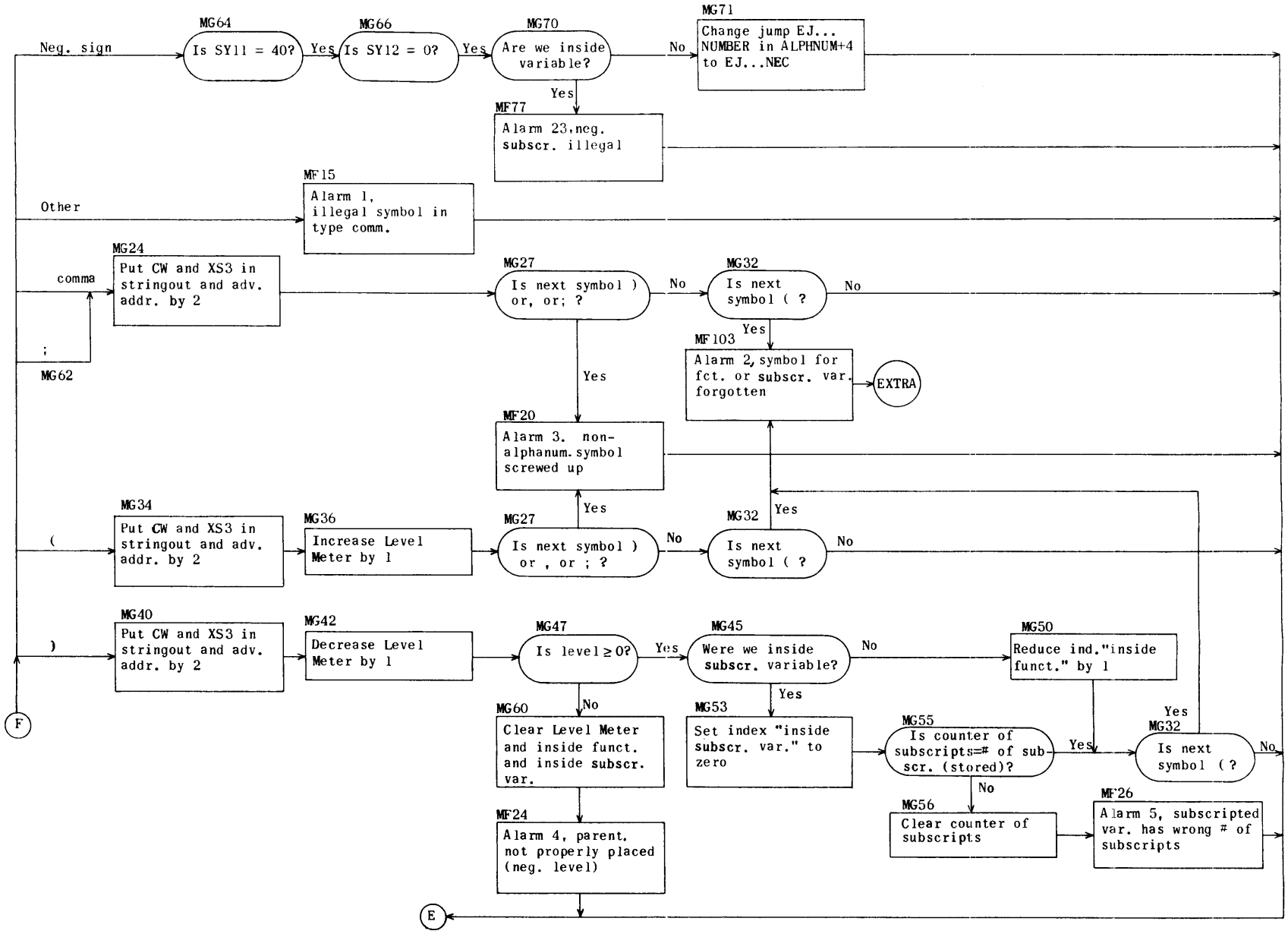
18) Negative subscript [ ] illegal.

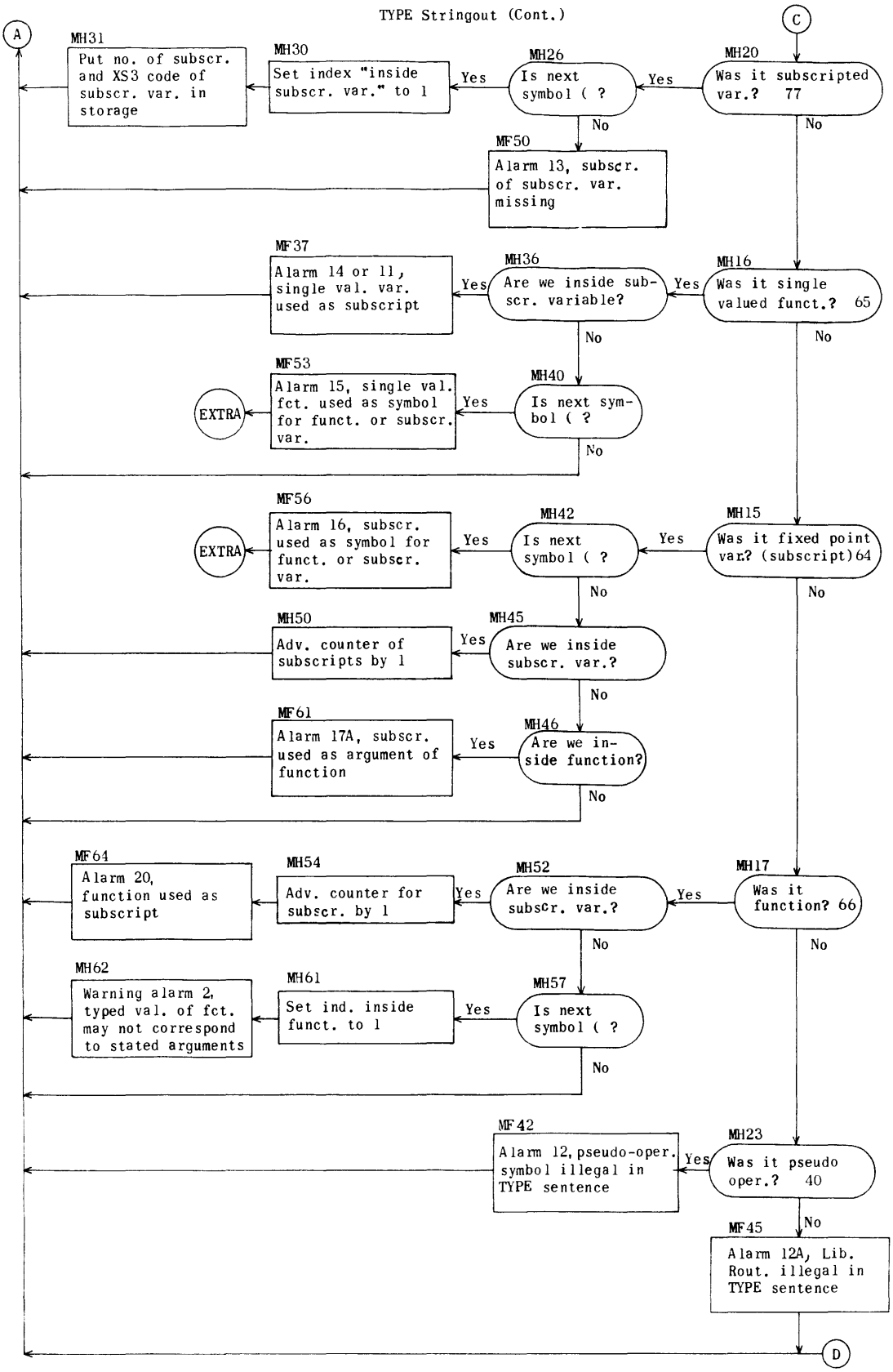
Warnings.

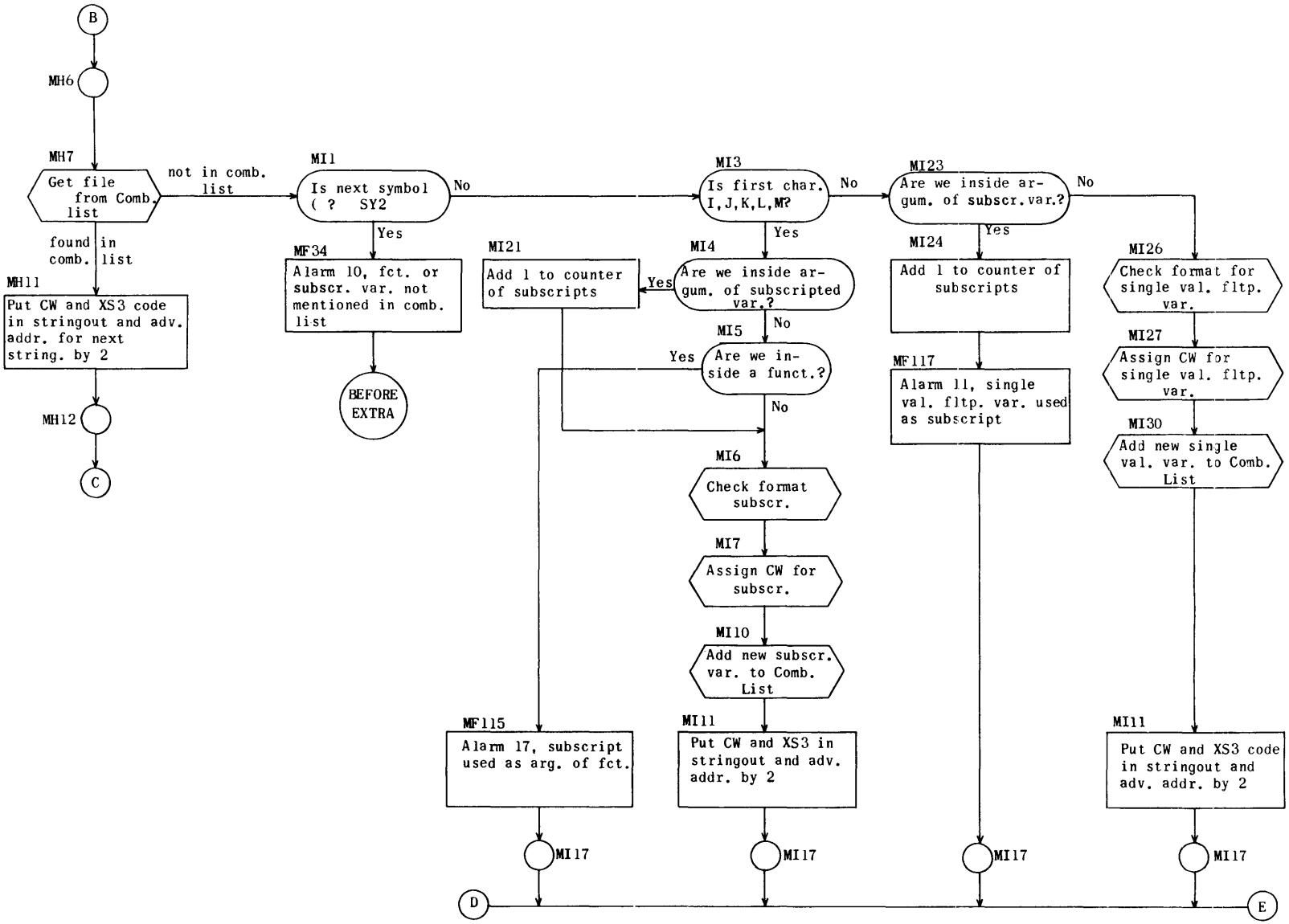
1) Comma assumed preceding symbol [ ].

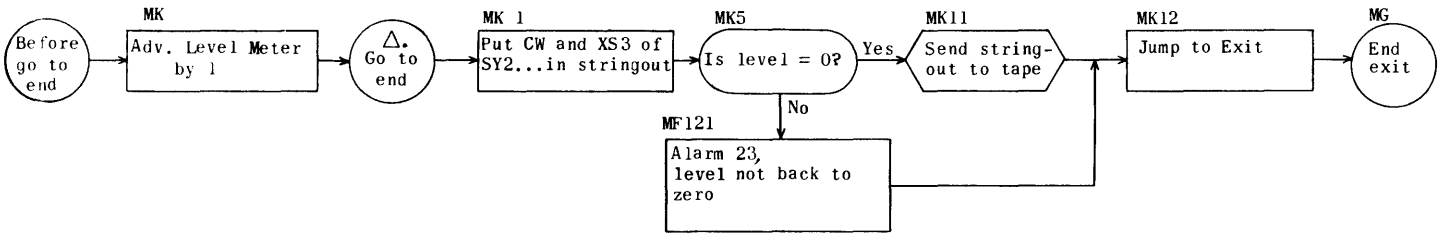
2) Typed value of function [ ] may not correspond to stated arguments.

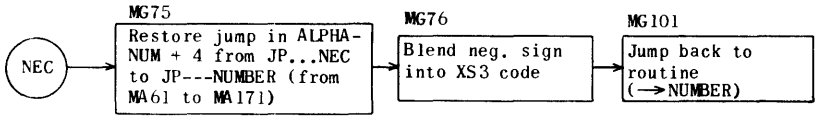

MG
Entrance from Stringout Control
Exit
MG1
Do presettings
MG102
Get next symbol
Symbol under consideration in SZ2-14
"Next symbol" to look at in SY2-14
MG6
Get next symbol
MG7
Is it SY Δ.? SY2
No
Yes
Place MJ to GO TO END in jump before get next symbol
MG4
MJ for use in RJ and exit
A
MF31
Alarm 6, alpha-numeric symbol starts with digit
MB43
Is next symbol (?
No
Yes
MF105
Alarm 6A, alpha-numeric symbol starts with digit
EXTRA
MH5
Is first character a letter?
No
Yes
B
MH4
Is symbol a number? SZ12 neg.
No
Yes
MH2
Put 1 in word index
MH1
Is word ind. = 0?
No
MB10
Warning alarm, space betw. words re-placed by comma
1
MB14
RJ set CW and XS3 for comma in stringout and adv. for next addr. by 2
Yes
MG11
Is it alphanumeric? SZ27 or SZ11 neg.
No
Yes
MG13
Clear word index
MG15
Which non-alphanumeric symbol?
F
MG106
Is it AND?
Yes
MG110
Replace in SZ2 the AND by comma
No
MJ1
Is next symbol (?
No
Yes
MF67
Alarm 21, number used as function
EXTRA
MJ3
Is level = 0?
No
Yes
MF73
Alarm 22, # in type comm. allowed only inside parent.
E
MJ4
MJ6
Are we inside argument of sub-scr. var.?
Yes
No
MJ23
Assign 75...call word for funct. subscr.
MJ20
Put CW and XS3 in stringout and adv. addr. by 2
MJ2
MJ7
Add 1 to counter of # of subscr.
MJ12
Check format fxd. pt.
MJ14
Convert to octal fxd. pt.
MJ16
Get const. CW from constant pool

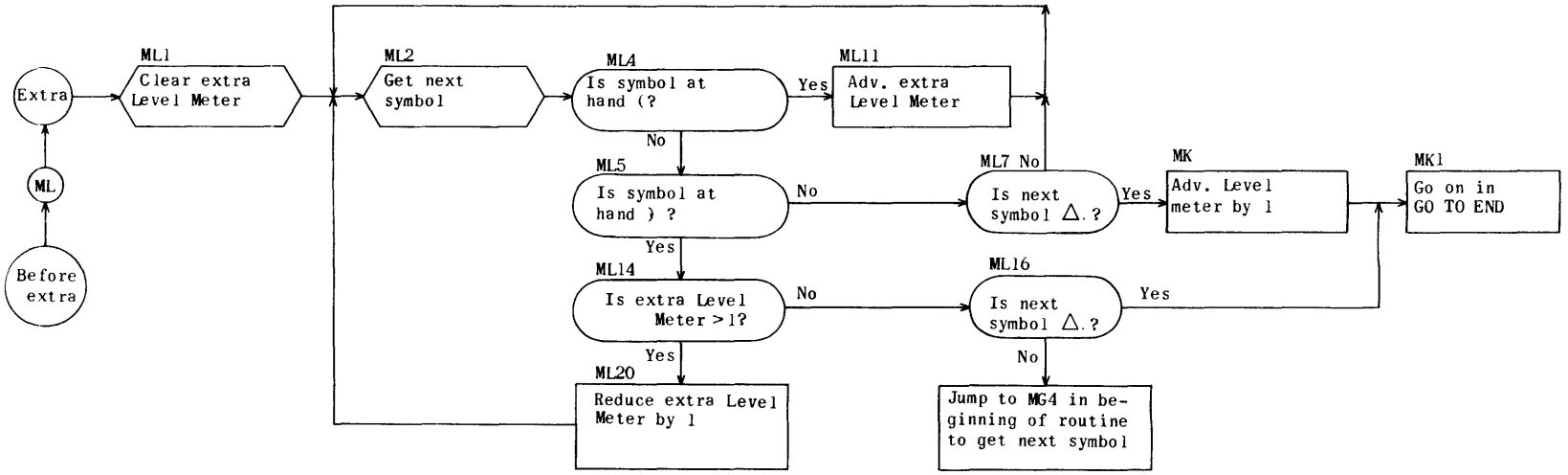

MG64
Is SY11 = 40?
Yes
MG66
Is SY12 = 0?
Yes
MG70
Are we inside variable?
No
MG71
Change jump EJ... NUMBER in ALPHNUM+4 to EJ...NEC
Yes
MF77
Alarm 23, neg. subscr. illegal
Neg. sign
Other
MF15
Alarm 1, illegal symbol in type comm.
comma
MG24
Put CW and XS3 in stringout and adv. addr. by 2
;
MG62
MG27
Is next symbol) or, or; ?
No
MG32
Is next symbol (?
No
Yes
MF103
Alarm 2, symbol for fct. or subscr. var. forgotten
EXTRA
Yes
MF20
Alarm 3. non-alphanum. symbol screwed up
(
MG34
Put CW and XS3 in stringout and adv. addr. by 2
MG36
Increase Level Meter by 1
MG27
Is next symbol) or , or ; ?
No
MG32
Is next symbol (?
No
Yes
Yes
)
MG40
Put CW and XS3 in stringout and adv. addr. by 2
MG42
Decrease Level Meter by 1
MG47
Is level ≥ 0?
Yes
MG45
Were we inside subscr. variable?
No
MG50
Reduce ind. "inside funct." by 1
Yes
No
MG60
Clear Level Meter and inside funct. and inside subscr. var.
MG53
Set index "inside subscr. var." to zero
MG55
Is counter of subscripts=# of subscr. (stored)?
Yes
MG32
Is next symbol (?
Yes
No
No
MG56
Clear counter of subscripts
MF26
Alarm 5, subscripted var. has wrong # of subscripts
MF24
Alarm 4, parent. not properly placed (neg. level)
F
E


TYPE Stringout (Cont.)
A
C
MH31
Put no. of subscr. and XS3 code of subscr. var. in storage
MH30
Set index "inside subscr. var." to 1
Yes
MH26
Is next symbol (?
Yes
MH20
Was it subscripted var.? 77
No
MF50
Alarm 13, subscr. of subscr. var. missing
MF37
Alarm 14 or 11, single val. var. used as subscript
Yes
MH36
Are we inside subscr. variable?
Yes
MH16
Was it single valued funct.? 65
No
MF53
Alarm 15, single val. fct. used as symbol for funct. or subscr. var.
EXTRA
Yes
MH40
Is next symbol (?
No
MF56
Alarm 16, subscr. used as symbol for funct. or subscr. var.
EXTRA
Yes
MH42
Is next symbol (?
Yes
MH15
Was it fixed point var.? (subscript)64
No
MH50
Adv. counter of subscripts by 1
Yes
MH45
Are we inside subscr. var.?
No
MF61
Alarm 17A, subscr. used as argument of function
Yes
MH46
Are we inside function?
No
MF64
Alarm 20, function used as subscript
MH54
Adv. counter for subscr. by 1
Yes
MH52
Are we inside subscr. var.?
Yes
MH17
Was it function? 66
No
MH62
Warning alarm 2, typed val. of fct. may not correspond to stated arguments
MH61
Set ind. inside funct. to 1
Yes
MH57
Is next symbol (?
No
MF42
Alarm 12, pseudo-oper. symbol illegal in TYPE sentence
Yes
MH23
Was it pseudo oper.? 40
No
MF45
Alarm 12A, Lib. Rout. illegal in TYPE sentence
D


B
MH6
MH7
Get file from Comb. list
not in comb. list
found comb. list
in
MH11
Put CW and XS3 code in stringout and adv. addr. for next string. by 2
MH12
C
MI1
Is next symbol (? SY2
No
Yes
MF34
Alarm 10, fct. or subscr. var. not mentioned in comb. list
BEFORE EXTRA
MI3
Is first char. I,J,K,L,M?
No
Yes
MI4
Are we inside argum. of subscripted var.?
Yes
No
MI5
Are we inside a funct.?
Yes
No
MI21
Add 1 to counter of subscripts
MI6
Check format subscr.
MI7
Assign CW for subscr.
MI10
Add new subscr. var. to Comb. List
MF115
Alarm 17, subscript used as arg. of fct.
MI11
Put CW and XS3 in stringout and adv. addr. by 2
MI17
MI23
Are we inside argum. of subscr.var.?
No
Yes
MI24
Add 1 to counter of subscripts
MF117
Alarm 11, single val. fltp. var. used as subscript
MI26
Check format for single val. fltp. var.
MI27
Assign CW for single val. fltp. var.
MI30
Add new single val. var. to Comb. List
MI11
Put CW and XS3 code in stringout and adv. addr. by 2
MI17
D
E


Before go to end
MK
Adv. Level Meter by 1
Δ. Go to end
MK 1
Put CW and XS3 of SY2...in stringout
MK5
Is level = 0?
Yes
MK11
Send string-out to tape
MK12
Jump to Exit
MG
End exit
No
MF121
Alarm 23, level not back to zero


NEC
MG75
Restore jump in ALPHA-NUM + 4 from JP...NEC to JP---NUMBER (from MA61 to MA171)
MG76
Blend neg. sign into XS3 code
MG101
Jump back to routine (→NUMBER)


Extra
ML
Before extra
ML1
Clear extra Level Meter
ML2
Get next symbol
ML4
Is symbol at hand (?
Yes
ML11
Adv. extra Level Meter
No
ML5
Is symbol at hand) ?
No
ML7 No
Is next symbol Δ.?
Yes
MK
Adv. Level meter by 1
MK1
Go on in GO TO END
Yes
ML14
Is extra Level Meter > 1?
No
ML16
Is next symbol Δ.?
Yes
Yes
ML20
Reduce extra Level Meter by 1
No
Jump to MG4 in beginning of routine to get next symbol

Type String-Out Regions

RE	MG4400	Program main part
RE	MH4512	
RE	MI4577	
RE	MJ4631	
RE	MK4656	
RE	ML4671	
RE	MB4713	Subroutines
RE	MC4773	Constants
RE	MF5034	Alarm entries
RE	YY5162	
RE	YB5175	
RE	YC5215	
RE	YD5227	
RE	YE5244	
RE	YF5263	
RE	YG5273	
RE	YH5312	
RE	YI5330	
RE	YJ5355	
RE	YK5370	
RE	YL5406	
RE	YM5424	
RE	YN5442	
RE	YO5464	
RE	YP5504	
RE	YQ5517	
RE	YR5531	
RE	YS5552	
RE	YT5571	
RE	YZ5603	
RE	ME5614	
RE	MM6140	
RE	MN6240	

String-Out Subroutine regions are also needed to assemble this tape.

		IA	MG		
	0	MJ	0	CT	Exit
	1	TP	MC36	EW3	Set EW3
	2	RP	10025	MG102	Clear temporaries and go to check
	3	TP	MC12	ME	for Δ . in start
	4	MJ	0	MG6	Come from MG104 "before get next symbol"
	5	0	0	0	Free (first RJ SY SY1 see in patch, MG102)
	6	RJ	SY	SY1	Get next symbol for SY (later → SZ)
	7	EJ	MC1	MB	Is next symbol Δ . (the end point)?
	10	TP	MC13	A	Put "40 Ø Ø" → A
	11	EJ	SZ7	MG105	Is first digit a letter?
	12	EJ	SZ11	MG105	Is first digit a number or decimal point?
Non-	13	TP	MC12	ME16	Clear word index
Alpha-	14	TP	SZ2	A	Put XS3 code at hand → A
Numeric	15	EJ	MC3	MG24	Is it , ?
	16	EJ	MC5	MG34	Is it (?
	17	EJ	MC4	MG40	Is it) ?
	20	EJ	MC2	MG62	Is it ; ?
	21	EJ	MC34	MG64	Is it neg. sign upper case?
	22	EJ	MC35	MG64	Is it neg. sign lower case?
	23	MJ	0	MF15	Illegal non-alpha-numeric symbol in type command
Comma	24	TP	MC26	ME13	Fill string-out for comma
	25	RJ	MB37	MB34	
	26	TP	SY2	A	Put next symbol → A
	27	EJ	MC2	MF20	Is it ; ?
	30	EJ	MC3	MF20	Is it , ?
	31	EJ	MC4	MF20	Is it) ?
	32	EJ	MC5	MF103	Is it (?
	33	MJ	0	MG4	Go to "before get next"
First Par	34	TP	MC	ME13	Fill string-out for (
	35	RJ	MB37	MB34	
	36	RA	ME17	MC14	Adv. Level Meter by 1
	37	MJ	0	MG26	Do the same error tests as comma
Last Par	40	TP	MC30	ME13	Fill string-out for)
	41	RJ	MB37	MB34	
	42	RS	ME17	MC14	Reduce Level Meter by 1
	43	SJ	MG60	MG44	Is Level Meter ≥ 0?
	44	TP	MC12	A	Yes, put zero → A
	45	TJ	ME20	MG53	Are we inside subscr. var.?
	46	TJ	ME21	MG50	No: are we inside function? Only alternative with level > 0
	47	MJ	0	MG4	No: jump to get next symbol (really not needed; should never occur)
	50	RS	ME21	MC14	Subtr. 1 from "inside funct" index
	51	TP	SY2	A	Jump to one more error test
	52	MJ	0	MG32	
	53	TP	MC12	ME20	Clear index "inside subscr. var."
	54	RS	ME22	ME23	Subtr. "counted # of subscr." - "stored # of subscr."

	55	ZJ	MG56	MG51	Is counted no. of var. of subscr. var. OK?	
	56	TP	MC12	ME22	Clear counter of subscripts	
	57	MJ	0	MF26	Jump to alarm 5 (no. of subscripts not correct)	
	60	RP	10003	MF24	Come from MG43	
	61	TP	MC12	ME17	Level Meter neg; clear Level Meter and inside subscr. variable and inside funct. and go to alarm 4	
Semicolon	62	TP	MC3	SZ2	Case; sneak XS3 code of comma in	
	63	MJ	0	MG24	SZ2 and jump to making string-out for comma	
Negative	64	TP	MC13	A	Put "40 Ø Ø"⟶A	
	65	TJ	SY11	MF15	Is next symbol not starting with decimal pt. or digit?	
	66	EJ	SY12	MF15	Contains next symbol a letter?	
	67	TP	MC14	A	Are we inside subscr. var.?	
	70	EJ	ME20	MF77		
	71	TV	MC6	MH4	Set jump for neg. no.	
	72	MJ	0	MG4	Jump to "before get next"	
	73	RA	ME22	MC14	Adv. no. of subscr.	No more used?
	74	MJ	0	MF77	Jump to alarm 1A	
NEC	75	TV	MC10	MH4	Restore jump to NUMBER	
	76	SP	MC30	25	Blend neg. sign (lower case) in XS3 code	
	77	SA	SZ2	36		
	100	LT	0	SZ2		
	101	MJ	0	MJ	Jump to NUMBER handling.	patch for case that sentence starts with Δ
	102	RJ	SY	SY1	Come from MG2, get first symbol	
	103	EJ	MC1	MF123	Is first symbol Δ . ? Yes, go to alarm.	
	104	MJ	0	MG4	No, go on in routine	
	105	TP	SZ2	A	Come from MG11 or MG12	patch for case "AND" instead of comma
	106	EJ	MC33	MG110	Is symbol "AND"?	
	107	MJ	0	MH	No, go on in rout.	
	110	TP	MC3	SZ2	Yes, sneak comma in SZ2	
	111	MJ	0	MG13	Go on in routine	
		CA	MG112			

		IA	MH		
Alpha-	0	TP	MC12	A	Put zero ⟶ A
Numeric	1	TJ	ME16	MB10	Is word ind. > zero? When yes, warning alarm 1
	2	TP	MC14	ME16	Put 1 in word index
	3	TP	MC12	A	Clear A
	4	EJ	SZ12	MJ	Has symbol no letter?
	5	EJ	SZ7	MB43	With letter: is first character not a letter?
	6	RJ	MB21	MB15	Write SZ2-14 ⟶ SY2-14 and save SY2 in ME
	7	RJ	TA	TA1	Is symbol at hand in Combination List?
	10	MJ	0	MI	No
In Comb. List	11	RJ	MB33	MB30	Yes, put CW and XS3 code in string-out
	12	RJ	MB24	MB22	Write SY2-14 back in its place
	13	TP	MC22	Q	Mask for call word symbol ⟶ Q
	14	QT	TA4	ME13	Call word symbol ⟶ ME13 and A
	15	EJ	MC17	MH42	Was symbol in Comb. List subscript? 64
	16	EJ	MC20	MH35	Was symbol in Comb. List single val. variable? 65
	17	EJ	MC21	MH52	Was symbol in Comb. List function? 66
	20	EJ	MC22	MH25	Was symbol in Comb. List subscripted variable? 77
	21	TP	MC31	Q	Was symbol in Comb. List pseudo op? 40
	22	QT	TA4	ME13	
	23	EJ	MC37	MF42	
Subvar 2	24	MJ	0	MF45	Was symbol in Comb. List Library Routine?
	25	TP	MC5	A	Is next symbol (?
	26	EJ	ME	MH30	
	27	MJ	0	MF50	No, go to alarm
	30	TP	MC14	ME20	Put 1 in "inside subscr. var."
	31	TP	MC23	Q	Put no. of subscr. in storage
	32	QT	TA5	ME23	
	33	TP	TA3	ME24	Put XS3 code of subscr. var. in storage
	34	MJ	0	MG4	Go to "bef. get next"
Sinvalf	35	TP	MC12	A	Are we inside subscr. var.?
2	36	TJ	ME20	MH63	
	37	TP	MC5	A	No: is next symbol (?
	40	EJ	ME	MF53	
	41	MJ	0	MG4	No: go to "bef. get next"
Sub-	42	TP	MC5	A	Is next symbol (?
script 2	43	EJ	ME	MF56	
	44	TP	MC12	A	Are we inside subscr. var.?
	45	TJ	ME20	MH50	
	46	TJ	ME21	MF61	No: are we inside funct.?
	47	MJ	0	MG4	No: go to get next symbol

Sub-	50	RA	ME22	MC14	Adv. counter of subscr. by 1
script 2A	51	MJ	0	MG4	Go to "before get next"
Func-	52	TP	MC12	A	Are we not inside subscr. var.?
tion 2	53	EJ	ME20	MH56	
	54	RA	ME22	MC14	We are inside: adv. counter of subscr. by 1
	55	MJ	0	MF64	Go to alarm 20
	56	TP	MC5	A	We are not inside: is next symbol (?
	57	EJ	ME	MH61	
	60	MJ	0	MG4	Go to "before get next"
	61	TP	MC14	ME21	Set "inside funct" to 1
	62	MJ	0	MF12	Go to warning alarm 2
	63	RA	ME22	MC14	Adv. counter of subscr.
	64	MJ	0	MF37	Go to alarm 11 or 14
		CA	MH65		
Not in		IA	MI		
List	0	TP	MC5	A	Is next symbol (? yes: go to alarm 10
	1	EJ	ME	MF34	
	2	TP	MC12	A	No: is first letter not I,J,K,L,M?
	3	EJ	SZ10	MI23	
	4	TJ	ME20	MI21	First letter I,J,K,L,M: Are we inside subscr. var.?
	5	TJ	ME21	MF115	Not inside sub. var.: are we inside function? Yes, go to alarm 17
	6	RJ	RH	RH1	No, check format
	7	RJ	TK	TK1	Get CW no. for 64, 65, 66 type
	10	AT	MC17	ME13	Add 64 to it and store formed CW in ME13
	11	RJ	MB37	MB34	Fill string-out, etc.
	12	TP	MC27	TF	Ø 3 Ø
	13	TP	SZ2	TF1	Add new XS3 code
	14	TP	ME13	TF2	file to CW
	15	TP	MC12	TF3	Combination zero
	16	RJ	TE	TE1	List
	17	RJ	MB24	MB22	Write SY2-14 back in its place
	20	MJ	0	MG4	Go to "before get next"
	21	RA	ME22	MC14	Adv. counter for subscripts
Sinvalf 1	22	MJ	0	MI6	Go to set subscr. in string-out and Comb. List
	23	EJ	ME20	MI26	Are we not inside subscr. var.?
	24	RA	ME22	MC14	We are inside: adv. counter for subscr.
	25	MJ	0	MF117	Jump to alarm
	26	RJ	RH	RH1	Check format
	27	RJ	TK	TK1	Get CW No. for 64, 65, 66
	30	AT	MC20	ME13	Add 65--- and store formed CW in ME13
	31	MJ	0	MI11	Go to set single val. funct. in String-out and Comb. List
		CA	MI32		

		IA	MJ		
Number	0	TP	MC5	A	Is next symbol (? Yes, go to alarm 21
	1	EJ	SY2	MF67	
	2	TP	MC12	A	Is level = 0? Yes, go to alarm 22
	3	EJ	ME17	MF73	
	4	RJ	MB21	MB15	No: store SZ2-14 ⟶ SY2-14 WORK-WORK12
	5	TP	MC12	A	Are we not inside subscr. var.?
	6	EJ	ME20	MJ23	
	7	RA	ME22	MC14	We are inside sub. var.: add 1 to counter of subscripts
	10	MJ	0	MJ12	Jump to over-next instr.
	11	0	0	0	Free
	12	RJ	RD	RD1	Check format of fixed point constant
	13	TP	SZ2	RS4	Convert XS3 decimal no. to fixed pt. octal no.
	14	RJ	RS2	RS	
	15	TP	TS3	A	Assign constant CW
	16	RJ	GW	GW1	
	17	TP	Q	ME13	Put CW and XS3 code in string-out and adv.
	20	RJ	MB37	MB34	
	21	RJ	MB24	MB22	Write SY2-14 back
	22	MJ	0	MG4	Go to "before get next"
Stated	23	TP	MC16	ME13	Assign 75--- CW for funct. subscr.
	24	MJ	0	MJ20	
		CA	MJ25		

```
                IA      MK
Bef.Go      0   RA      ME17    MC14        Adv. Level Meter                    )
To End      1   TP      MC32    ME13    )                                       )
            2   TP      SY2     SZ2     }   Write Δ . in string-out             )
Go To       3   RJ      MB37    MB34    )                                       )
End         4   TP      MC12    A       }   Is Level Meter = 0?                 )
            5   TJ      ME17    MF121   }   No, go to alarm 4                   )
            6   RS      EW3     MC40    )                                       ) END
            7   AT      MC14    A       )                                       )
           10   TV      A       WL      }   Send string-out to tape            )
           11   RJ      WT      WT1     )                                       )
           12   MJ      0       MG      }   Exit                               )
                CA      MK13    )                                               )

                IA      ML
Bef.        0   RJ      MB24    MB22        Restore SY2-14
Extra       1   TP      MC12    ME15        Put zero in "extra Level Meter"
Extra       2   RJ      SY      SY1         Get next symbol
            3   TP      SZ2     A       )
            4   EJ      MC5     ML11    }   Is symbol at hand ( ?
            5   EJ      MC4     ML13        Is symbol at hand ) ?
            6   TP      SY2     A       )
            7   EJ      MC1     MK      }   Is next symbol Δ . ?
           10   MJ      0       ML2
NOCH       11   RA      ME15    MC14        Adv. extra Level Meter
           12   MJ      0       ML2
           13   TP      MC14    A       )
MEHR       14   TJ      ME15    ML20    }   Is extra Level Meter > 1 ?
           15   TP      SY2     A       )
           16   EJ      MC1     MK1     }   Is next symbol Δ . ?
           17   MJ      0       MG4         Go to "bef. get next"
Most       20   RS      ME15    MC14        Reduce extra Level Meter by 1
           21   MJ      0       ML2
                CA      ML22
```

	IA	MB		
0	TV	ML16	MG4	Put "JUMP to EXIT" in place "bef. get next"
1	MJ	0	MG10	Back to routine
2	TP	30000	EW2	CW
3	TP	30000	ME14	XS3 symbol
4	RJ	EW	EW1	Place CW and XS3 code in string-out and adv. addresses concerned
5	TP	ME14	EW2	
6	RJ	EW	EW1	
7	MJ	0	30000	
10	RJ	MB54	MF	Warning 1
11	TU	MC25	MB2	
12	TU	MG30	MB3	Go to print warning alarm, and put string-out for comma in list and go back to routine.
13	RJ	MB7	MB2	
14	MJ	0	MH2	
15	RP	30013	MB17	
16	TP	SY2	ME	Write "get next symbol" SZ2-14 ⟶ SY2-14 and save SY2-14 in ME0-12
17	RP	30013	MB21	
20	TP	SZ2	SY2	
21	MJ	0	30000	
22	RP	30013	MB24	
23	TP	ME	SY2	Write SY2-14 back
24	MJ	0	30000	
25	TP	MC5	A	
26	ST	SY2	A	Is next symbol (? when no, go to get next symbol. When yes, come back to RJ
27	ZJ	MG4	ML1	
30	TU	MC10	MB2	
31	TU	MC11	MB3	Place CW and XS3 code in string-out when found in Comb. List
32	RJ	MB7	MB2	
33	MJ	0	30000	
34	TU	MC6	MB2	Place CW and XS3 code in string-out when CW placed in ME13
35	TU	MC7	MB3	
36	RJ	MB7	MB2	
37	MJ	0	30000	
40	0	0	0	
41	0	0	0	Free
42	0	0	0	
43	TP	MC5	A	
44	EJ	SY2	MF105	Alpha-numeric symbol with first char. digit (SY not yet moved, OK)
45	MJ	0	MF31	

```
46    RJ      UZ      UZ1
47    TP      YZ1     UP3
50    TP      WL1     YZ4
51    RJ      UP2     UP
52    TP      30000   UP3
53    RJ      UP2     UP
54    RJ      MB54    MB55
55    MJ      0       MG4
56    TP      YZ      UP3
57    MJ      0       MB50
      CA      MB60
```

Error print

Error count

heading printed

text printed

Exit for RJ used

Normal exit

Entry for warnings

Alarm Entries and Exits

	IA	MF		
0	TP	SZ2	YY10	Warning alarm 1
1	TP	SZ3	YY11	
2	TU	YY	MB52	
3	MJ	0	MB56	
4	0	0	0	Free
5	0	0	0	
6	0	0	0	
7	0	0	0	
10	0	0	0	
11	0	0	0	
12	TP	SZ2	YB6	Warning alarm 2
13	TU	YB	MB52	
14	MJ	0	MB56	
15	TP	SZ2	YC4	Alarm 1
16	TU	YC	MB52	
17	MJ	0	MB46	
20	TP	SZ2	YE5	Alarm 3
21	TP	SY2	YE7	
22	TU	YE	MB52	
23	MJ	0	MB46	
24	TU	YF	MB52	Alarm 4
25	MJ	0	MB46	
26	TP	ME24	YG6	Alarm 5
27	TU	YG	MB52	
30	MJ	0	MB46	
31	TP	SZ2	YH6	Alarm 6
32	TU	YH	MB52	
33	MJ	0	MB46	
34	TP	SZ2	YI2	Alarm 10
35	TU	YI	MB52	
36	MJ	0	MF113	
37	TP	SZ2	YJ6	Alarm 11
40	TU	YJ	MB52	
41	MJ	0	MB46	
42	TP	SZ2	YK6	Alarm 12
43	TU	YK	MB52	
44	MJ	0	MF111	
45	TP	SZ2	YM6	Alarm 14 = 12A
46	TU	YM	MB52	
47	MJ	0	MF111	
50	TP	SZ2	YL10	Alarm 13
51	TU	YL	MB52	
52	MJ	0	MB46	
53	TP	SZ2	YN6	Alarm 15
54	TU	YN	MB52	
55	MJ	0	MF107	
56	TP	SZ2	Y04	Alarm 16
57	TU	Y0	MB52	
60	MJ	0	MF107	

```
61    TP    SZ2     YP4
62    TU    YP      MB52     Alarm 17
63    MJ    0       MB46
64    TP    SZ2     YQ5
65    TU    YQ      MB52     Alarm 20
66    MJ    0       MB46
67    TP    SZ2     YR4
70    TP    SZ3     YR5
71    TU    YR      MB52     Alarm 21
72    MJ    0       MF107
73    TP    SZ2     YS4
74    TP    SZ3     YS5
75    TU    YS      MB52     Alarm 22
76    MJ    0       MB46
 77   TP    SZ2     YT6
100   TP    SY2     YT7      Alarm 23 = 1A
101   TU    YT      MB52
102   MJ    0       MB46
103   RJ    MB54    MF20     Alarm old 2 = 3A
104   MJ    0       ML1
105   TP    SZ2     YH6
106   TU    YH      MB52
107   RJ    MB54    MB46     Alarm 6A
110   MJ    0       ML1
111   RJ    MB54    MB46
112   MJ    0       MB25
113   RJ    MB54    MB46
114   MJ    0       ML
115   RJ    MB54    MF61     Alarm 17
116   MJ    0       MI17
117   RJ    MB54    MF37     Alarm 11
120   MJ    0       MI17
121   RJ    MB54    MF24
122   MJ    0       MG
123   TU    YD      MB52
124   RJ    MB54    MB46     Alarm 2 (first symbol Δ .)
125   MJ    0       MG
      CA    MF126
```

Warning Alarm 1

	IA	YY		
0	0	YY1	0	
1	40	YY2	11	
2	26	51474	72401	Comma assumed preceding symbol[ ].
3	24	65656	74730	
4	27	01525	43026	
5	30	27345	03201	
6	77	77777	77777	
7	77	77777	77777	
10	0	0	0	SZ2
11	0	0	0	SZ3
12	01	22777	77777	
	CA	YY13		

Warning Alarm 2

	IA	YB		
0	0	YB1	0	
1	40	YB2	16	Typed value of function[ ]may not
2	66	73523	02701	correspond to stated arguments.
3	70	24466	73001	
4	51	31013	16750	
5	26	66345	15001	
6	0	0	0	SZ2
7	01	47247	30150	
10	51	66010	10101	
11	01	01010	10101	
12	01	01010	10126	
13	51	54543	06552	
14	51	50270	16651	
15	01	65662	46630	
16	27	01245	43267	
17	47	30506	66522	
	CA	YB20		

Alarm 1

```
      IA    YC
 0    0     YC1      0
 1    40    YC2      10          Symbol [ ] illegal
 2    65    73472    55146       in type sentence.
 3    01    77777    77777
 4    0     0        0           SZ2
 5    01    34464    63032
 6    24    46013    45001
 7    66    73523    00165
10    30    50663    05026
11    30    22777    77777
      CA    YC12
```

Alarm 2

```
      IA    YD
 0    0     YD1      0
 1    40    YD2      13
 2    31    34546    56601       First symbol space point.
 3    65    73472    55146       Sentence not further checked.
 4    01    65522    42630
 5    01    52513    45066
 6    22    01653    05066
 7    30    50263    00150
10    51    66013    16754
11    66    33305    40101
12    01    01010    10101
13    01    01010    12633
14    30    26453    02722
      CA    YD15
```

Alarm 3

```
     IA   YE
0    0    YE1     0
1    40   YE2     15
2    24   27442   42630        Adjacent symbols[ ][ ]
3    50   66016   57347        meaningless in type
4    25   51466   50177        sentence.
5    0    0       0            SZ2
6    01   77777   77777
7    0    0       0            SY2
10   01   47302   45034
11   50   32463   06565
12   01   34500   16673
13   52   30010   10101
14   01   01010   10101
15   01   01016   53050
16   66   30502   63022
     CA   YE17
```

Alarm 4

```
     IA   YF
0    0    YF1     0
1    40   YF2     6
2    52   24543   05066        Parentheses not correctly placed.
3    33   30653   06501
4    50   51660   12651
5    54   54302   66646
6    73   01524   62426
7    30   27227   77777
     CA   YF10
```

Alarm 5

```
      IA    YG
 0    0     YG1      0
 1    40    YG2      15          Subscripted variable[  ] has
 2    65    67256    52654       incorrect number of subscripts.
 3    34    52663    02701
 4    70    24543    42425
 5    46    30017    77777
 6    0     0        0           ME24
 7    01    33246    50134
10    50    26515    45430
11    26    66015    06747
12    25    30540    10101
13    01    01010    10101
14    01    01010    15131
15    01    65672    56526
16    54    34526    66522
      CA    YG17
```

Alarm 6

```
      IA    YH
 0    0     YH1      0
 1    40    YH2      14
 2    24    46523    32450       Alpha-numeric symbol[  ]has
 3    67    47305    43426       digit as first character.
 4    01    65734    72551
 5    46    01777    77777
 6    0     0        0           SZ2
 7    01    33246    50127
10    34    32346    60124
11    65    01313    45465
12    66    01010    10101
13    01    01010    10101
14    01    01263    32454
15    24    26663    05422
      CA    YH16
```

Alarm 10

```
      IA    YI
 0    0     YI1      0
 1    40    YI2      23
 2    0     0        0          SZ2
 3    01    24525    23024      [ ] Appears as function or
 4    54    65012    46501      subscripted variable, but is
 5    24    01316    75026      not previously mentioned in problem.
 6    66    34515    00151
 7    54    01656    72565
10    26    54345    26630
11    27    01010    10101
12    01    01010    10101
13    01    01010    10101
14    01    70245    43424
15    25    46302    10125
16    67    66013    46501
17    50    51660    15254
20    30    70345    16765
21    46    73014    73050
22    66    34515    03027
23    01    34500    15254
24    51    25463    04722
      CA    YI25
```

Alarm 11 = Alarm 14

```
      IA    YJ
 0    0     YJ1      0
 1    40    YJ2      11
 2    31    46512    46634      Floating point variable[ ]
 3    50    32015    25134      used as subscript.
 4    50    66017    02454
 5    34    24254    63001
 6    0     0        0       SZ2
 7    01    67653    02701
10    24    65016    56725
11    65    26543    45266
12    22    77777    77777
      CA    YJ13
```

Alarm 12

	IA	YK		
0	0	YK1	0	
1	40	YK2	14	Pseudo operation symbol[ ]
2	52	65306	72751	illegal in type sentence.
3	01	51523	05424	
4	66	34515	00165	
5	73	47255	14601	
6	0	0	0	SZ2
7	01	34464	63022	
10	24	46013	45001	
11	66	73523	00101	
12	01	01010	10101	
13	01	01010	10101	
14	01	01016	53050	
15	66	30502	63022	
	CA	YK16		

Alarm 13

	IA	YL		
0	0	YL1	0	
1	40	YL2	14	
2	65	67256	52654	Subscripts of subscripted
3	34	52666	50151	variable[ ] are missing.
4	31	01656	72565	
5	26	54345	26630	
6	27	01702	45434	
7	24	25463	00177	
10	0	0	0	SZ2
11	01	24543	00101	
12	01	01010	10101	
13	01	01010	10101	
14	01	01010	14734	
15	65	65345	03222	
	CA	YL16		

Alarm 14 = 12A

	IA	YM		
0	0	YM1	0	
1	40	YM2	14	
2	46	34255	42454	Library Routine symbol [ ]
3	73	01545	16766	illegal in type sentence.
4	34	50300	16573	
5	47	25514	60177	
6	0	0	0	SZ2
7	01	34464	63032	
10	24	46013	45001	
11	66	73523	00101	
12	01	01010	10101	
13	01	01010	10101	
14	01	01016	53050	
15	66	30502	63022	
	CA	YM16		

Alarm 15

	IA	YN		
0	0	YN1	0	
1	40	YN2	20	
2	31	46512	46634	Floating point variable[ ] used as function symbol
3	50	32015	25134	or as subscripted variable.
4	50	66017	02454	
5	34	24254	63001	
6	0	0	0	SZ2
7	01	67653	02701	
10	24	65013	16750	
11	26	66345	15001	
12	01	01010	10101	
13	01	01010	10101	
14	65	73472	55146	
15	01	51540	12465	
16	01	65672	56526	
17	54	34526	63027	
20	01	70246	43424	
21	25	46302	27777	
	CA	YN22		

Alarm 16

```
      IA    YO
 0    0     YO1     0
 1    40    YO2     16          Subscript[ ] appears as
 2    65    67256   52654       function or subscripted
 3    34    52660   17777       variable.
 4    0     0       0           SZ2
 5    01    24525   23024
 6    54    65012   46501
 7    31    67502   66634
10    51    50015   15401
11    01    01010   10101
12    01    01010   10101
13    01    01010   10101
14    65    67256   52654
15    34    52663   02701
16    70    24543   42425
17    46    30227   77777
      CA    YO20
```

Alarm 17

```
      IA    YP
0     0     YP1     0
1     40    YP2     11          Subscript[ ] used as
2     65    67256   52654       argument of a function.
3     34    52660   17777
4     0     0       0           SZ2
5     01    67653   02701
6     24    65012   45432
7     67    47305   06601
10    51    31012   40131
11    67    50266   63451
12    50    22777   77777
      CA    YP13
```

Alarm 20

```
      IA    YQ
0     0     YQ1     0
1     40    YQ2     10          Function symbol [ ]
2     31    67502   66634       appears as subscript.
3     51    50016   57347
4     25    51460   17777
5     0     0       0           SZ2
6     01    24525   23024
7     54    65012   46501
10    65    67256   52654
11    34    52662   27777
      CA    YQ12
```

Alarm 21

```
      IA    YR
 0    0     YR1      0
 1    40    YR2      17
 2    26    51506    56624        Constant [ ] appears as
 3    50    66017    77777        function or subscripted variable
 4    0     0        0            SZ2
 5    0     0        0            SZ3
 6    01    24525    23024
 7    54    65012    46501
10    31    67502    66634
11    51    50015    15401
12    01    01010    10101
13    01    01010    10101
14    01    01010    10101
15    01    01016    56725
16    65    26543    45266
17    30    27017    02454
20    34    24254    63022
      CA    YR21
```

Alarm 22

```
      IA    YS
 0    0     YS1      0
 1    40    YS2      15
 2    26    51506    56624        Constant [ ] appears as
 3    50    66017    77777        variable to be typed
 4    0     0        0            SZ2
 5    0     0        0            SZ3
 6    01    24525    23024
 7    54    65012    46501
10    70    24543    42425
11    46    30016    65101
12    25    30010    10101
13    01    01010    10101
14    01    01010    10101
15    01    01010    10101
16    66    73523    02722
      CA    YS17
```

Alarm 23 = 1A

	IA	YT		
0	0	YT1	0	
1	40	YT2	10	Negative subscript [ ] illegal.
2	50	30322	46634	
3	70	30016	56725	
4	65	26543	45266	
5	01	77777	77777	
6	0	0	0	SZ2
7	0	0	0	SY2
10	01	34464	63032	
11	24	46227	77777	
	CA	YT12		

Headings for Warnings and Alarms

	IA	YZ		
0	0	YZ2	7	
1	0	YZ2	5	
2	65	30506	63050	
3	26	30010	17777	
4	0	0	0	Sentence no. Sentence ΔΔ [ ] ΔΔΔ (Type) ΔΔ
5	01	01011	76673	
6	52	30430	10177	Warning, ΔΔ
7	71	24545	03450	
10	32	21010	17777	
	CA	YZ11		

Constants

	IA	MC		
0	0	1	0	u adv. and CW for (
1	01	22777	77777	XS3 code of Δ .
2	23	77777	77777	XS3 code of ;
3	21	77777	77777	XS3 code of ,
4	43	77777	77777	XS3 code of)
5	17	77777	77777	XS3 code of (
6	0	ME13	MG75	For symbols with CW in ME13
7	0	SZ2	MG4	
10	0	TA4	MJ	For symbols found in Comb. List
11	0	TA3	0	
12	0	0	0	Zero
13	40	0	0	Indicator
14	0	0	1	One
15	0	0	2	Two
16	0	0	75000	For Dummy CW
17	0	0	64000	For CW fixed point variable
20	0	0	65000	For CW single val.
21	0	0	66000	For CW function
22	0	0	77000	For CW and mask
23	0	0	77777	Mask
24	0	77777	0	Mask
25	0	MC26	0	Address of CW for comma
26	0	0	40	CW for comma
27	0	3	0	For adv. u
30	0	2	0	For adv. u and CW last par.
31	0	0	70000	Mask for pseudo op
32	0	0	120	CW for Δ
33	24	50277	77777	"and"
34	02	77777	77777	"---" XS3 lower case
35	0	77777	77777	"---" XS3 upper case
36	0	WL3	WL3	Constant for EW3
37	0	0	40000	
40	0	WL	WL	
	CA	MC41		

Temporaries: Type String-out

ME	0-12	Storage for SY2-14
	13	Temp. storage for CW (call word)
	14	Temp. storage for new MB3
	15	Extra Level Meter
	16	Word index
	17	Level Meter
	20	Index "inside subscripted variable"
	21	Index "inside subscripted function"
	22	Counter for subscripts
	23	Stored value for number of subscripts
ME	24	Stored XS3 code of subscripted variable

List String-Out Routine

The List String-Out Routine translates the pseudo code "List" sentences from the corrected Problem Tape on Uniservo 5 and edits and stores the information in the format prescribed for input to the List Generation routine (see List Generation write-up). During the translation of the input sentence, extensive checks are made for strict adherence to the List sentence format specified in the Unicode manual, including the proper use of commas where required. When an error is detected, a statement describing the error condition is typed on the on-line Flexowriter. Errors are of two types; those which must be corrected before compilation can continue beyond the string-out (translation) phase, and those which give rise to a "warning" typeout on the Flexowriter to point out the inconsistency, but which do not preclude the successful compilation and execution of the Object Program. Errors of the latter kind include the word "warning" in the Flexowriter typeout. The texts of all error typeouts are included in the annotated coding. In general, the errors detected prior to the Title or Column Headings are of the first kind and those detected within the title or column headings are of the latter kind.

The list which is produced as input to the List Generation routine is referred to as the List String-out and is written on magnetic tape when it is completed. During translation, this routine records the call word for each of the variables to be listed as well as the call word designating the Uniservo on which the listing is to be made during the running of the Object Program. In addition to the above call words, the List String-out includes a section containing the headings for the listing and a count of the total number of words in the headings. The headings are stored in the list exactly as they are to be transferred to the listing tape.

The texts of the headings are included just as they appear on the input tape; however, they are edited for position on the High Speed Printer sheets. The title is edited so that it is always centered on the page, regardless of the number of characters it includes. The column headings and/or the names of the variables are edited so that they are centered on the page, regardless of the number. In addition, within the four words (23 characters) allowed for each column heading or variable name, the characters are positioned with

respect to the decimal point for floating point quantities or the assumed point for fixed point quantities. The assumed point is to the right of fixed point numbers.

Figure 1 shows the format of the heading list within the List String-out when all three sections are included, i.e., title, column headings and variable names. It also shows the position of the column headings and/or variable names within their assigned blockettes, depending on the number of such column heading and/or variable names.

Figure 2 shows the positions of the XS3 characters within the title blockette and within the four word items for the column headings and variable names.

If either the column headings, or title, or both are omitted from the input tape, the corresponding blockette(s) is omitted from the heading list in the List String-out, and the succeeding blockette(s) packed upward (see Fig. 1). The leftmost character of the first word of the heading list is a Fast Feed 1 symbol.

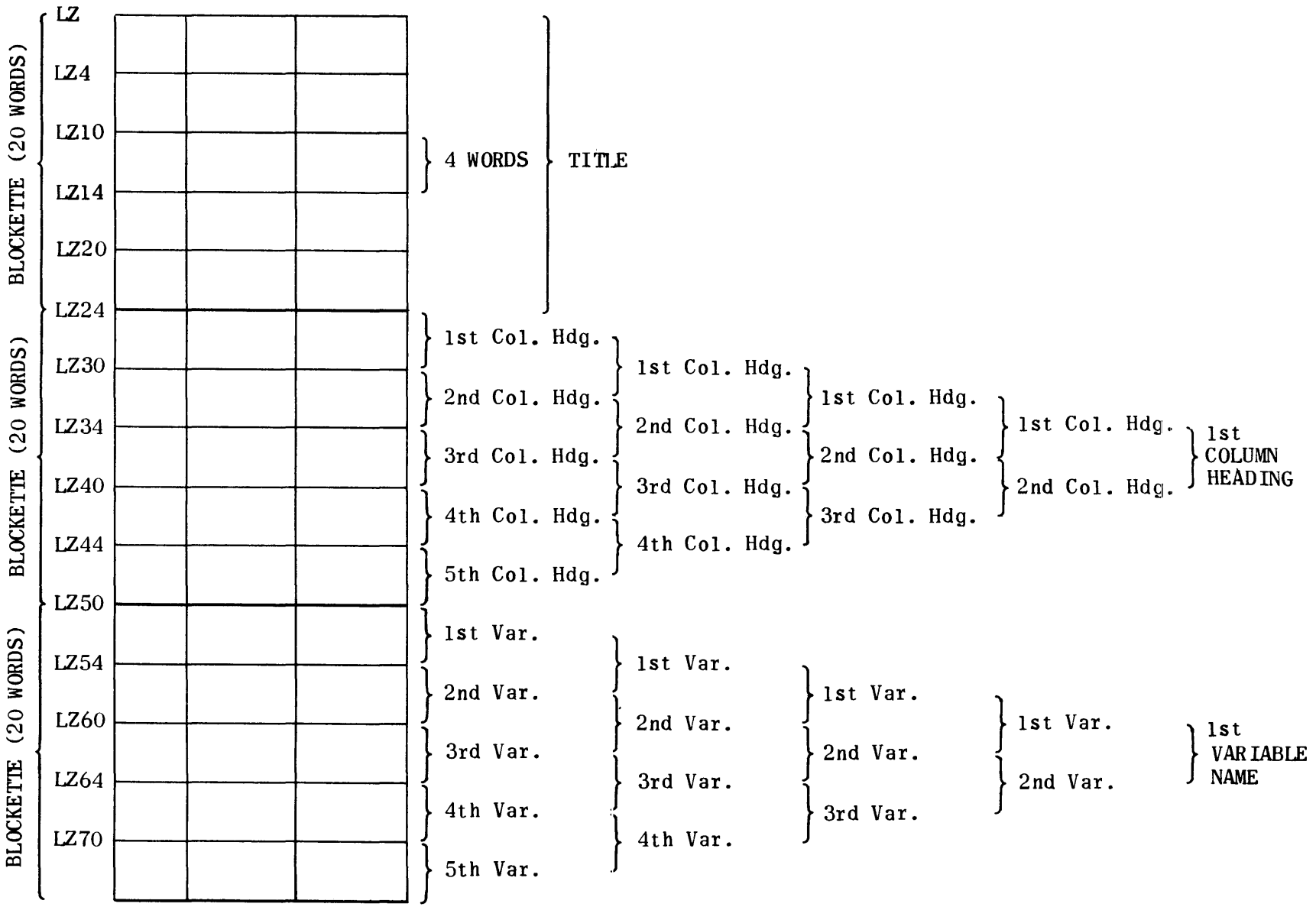
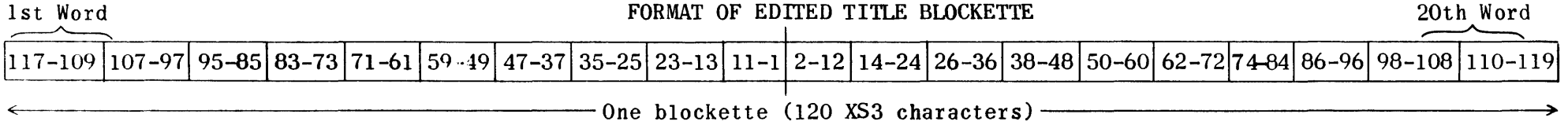


Figure 1. Heading List Format

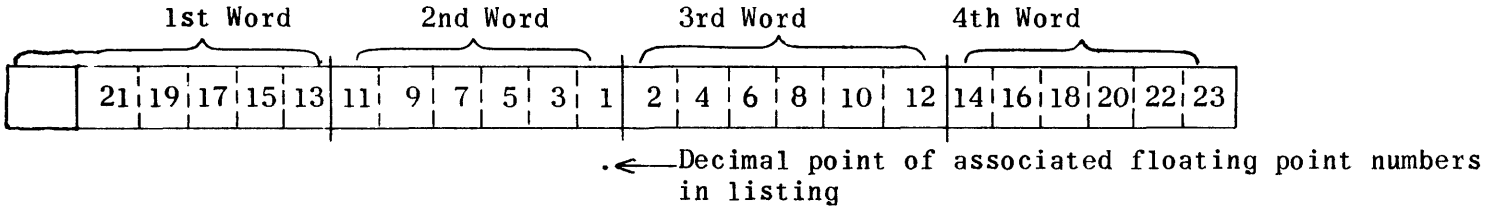
FORMAT OF EDITED TITLE BLOCKETTE



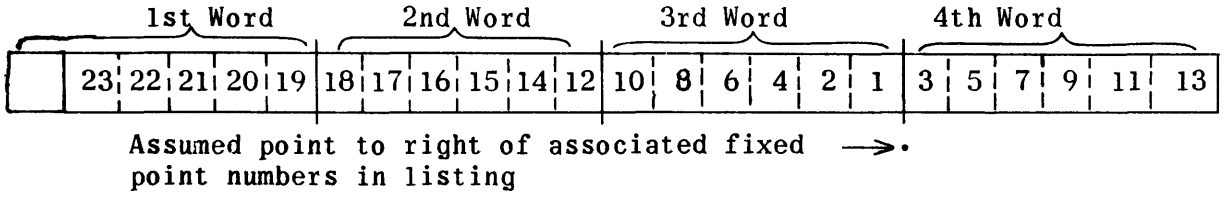
The numbers in the above diagram indicate positions of the XS3 characters of the title within the blockette, depending on the total number of characters. If the title has only one character, it appears to the left of center; if it has two characters, the first appears to the left of center, the other to the right; if it has three characters, the first two appear to the left of center and the third to the right of center; etc., only 119 of the 120 characters are allowed for the title.

FORMAT OF EDITED COLUMN HEADINGS OR VARIABLE NAMES

Floating Point Quantities (position about decimal point)

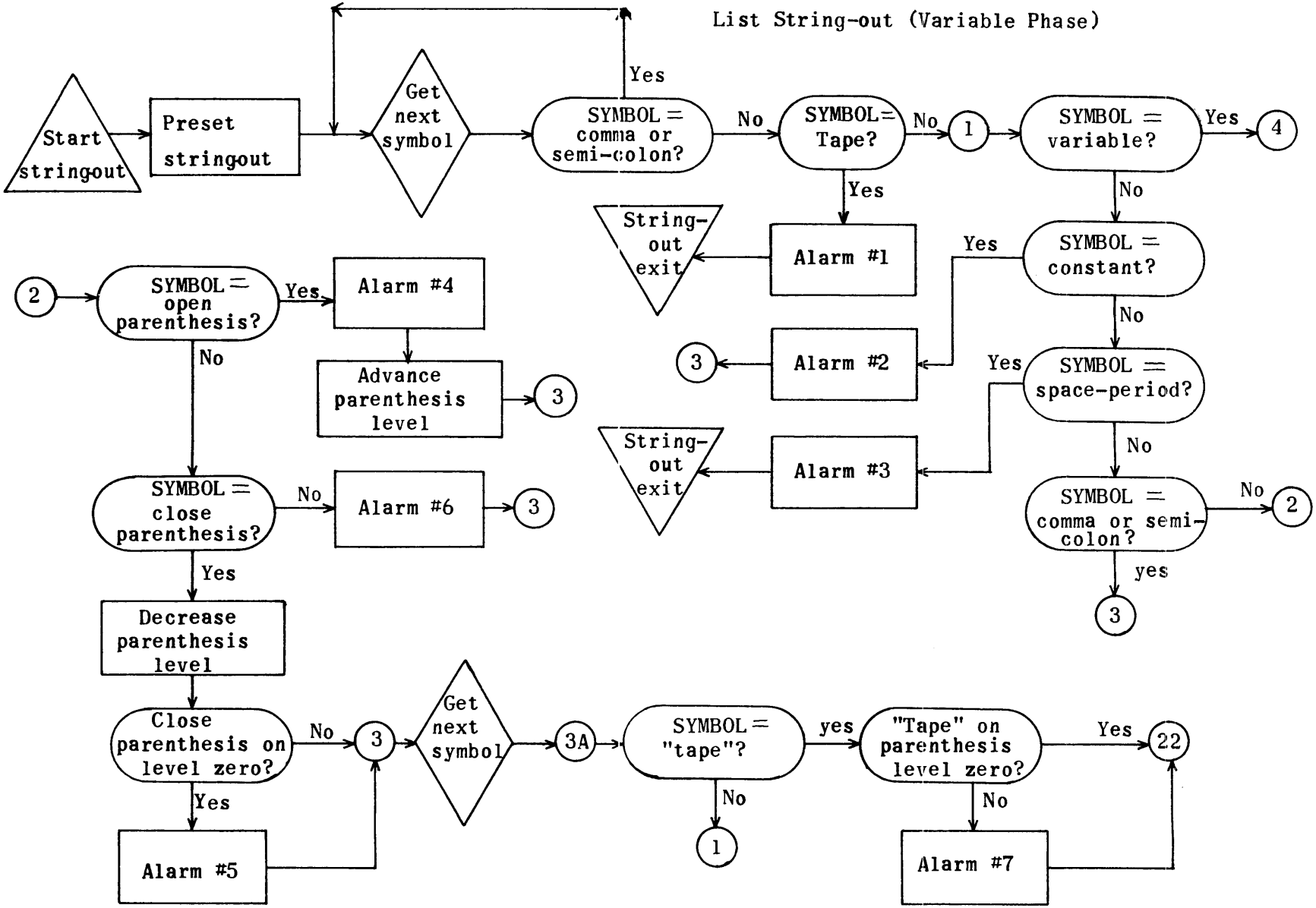


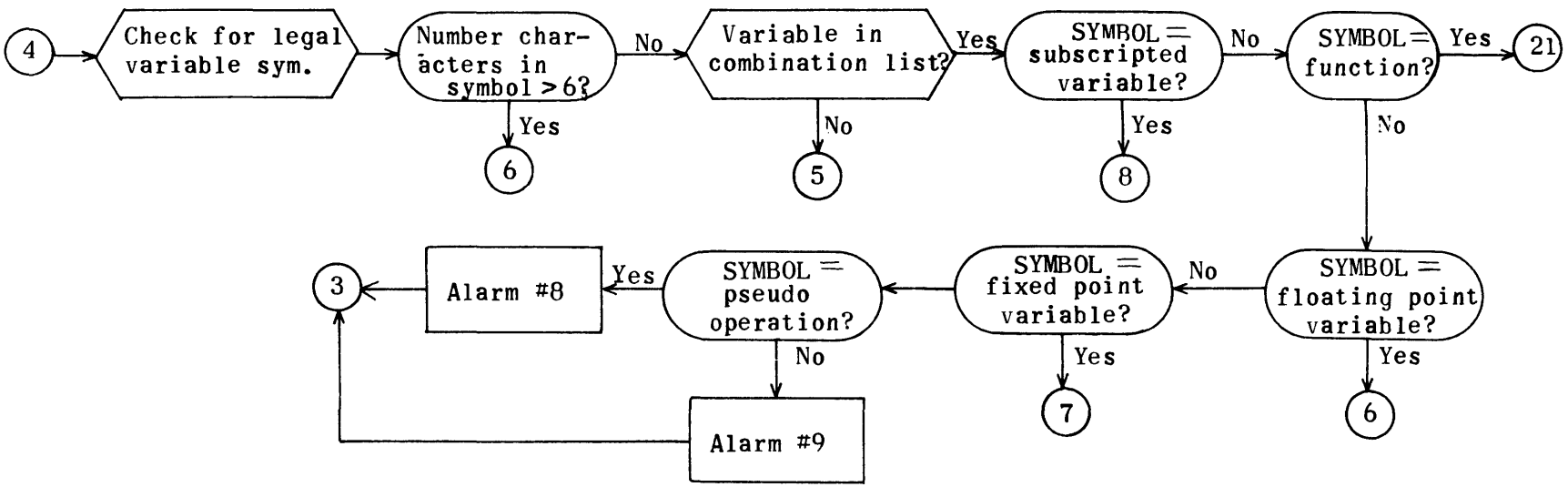
Fixed Point Quantities (position about assumed point)

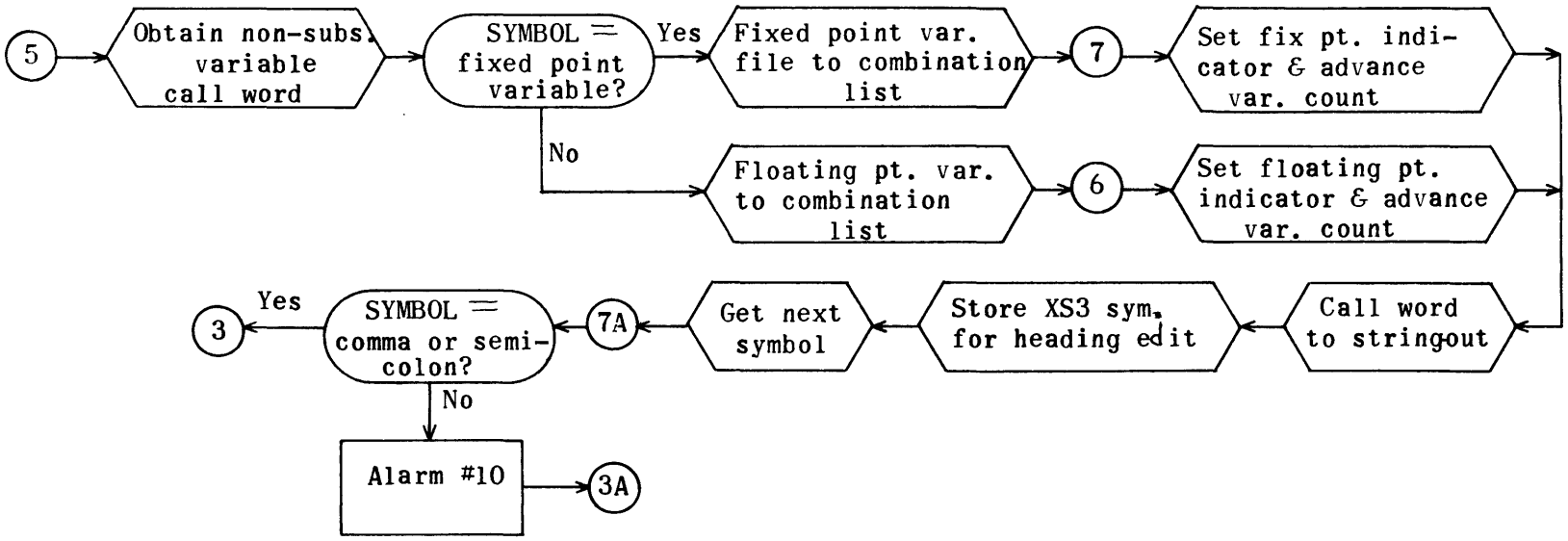


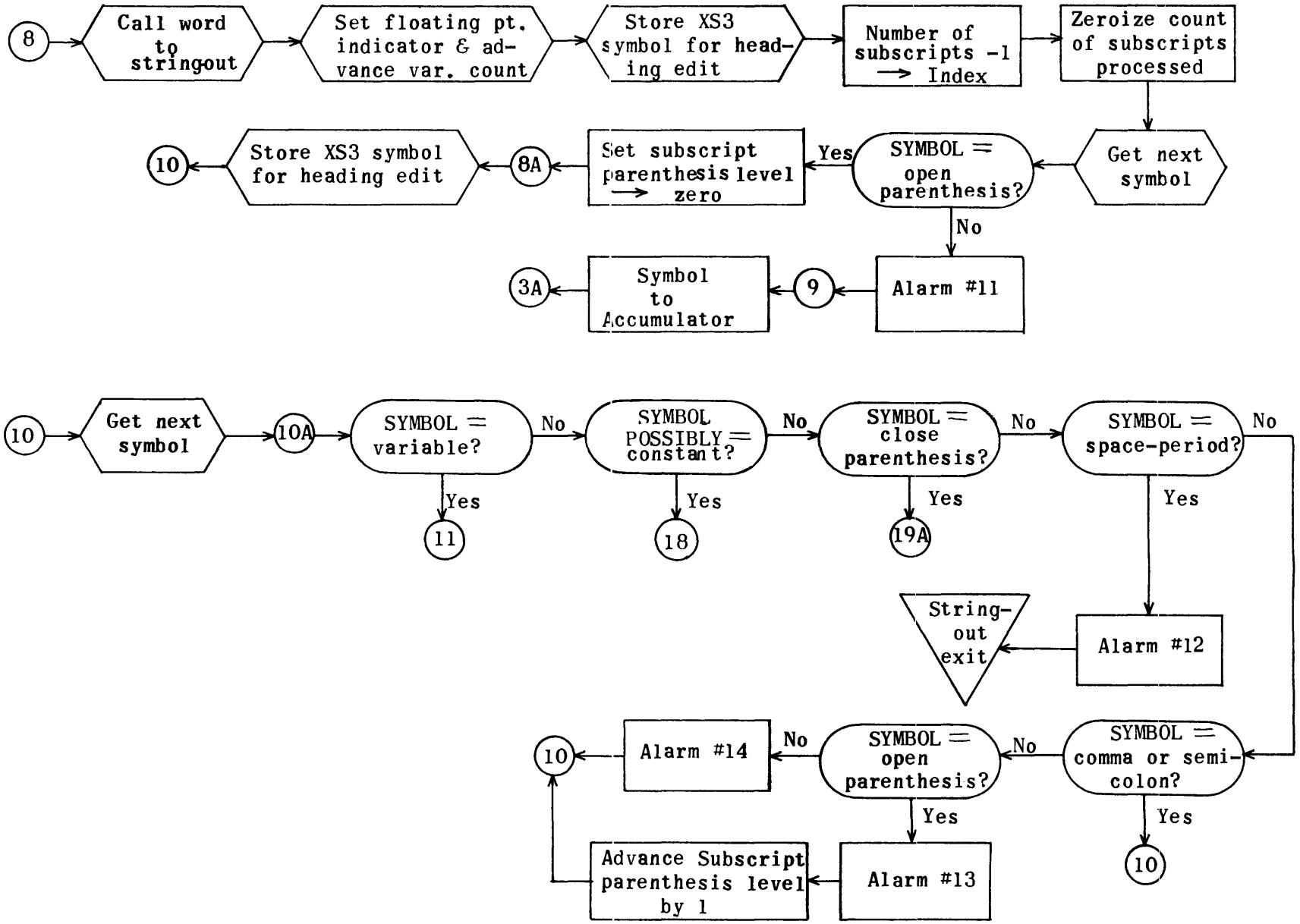
The numbers in the above diagram indicate positions of the XS3 characters of the column headings or variable names, depending on the total number of characters in the column heading or variable name.

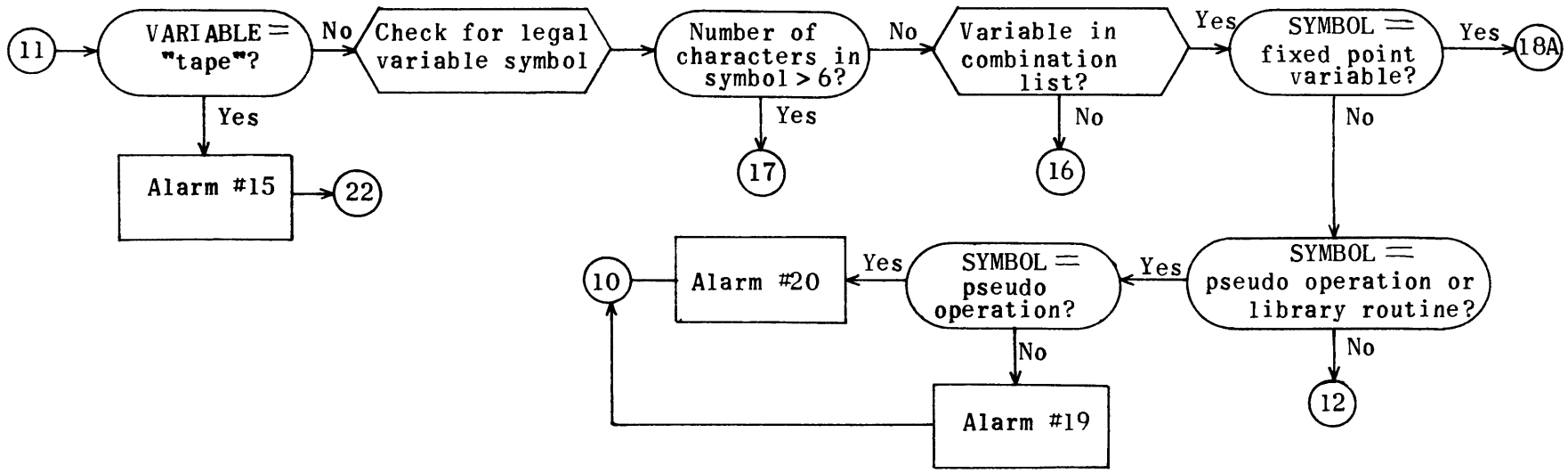
Figure 2. Format of Edited Title Blockette, Edited Column Headings or Variable Names, Fixed Point Quantities

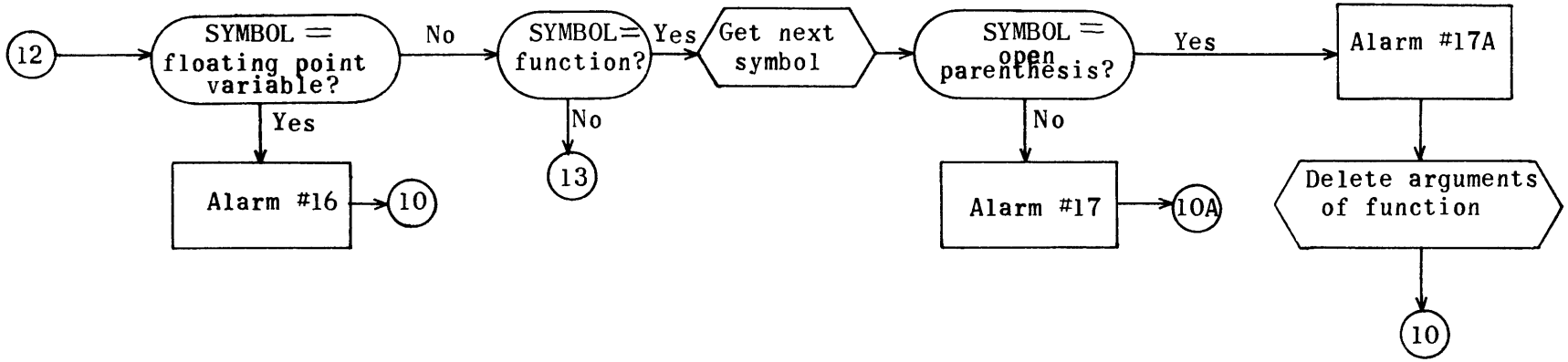

List String-out (Variable Phase)
Start stringout
Preset stringout
Get next symbol
SYMBOL = comma or semi-colon?
Yes
No
SYMBOL = Tape?
No
1
SYMBOL = variable?
Yes
4
Yes
No
String-out exit
Alarm #1
Yes
SYMBOL = constant?
2
SYMBOL = open parenthesis?
Yes
Alarm #4
No
Advance parenthesis level
3
3
Alarm #2
Yes
SYMBOL = space-period?
String-out exit
Alarm #3
No
SYMBOL = close parenthesis?
No
Alarm #6
3
SYMBOL = comma or semi-colon?
No
2
Yes
yes
3
Decrease parenthesis level
Close parenthesis on level zero?
No
3
Get next symbol
3A
SYMBOL = "tape"?
yes
"Tape" on parenthesis level zero?
Yes
22
Yes
No
1
No
Alarm #5
Alarm #7

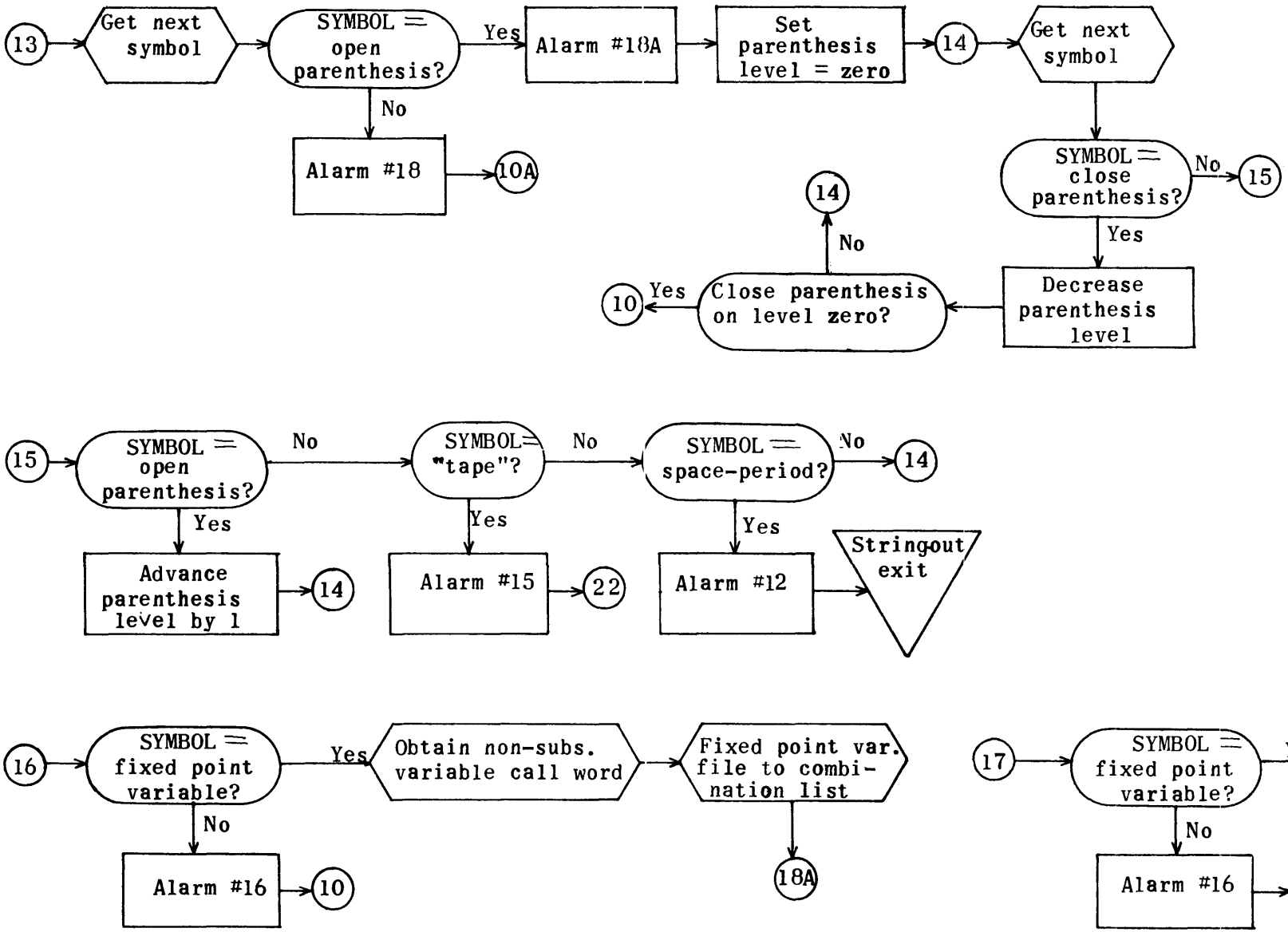

4
Check for legal variable sym.
Number characters in symbol > 6?
No
Yes
6
Variable in combination list?
Yes
No
5
SYMBOL = subscripted variable?
No
Yes
8
SYMBOL = function?
Yes
21
No
SYMBOL = floating point variable?
Yes
6
SYMBOL = fixed point variable?
No
Yes
7
SYMBOL = pseudo operation?
Yes
No
Alarm #8
3
Alarm #9

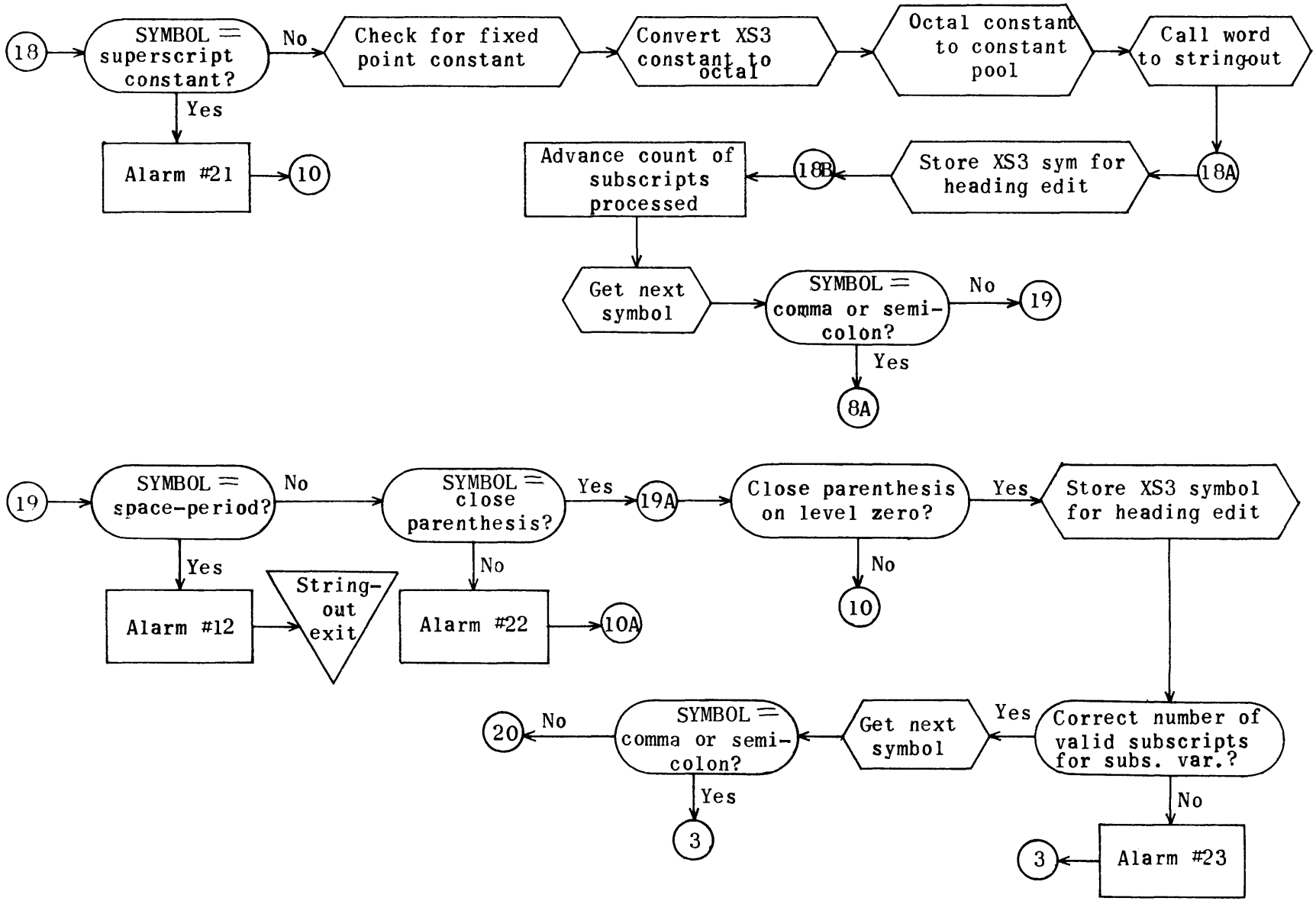

5
Obtain non-subs. variable call word
SYMBOL = fixed point variable?
Yes
No
Fixed point var. file to combination list
7
Set fix pt. indicator & advance var. count
Floating pt. var. to combination list
6
Set floating pt. indicator & advance var. count
Call word to stringout
Store XS3 sym. for heading edit
Get next symbol
7A
SYMBOL = comma or semicolon?
Yes
3
No
Alarm #10
3A

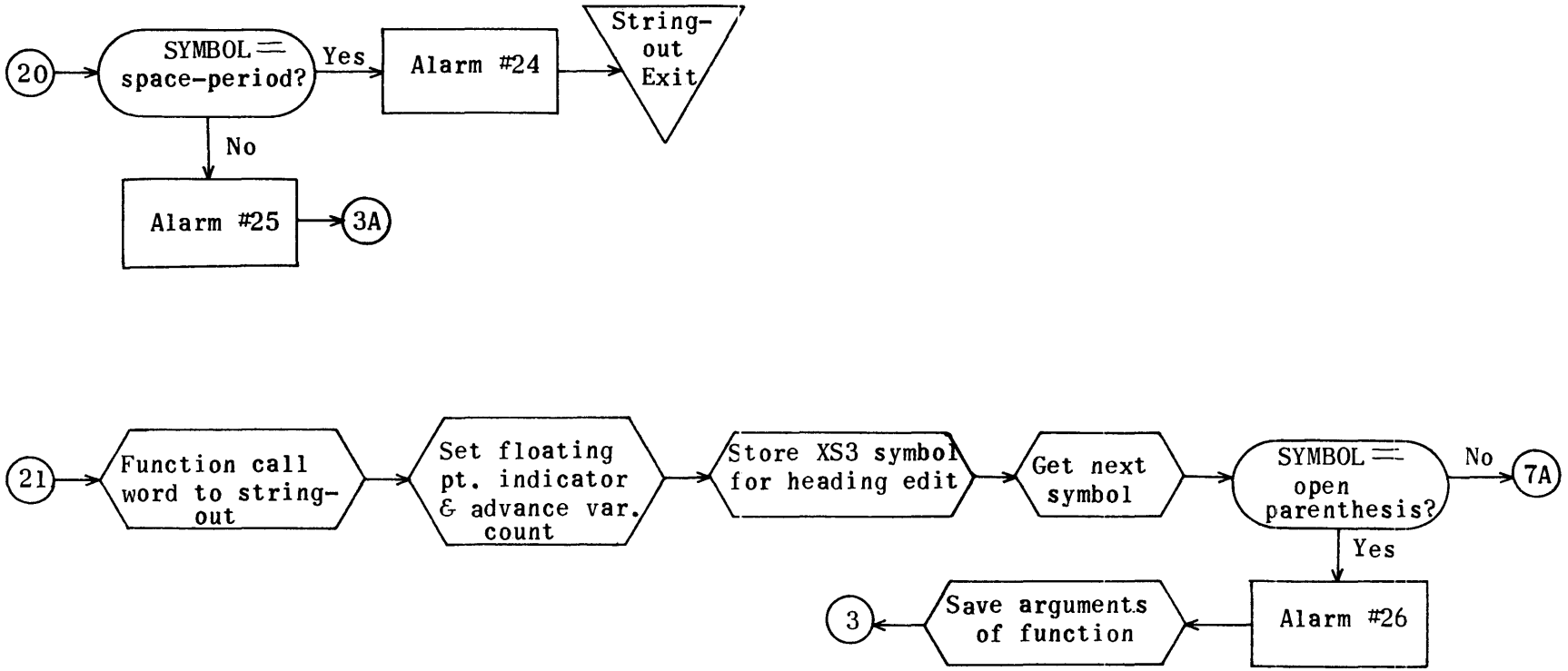

8
Call word to stringout
Set floating pt. indicator & advance var. count
Store XS3 symbol for heading edit
Number of subscripts -1 → Index
Zeroize count of subscripts processed
Get next symbol
SYMBOL = open parenthesis?
Yes
No
Set subscript parenthesis level → zero
8A
Store XS3 symbol for heading edit
10
Alarm #11
9
Symbol to Accumulator
3A
10
Get next symbol
10A
SYMBOL = variable?
No
Yes
11
SYMBOL POSSIBLY = constant?
No
Yes
18
SYMBOL = close parenthesis?
No
Yes
19A
SYMBOL = space-period?
No
Yes
String-out exit
Alarm #12
SYMBOL = comma or semi-colon?
No
Yes
10
SYMBOL = open parenthesis?
No
Yes
Alarm #14
10
Alarm #13
Advance Subscript parenthesis level by 1


11
VARIABLE = "tape"?
No
Check for legal variable symbol
Number of characters in symbol > 6?
No
Variable in combination list?
Yes
SYMBOL = fixed point variable?
Yes
18A
Yes
Alarm #15
22
Yes
17
No
16
No
10
Alarm #20
Yes
SYMBOL = pseudo operation?
Yes
SYMBOL = pseudo operation or library routine?
No
12
Alarm #19

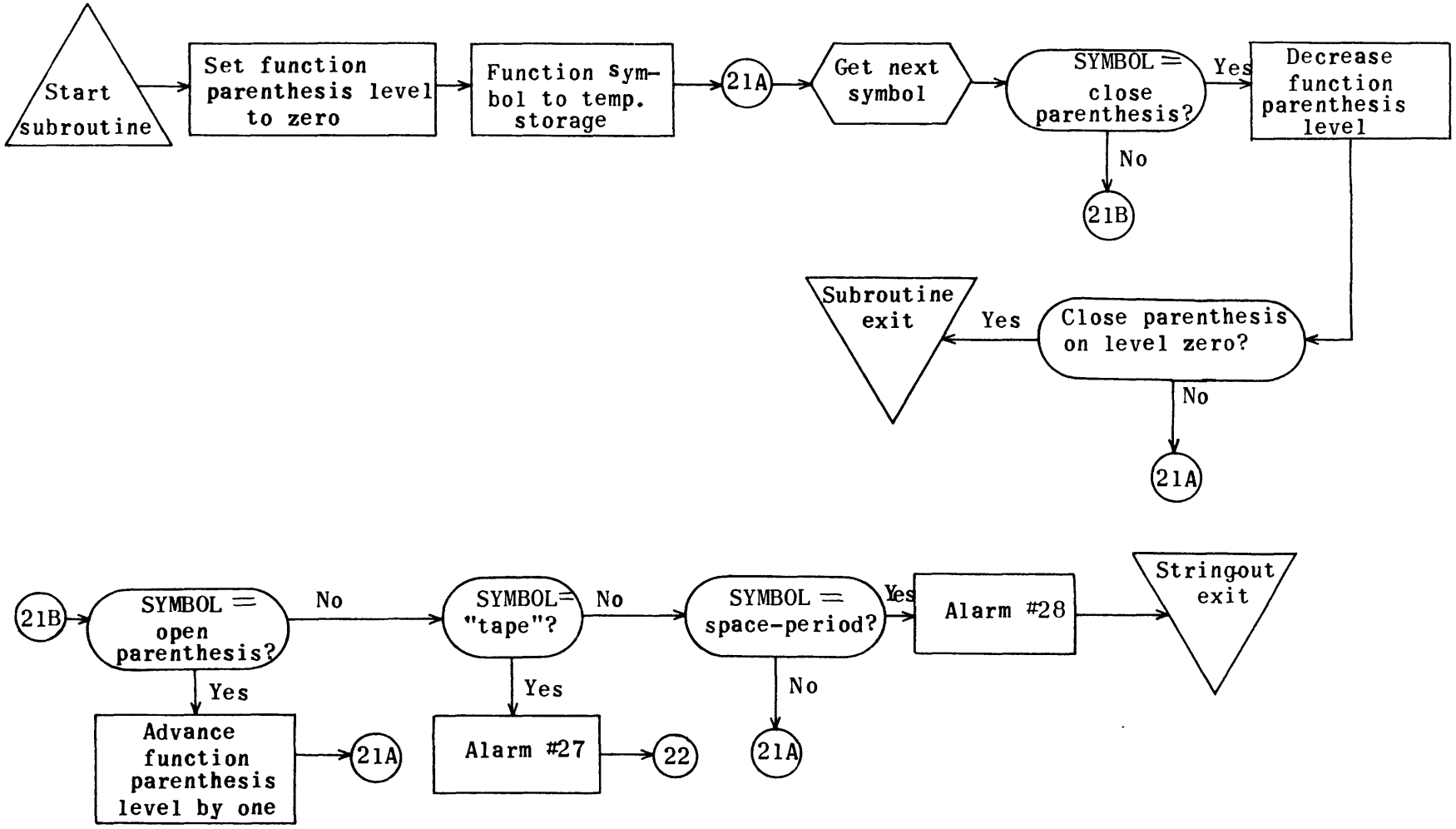

12
SYMBOL = floating point variable?
No
SYMBOL = function?
Yes
Get next symbol
SYMBOL = open parenthesis?
Yes
Alarm #17A
Yes
Alarm #16
10
No
13
No
Alarm #17
10A
Delete arguments of function
10

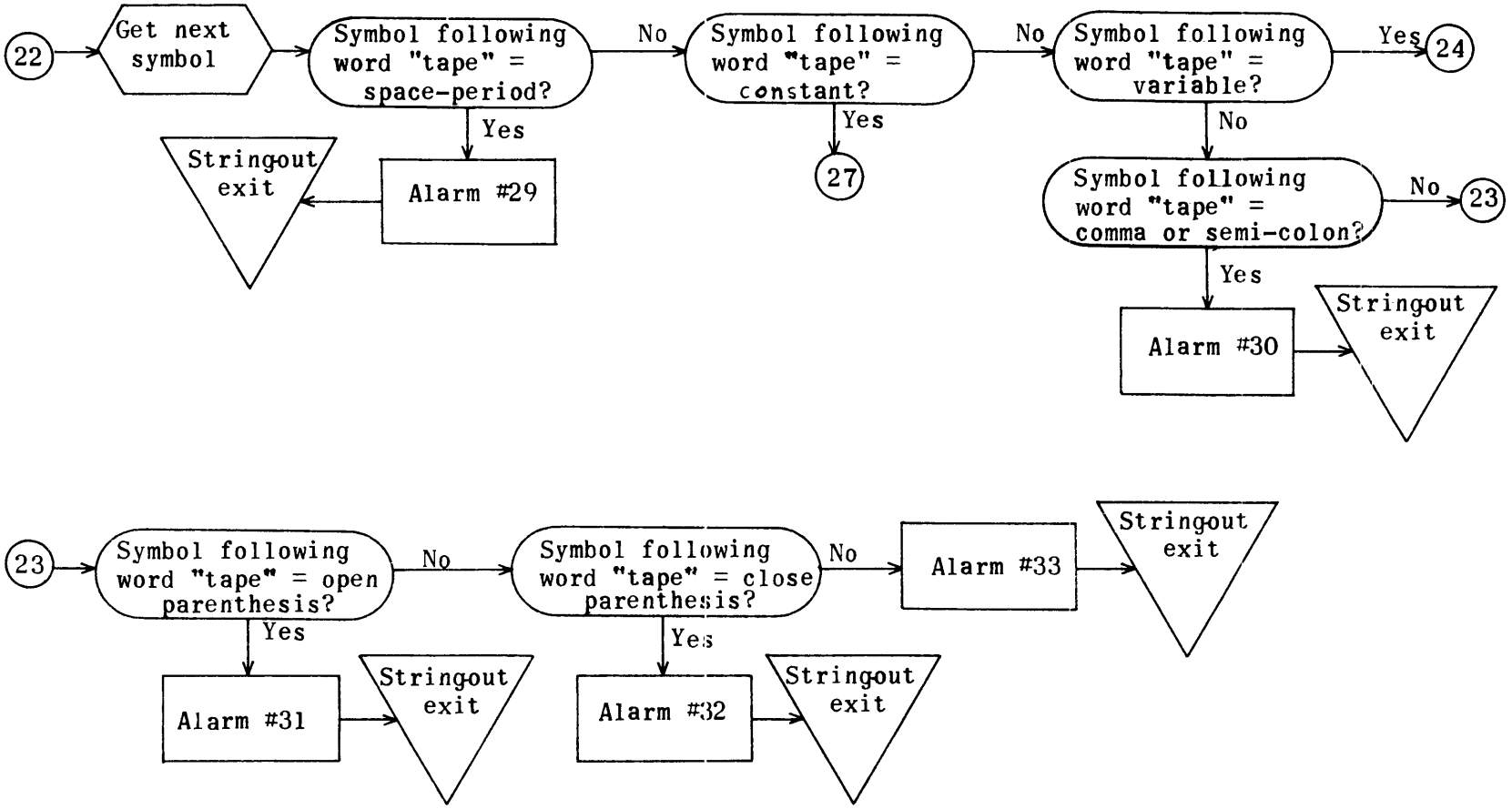

13
Get next symbol
SYMBOL = open parenthesis?
Yes
Alarm #18A
Set parenthesis level = zero
14
Get next symbol
No
Alarm #18
10A
SYMBOL = close parenthesis?
No
15
Yes
14
No
Decrease parenthesis level
Close parenthesis on level zero?
Yes
10
15
SYMBOL = open parenthesis?
No
SYMBOL = "tape"?
No
SYMBOL = space-period?
No
14
Yes
Yes
Yes
Advance parenthesis level by 1
14
Alarm #15
22
Alarm #12
Stringout exit
16
SYMBOL = fixed point variable?
Yes
Obtain non-subs. variable call word
Fixed point var. file to combination list
No
Alarm #16
10
18A
17
SYMBOL = fixed point variable?
Yes
18B
No
Alarm #16
10

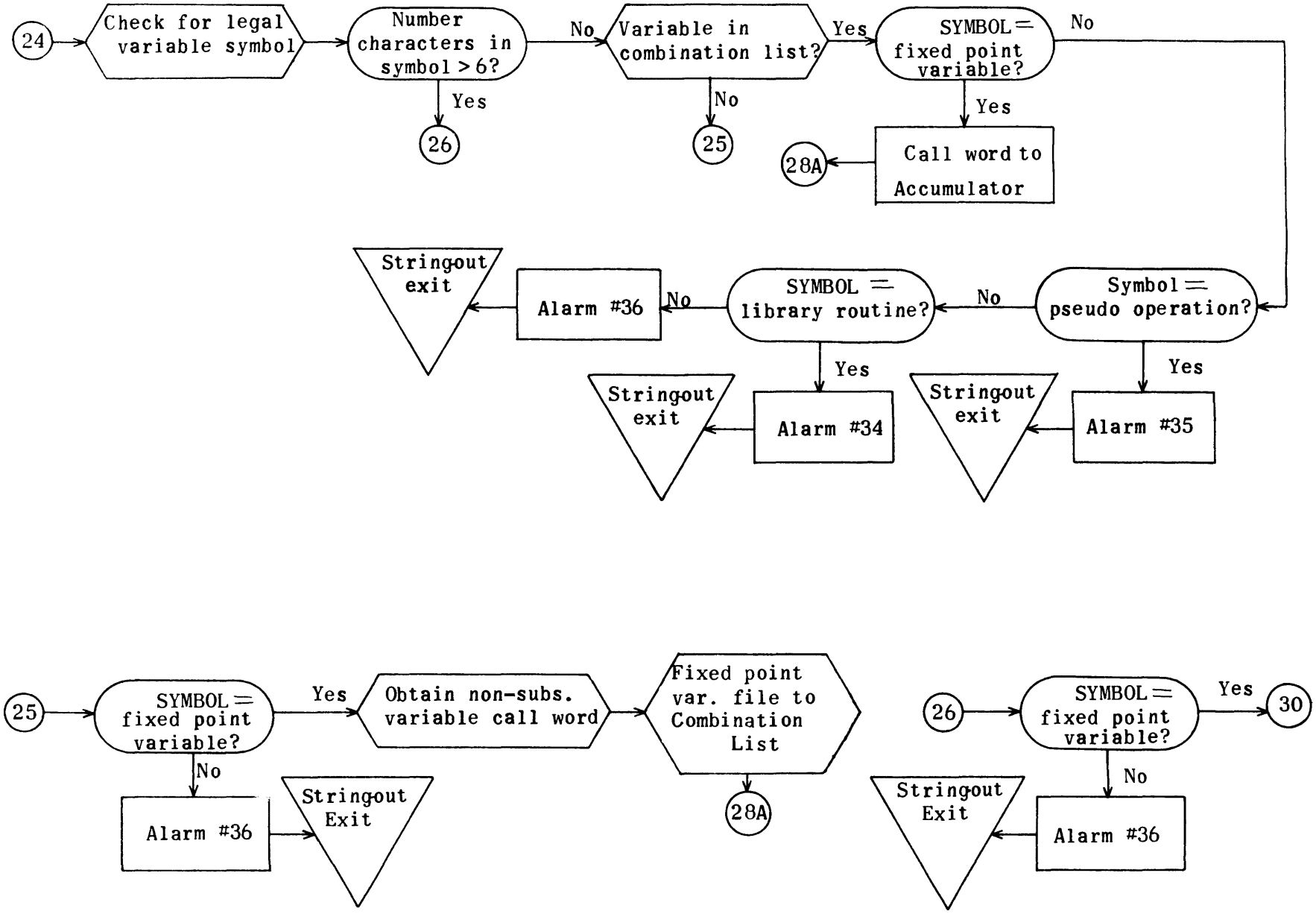

18
SYMBOL = superscript constant?
No
Check for fixed point constant
Convert XS3 constant to octal
Octal constant to constant pool
Call word to stringout
18A
Yes
Alarm #21
10
Advance count of subscripts processed
18B
Store XS3 sym for heading edit
Get next symbol
SYMBOL = comma or semi-colon?
No
19
Yes
8A
19
SYMBOL = space-period?
No
SYMBOL = close parenthesis?
Yes
19A
Close parenthesis on level zero?
Yes
Store XS3 symbol for heading edit
Yes
No
No
Alarm #12
String-out exit
Alarm #22
10A
10
20
No
SYMBOL = comma or semi-colon?
Get next symbol
Yes
Correct number of valid subscripts for subs. var.?
Yes
No
3
3
Alarm #23

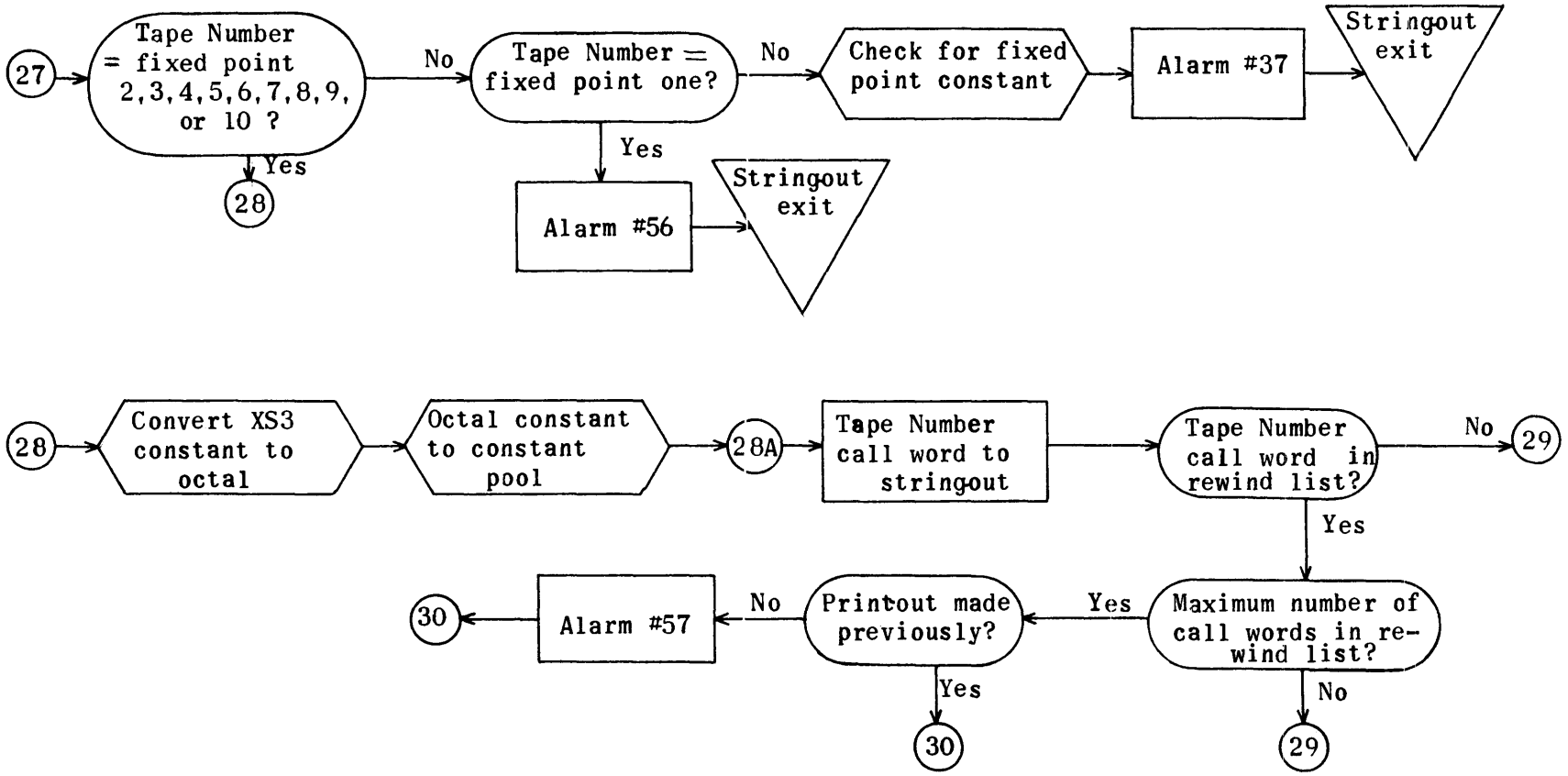

20
SYMBOL = space-period?
Yes
Alarm #24
String-out Exit
No
Alarm #25
3A
21
Function call word to string-out
Set floating pt. indicator & advance var. count
Store XS3 symbol for heading edit
Get next symbol
SYMBOL = open parenthesis?
No
7A
Yes
Alarm #26
Save arguments of function
3

Subroutine to Process Arguments of Function (Referenced by Alarms #17A & #26)

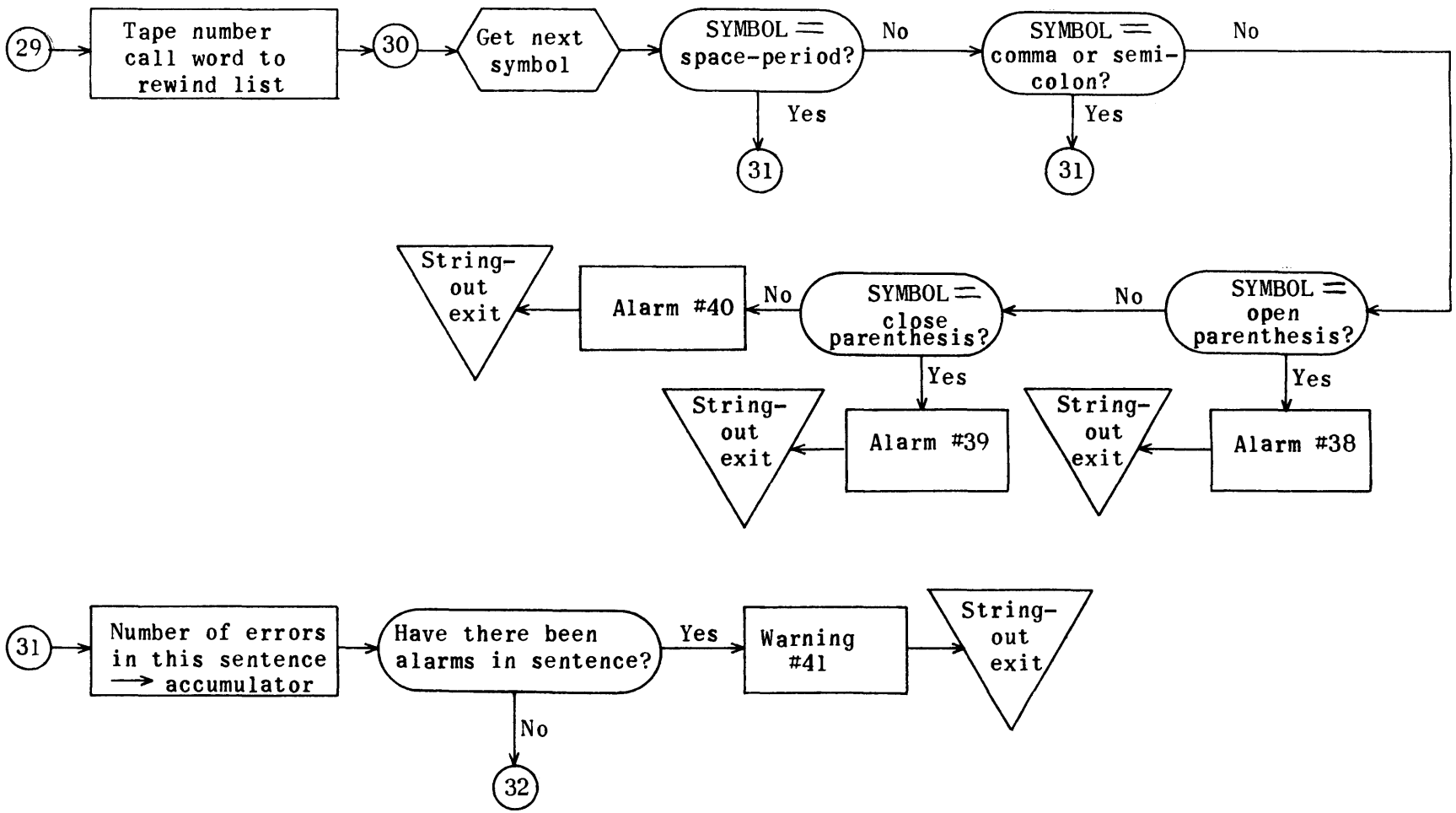


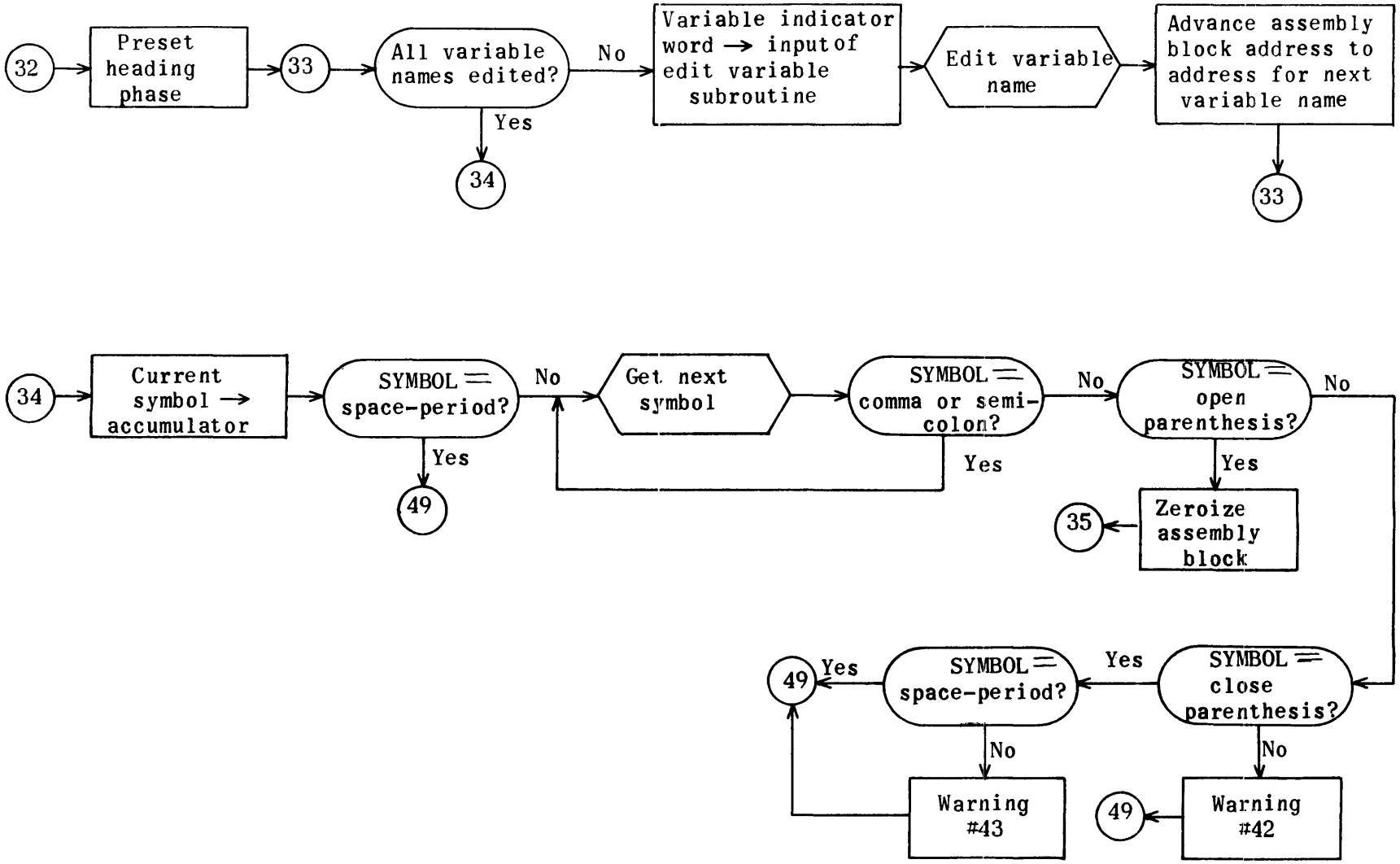

22
Get next symbol
Symbol following word "tape" = space-period?
No
Symbol following word "tape" = constant?
No
Symbol following word "tape" = variable?
Yes
24
Yes
Stringout exit
Alarm #29
27
Yes
No
Symbol following word "tape" = comma or semi-colon?
No
23
Yes
Alarm #30
Stringout exit
23
Symbol following word "tape" = open parenthesis?
No
Symbol following word "tape" = close parenthesis?
No
Alarm #33
Stringout exit
Yes
Yes
Alarm #31
Stringout exit
Alarm #32
Stringout exit

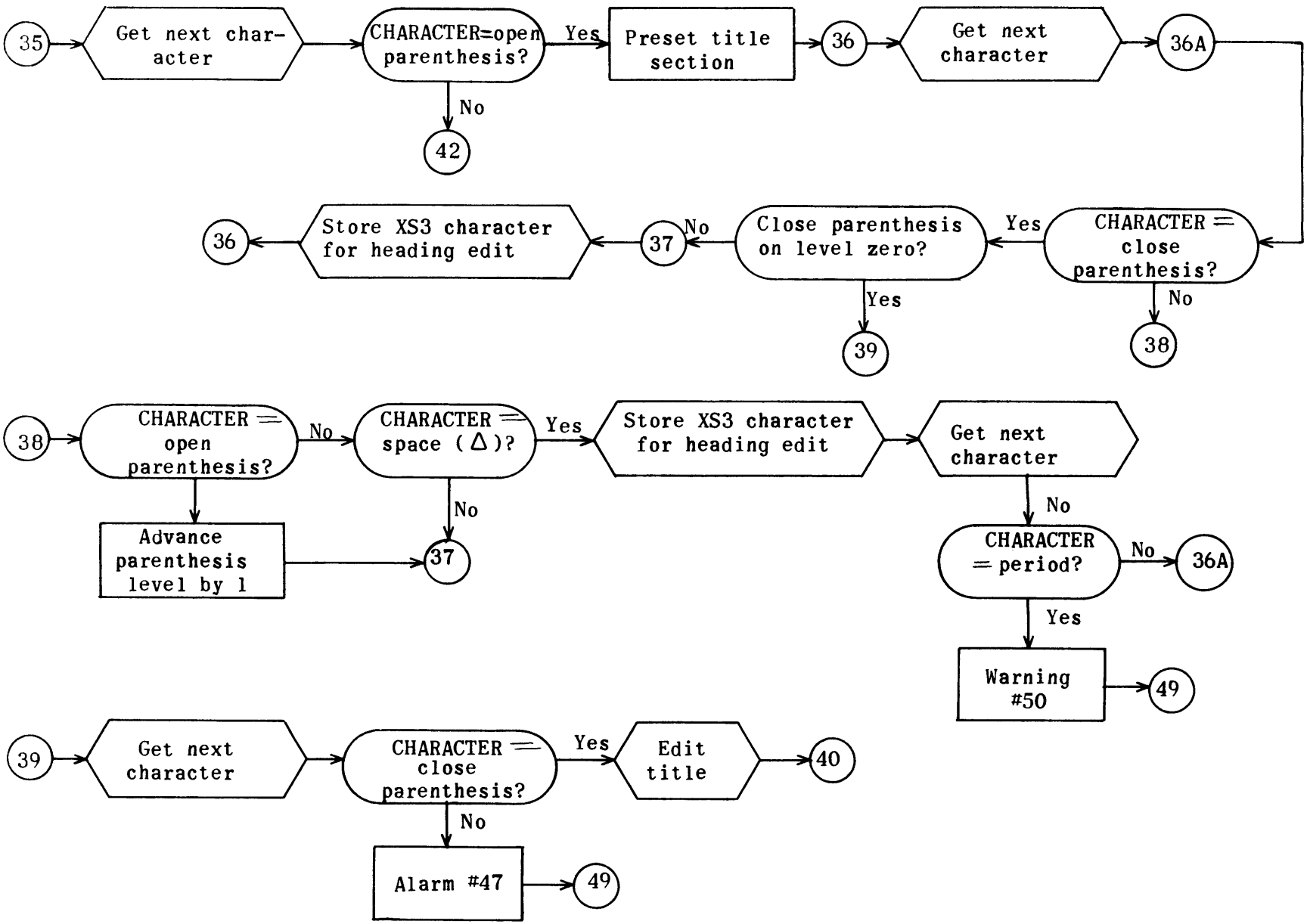

24
Check for legal variable symbol
Number characters in symbol > 6?
Yes
26
No
Variable in combination list?
No
25
Yes
SYMBOL = fixed point variable?
Yes
Call word to Accumulator
28A
No
Symbol = pseudo operation?
No
SYMBOL = library routine?
No
Alarm #36
Stringout exit
Yes
Alarm #34
Stringout exit
Yes
Alarm #35
Stringout exit
25
SYMBOL = fixed point variable?
Yes
Obtain non-subs. variable call word
Fixed point var. file to Combination List
28A
No
Alarm #36
Stringout Exit
26
SYMBOL = fixed point variable?
Yes
30
No
Alarm #36
Stringout Exit

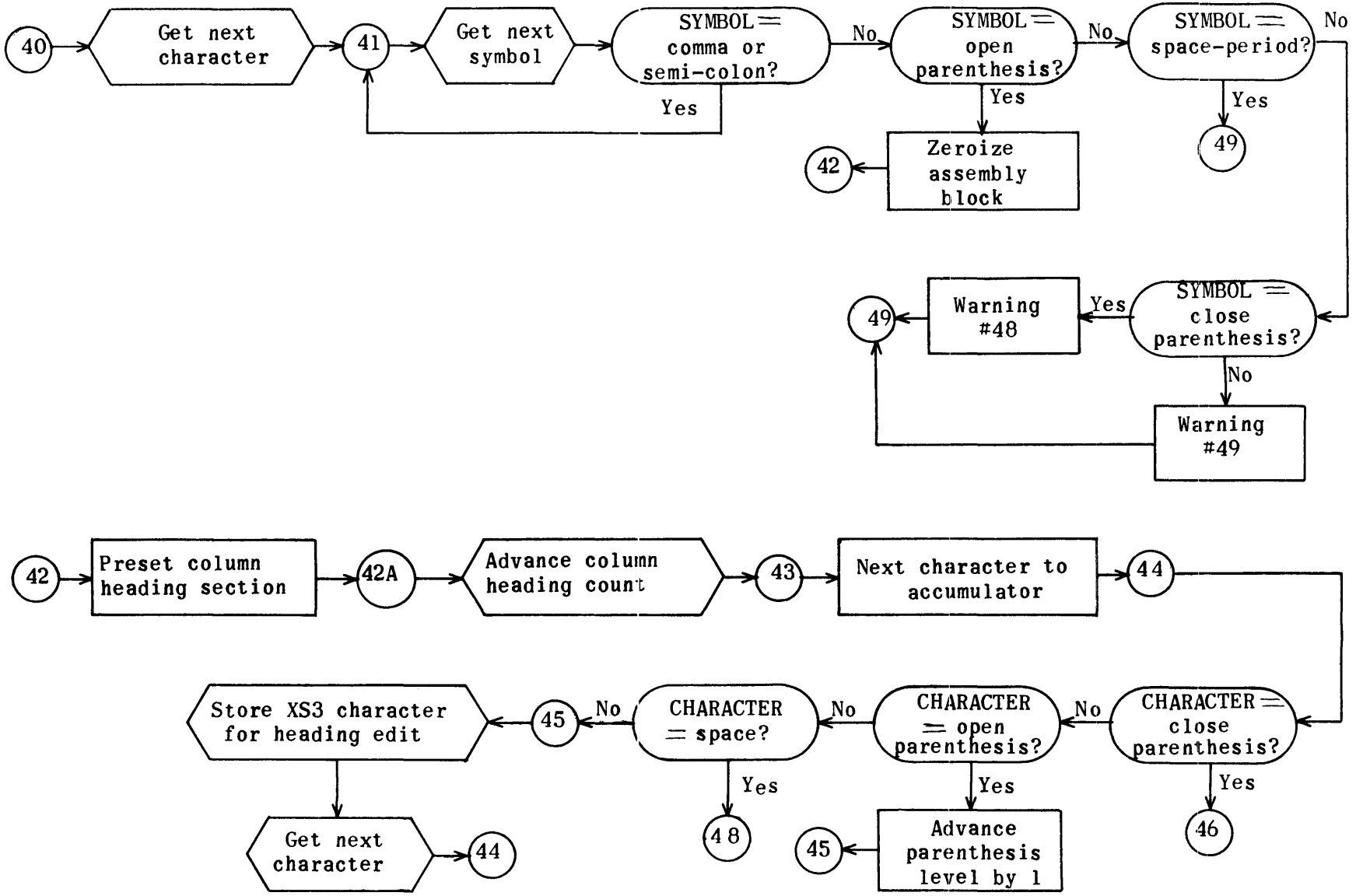

27
Tape Number = fixed point 2,3,4,5,6,7,8,9, or 10 ?
No
Tape Number = fixed point one?
No
Check for fixed point constant
Alarm #37
Stringout exit
Yes
28
Yes
Alarm #56
Stringout exit
28
Convert XS3 constant to octal
Octal constant to constant pool
28A
Tape Number call word to stringout
Tape Number call word in rewind list?
No
29
Yes
30
Alarm #57
No
Printout made previously?
Yes
Maximum number of call words in rewind list?
Yes
30
No
29

List Stringout (Tape Designation Phase, Cont.)

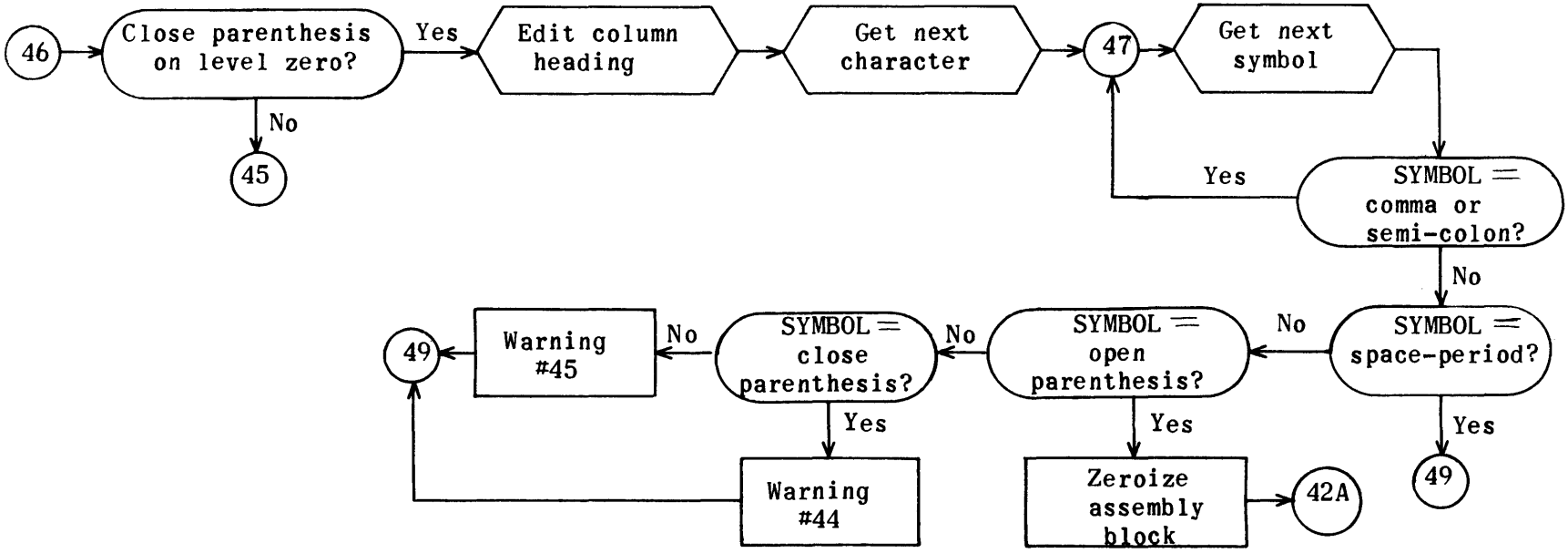


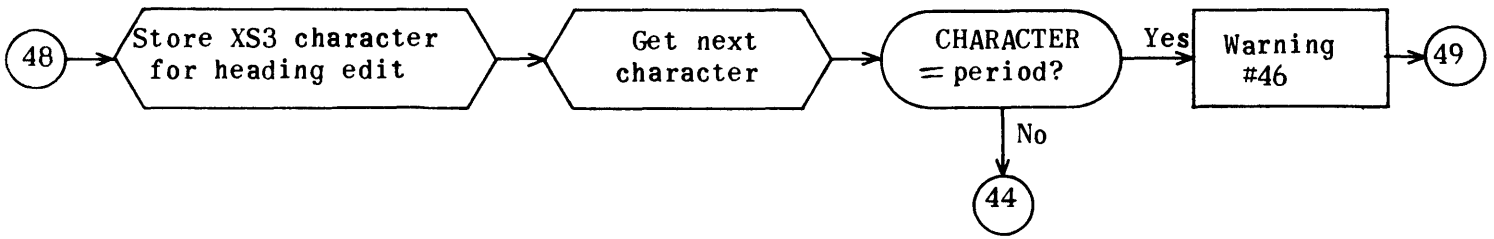

32
Preset heading phase
33
All variable names edited?
No
Yes
34
Variable indicator word → input of edit variable subroutine
Edit variable name
Advance assembly block address to address for next variable name
33
34
Current symbol → accumulator
SYMBOL = space-period?
No
Yes
49
Get next symbol
SYMBOL = comma or semi-colon?
No
Yes
SYMBOL = open parenthesis?
No
Yes
35
Zeroize assembly block
49
Yes
SYMBOL = space-period?
Yes
SYMBOL = close parenthesis?
No
No
Warning #43
49
Warning #42


35
Get next character
CHARACTER=open parenthesis?
Yes
Preset title section
36
Get next character
36A
No
42
36
Store XS3 character for heading edit
37
No
Close parenthesis on level zero?
Yes
CHARACTER = close parenthesis?
Yes
39
No
38
38
CHARACTER = open parenthesis?
No
CHARACTER = space (Δ)?
Yes
Store XS3 character for heading edit
Get next character
Advance parenthesis level by 1
No
37
No
CHARACTER = period?
No
36A
Yes
Warning #50
49
39
Get next character
CHARACTER = close parenthesis?
Yes
Edit title
40
No
Alarm #47
49

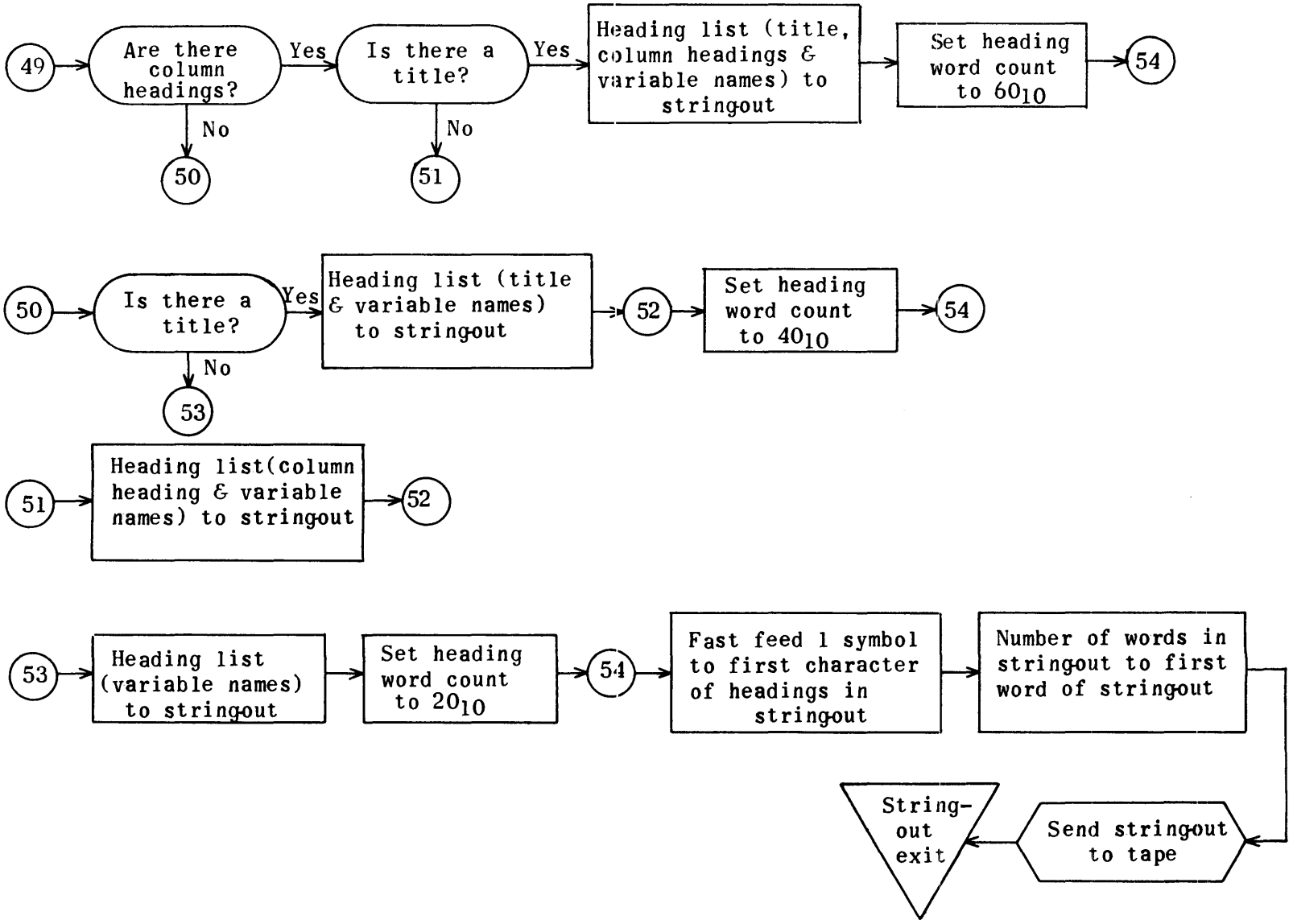

40
Get next character
41
Get next symbol
SYMBOL = comma or semi-colon?
No
SYMBOL = open parenthesis?
No
SYMBOL = space-period?
No
Yes
Yes
Yes
42
Zeroize assembly block
49
49
Warning #48
Yes
SYMBOL = close parenthesis?
No
Warning #49
42
Preset column heading section
42A
Advance column heading count
43
Next character to accumulator
44
Store XS3 character for heading edit
45
No
CHARACTER = space?
No
CHARACTER = open parenthesis?
No
CHARACTER = close parenthesis?
Yes
Yes
Yes
Get next character
44
48
45
Advance parenthesis level by 1
46

List Stringout (Heading Phase, Cont.)

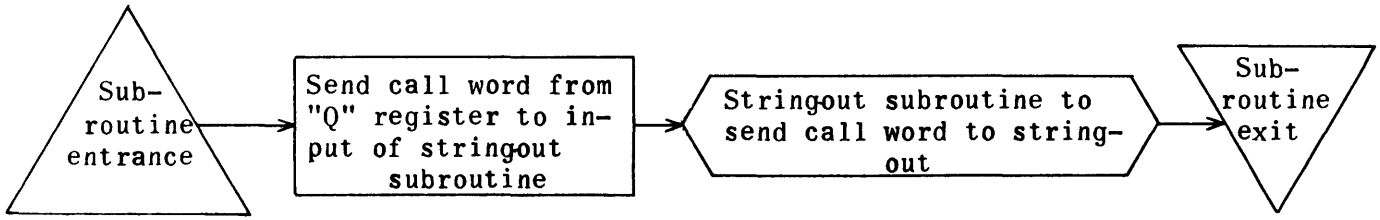




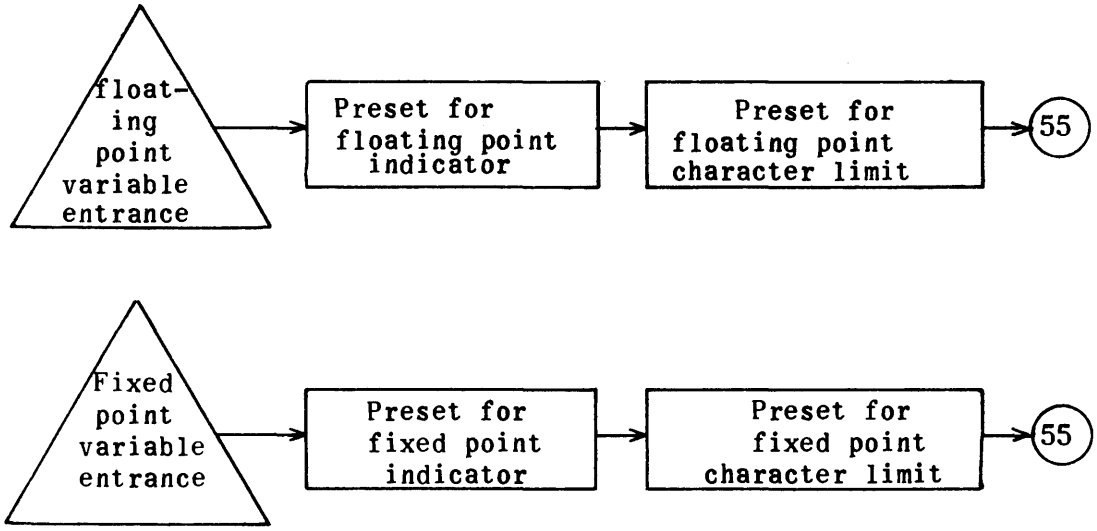
End List String-out

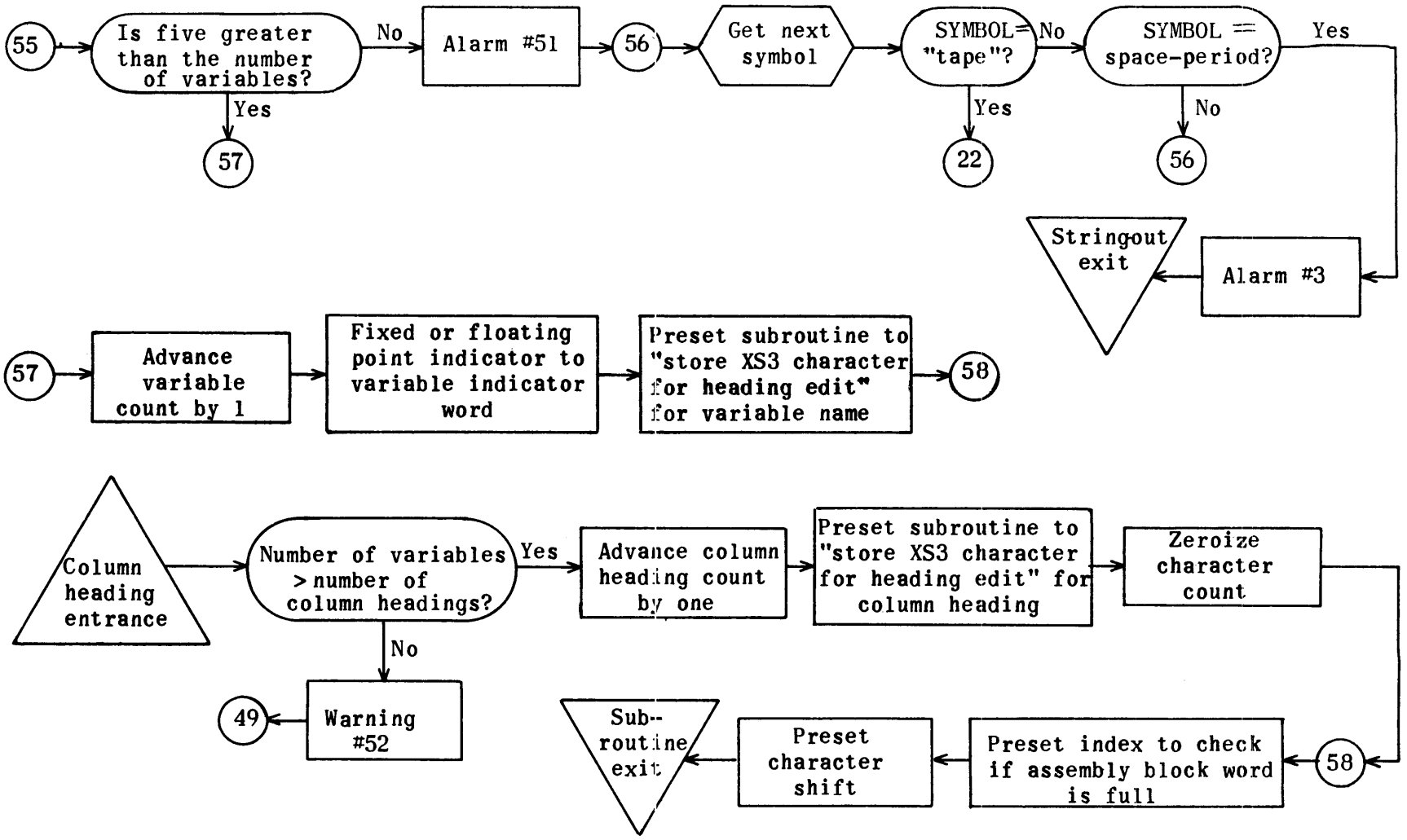


List Subroutine to Send Call Word to Stringout

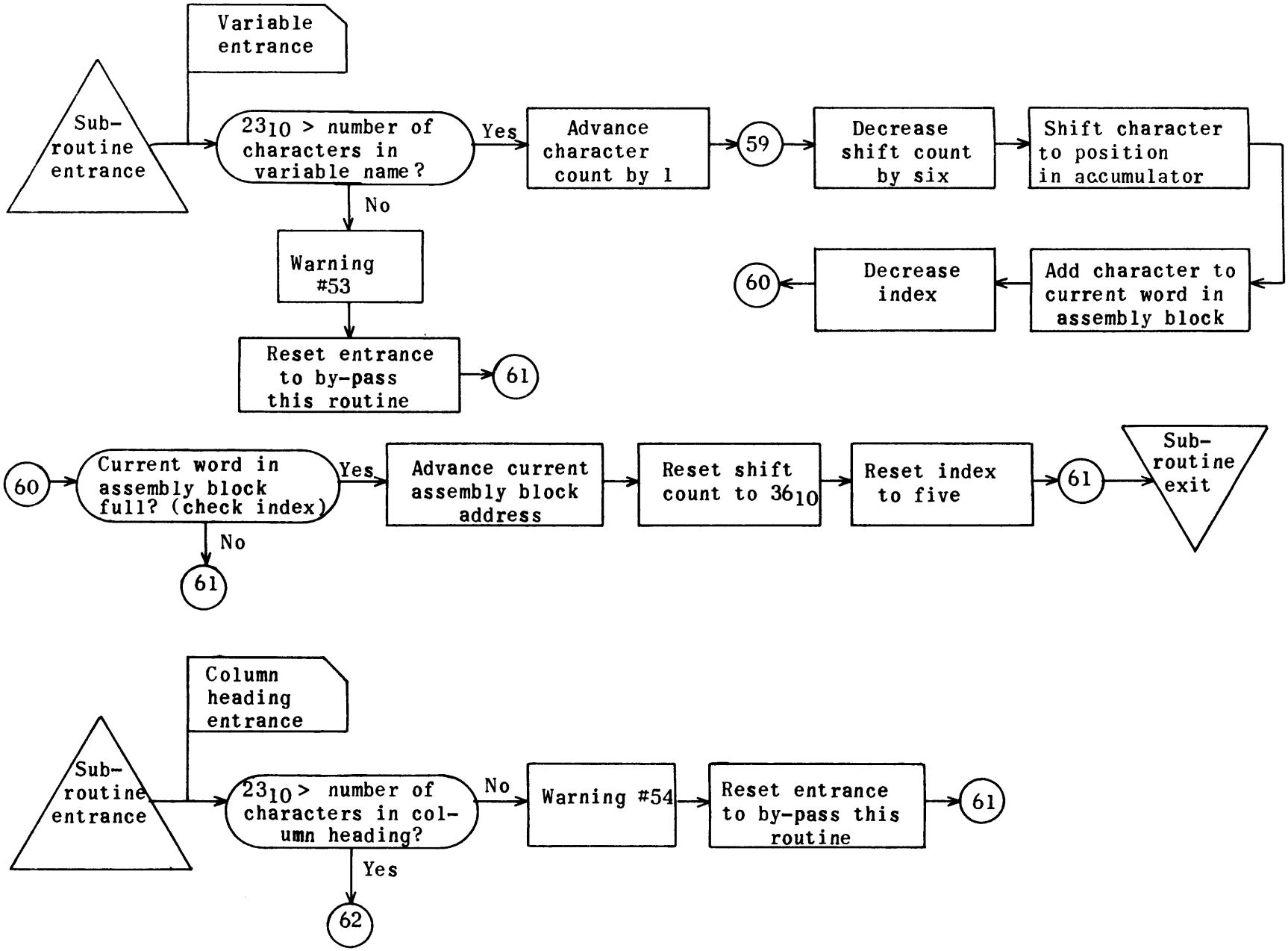


List Subroutine to Advance and Check Variable or Column Heading Count

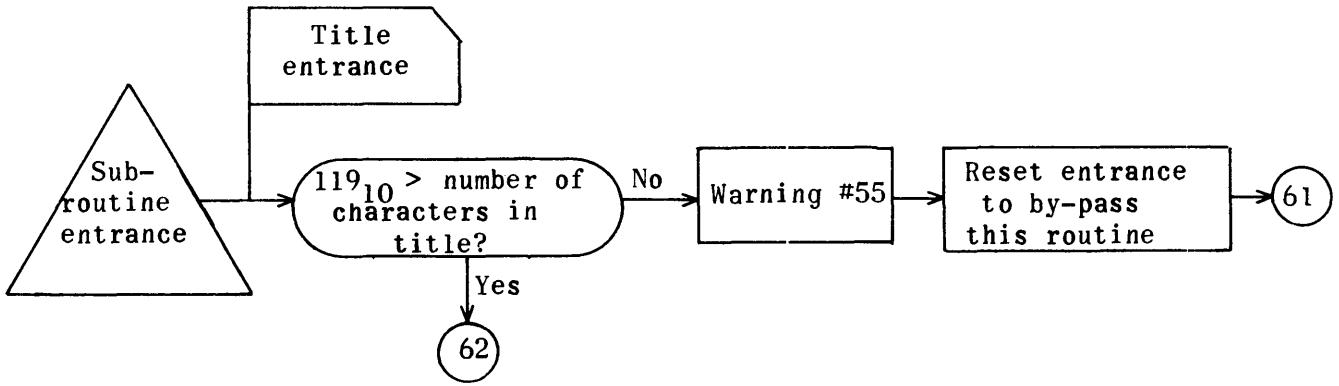


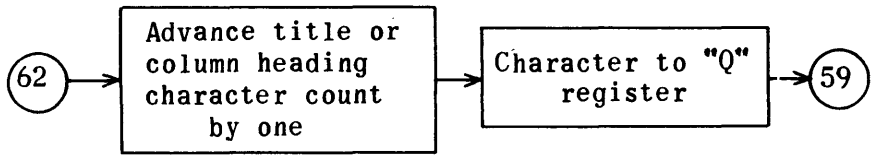

55
Is five greater than the number of variables?
No
Alarm #51
56
Get next symbol
SYMBOL = "tape"?
No
SYMBOL = space-period?
Yes
Yes
57
22
No
56
Stringout exit
Alarm #3
57
Advance variable count by 1
Fixed or floating point indicator to variable indicator word
Preset subroutine to "store XS3 character for heading edit" for variable name
58
Column heading entrance
Number of variables > number of column headings?
Yes
Advance column heading count by one
Preset subroutine to "store XS3 character for heading edit" for column heading
Zeroize character count
No
49
Warning #52
Sub-routine exit
Preset character shift
Preset index to check if assembly block word is full
58

List Subroutine to Store XS3 Character for Heading Edit

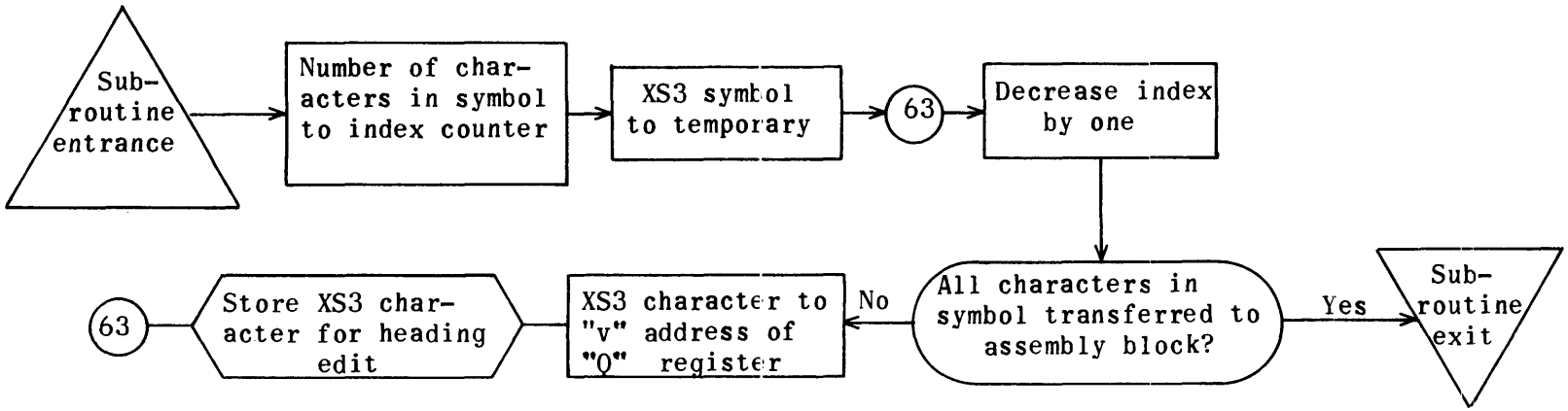


List Subroutine to Store XS3 Character for Heading Edit, Cont.

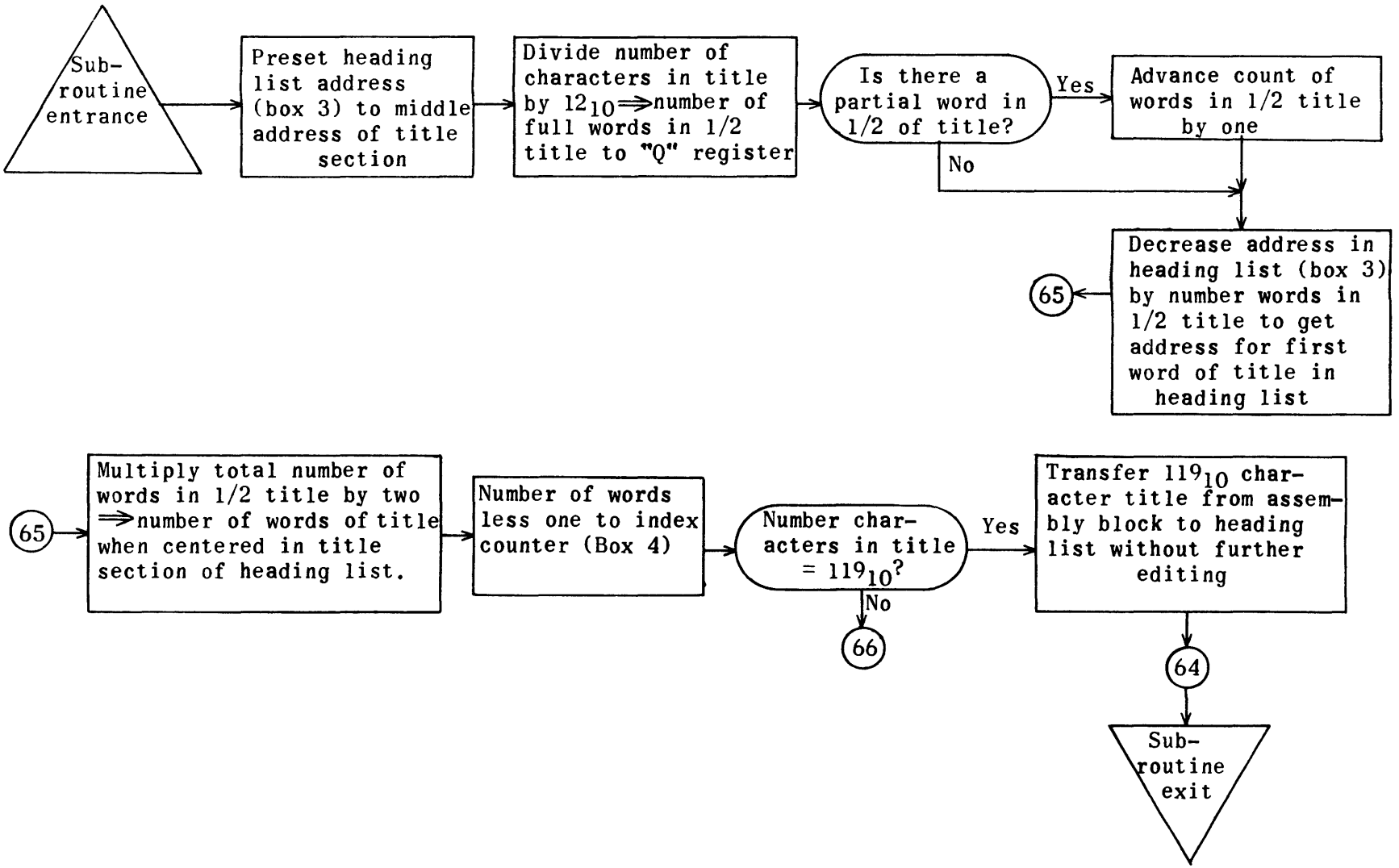


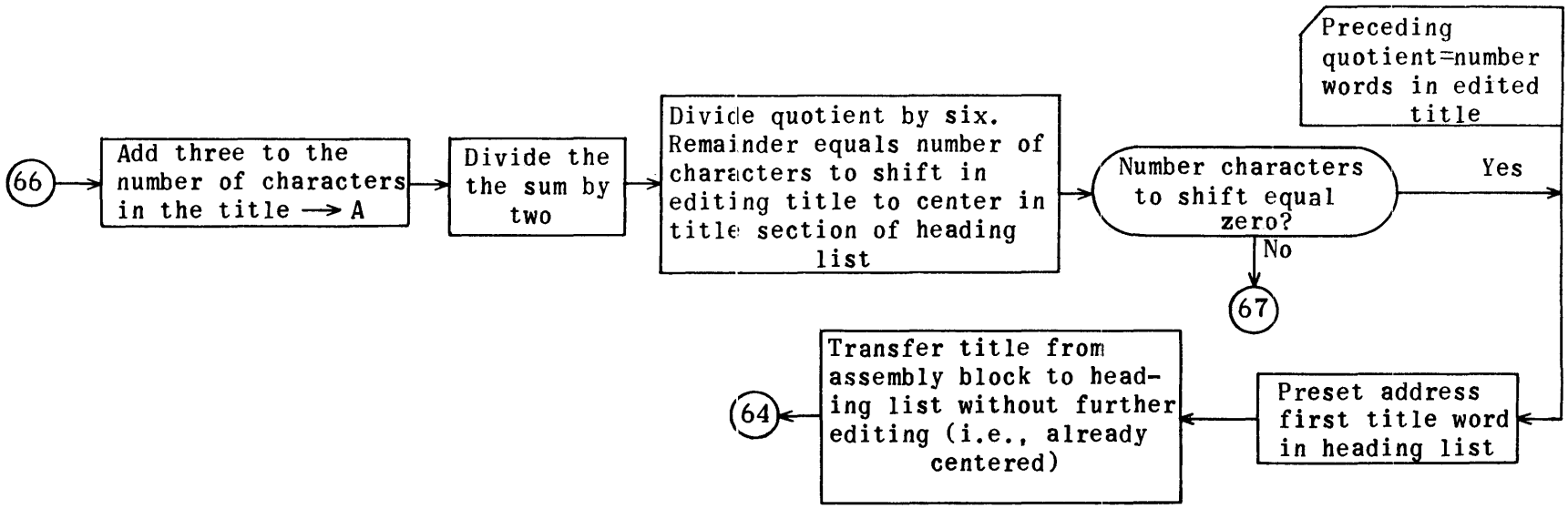


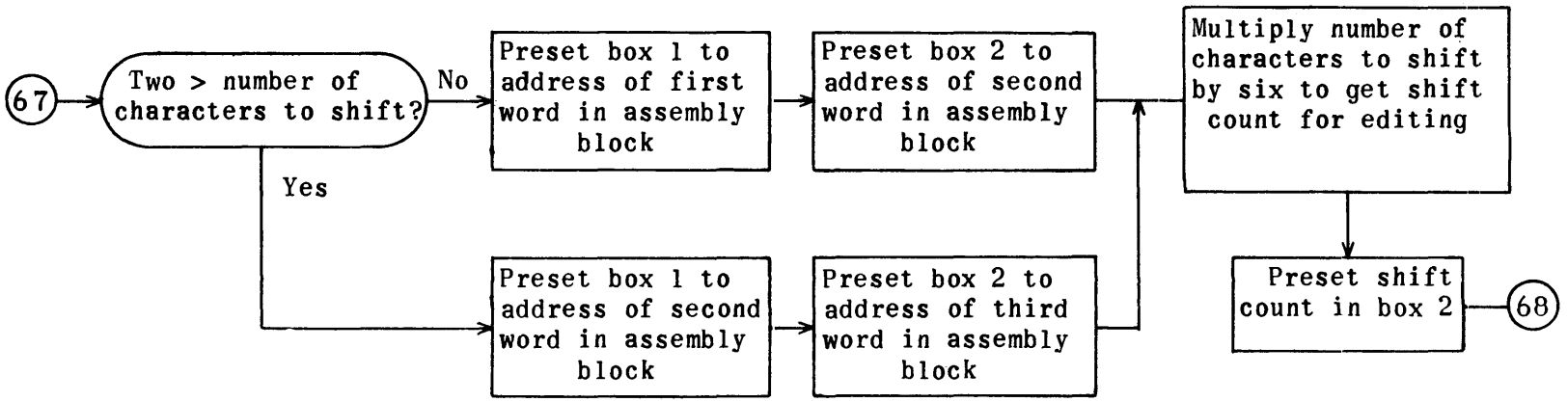
List Subroutine to Store XS3 Symbol for Heading Edit



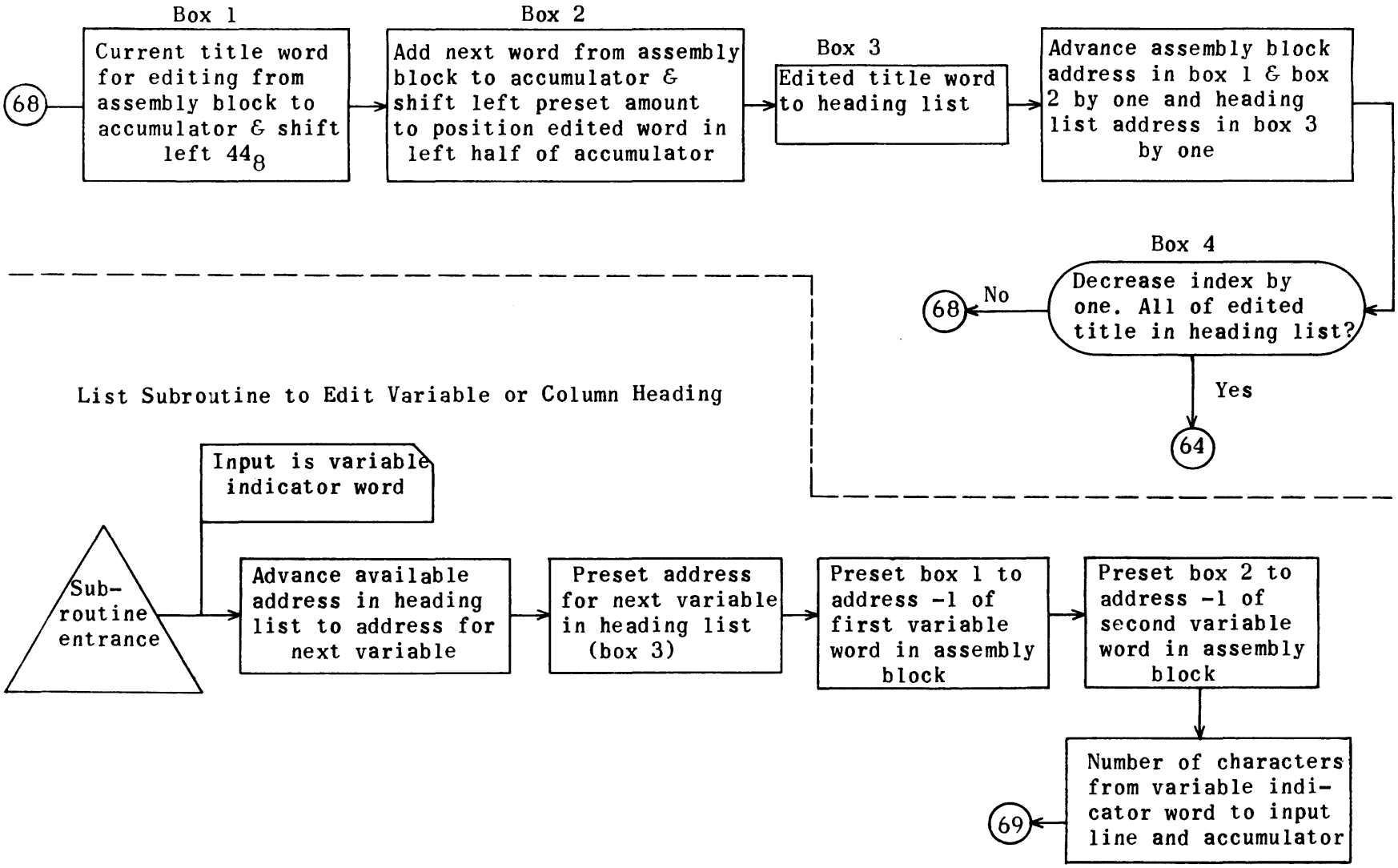
List Subroutine to Edit Title

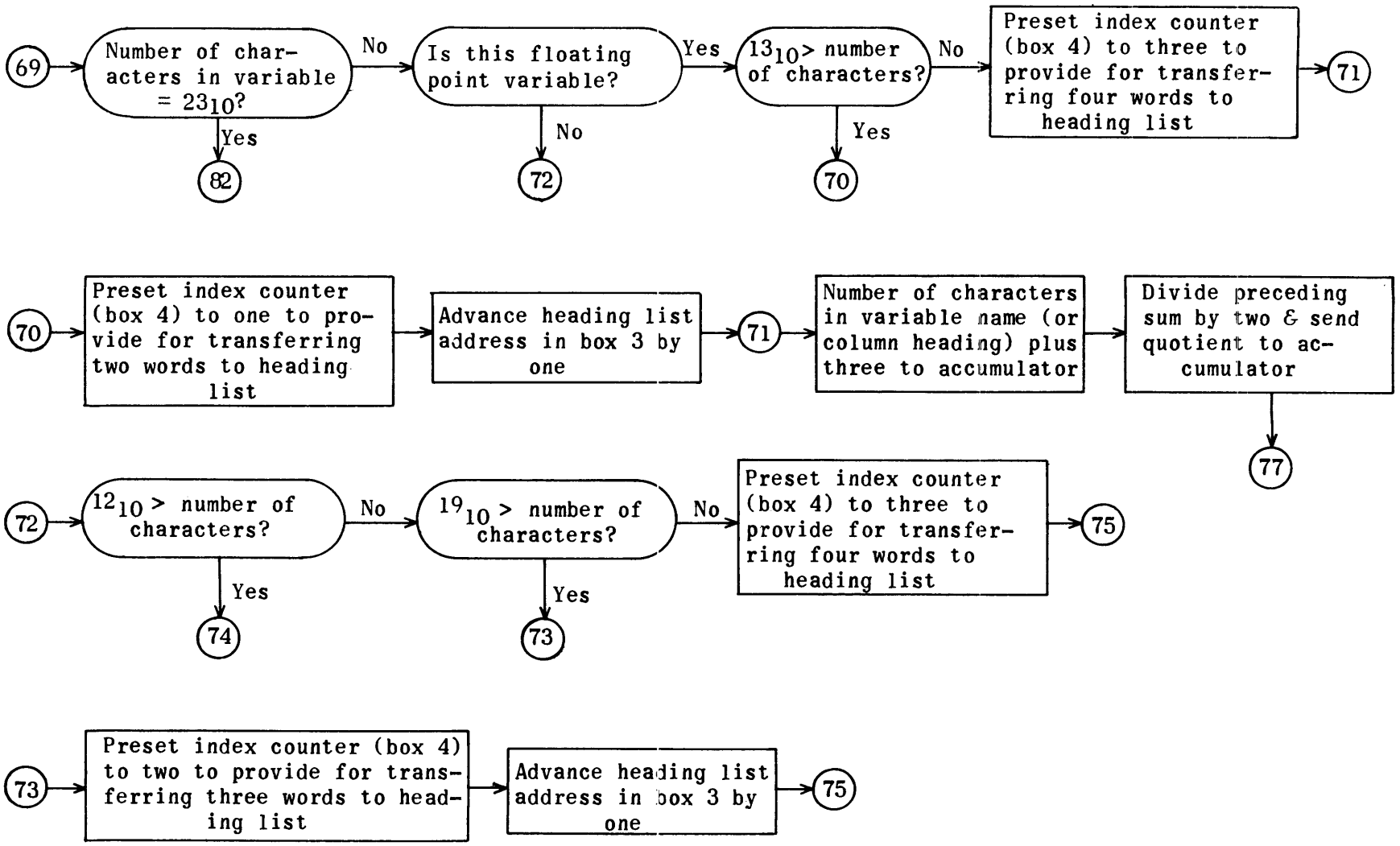



66
Add three to the number of characters in the title → A
Divide the the sum by two
Divide quotient by six. Remainder equals number of characters to shift in editing title to center in title section of heading list
Number characters to shift equal zero?
Yes
No
67
Preceding quotient=number words in edited title
Preset address first title word in heading list
Transfer title from assembly block to heading list without further editing (i.e., already centered)
64

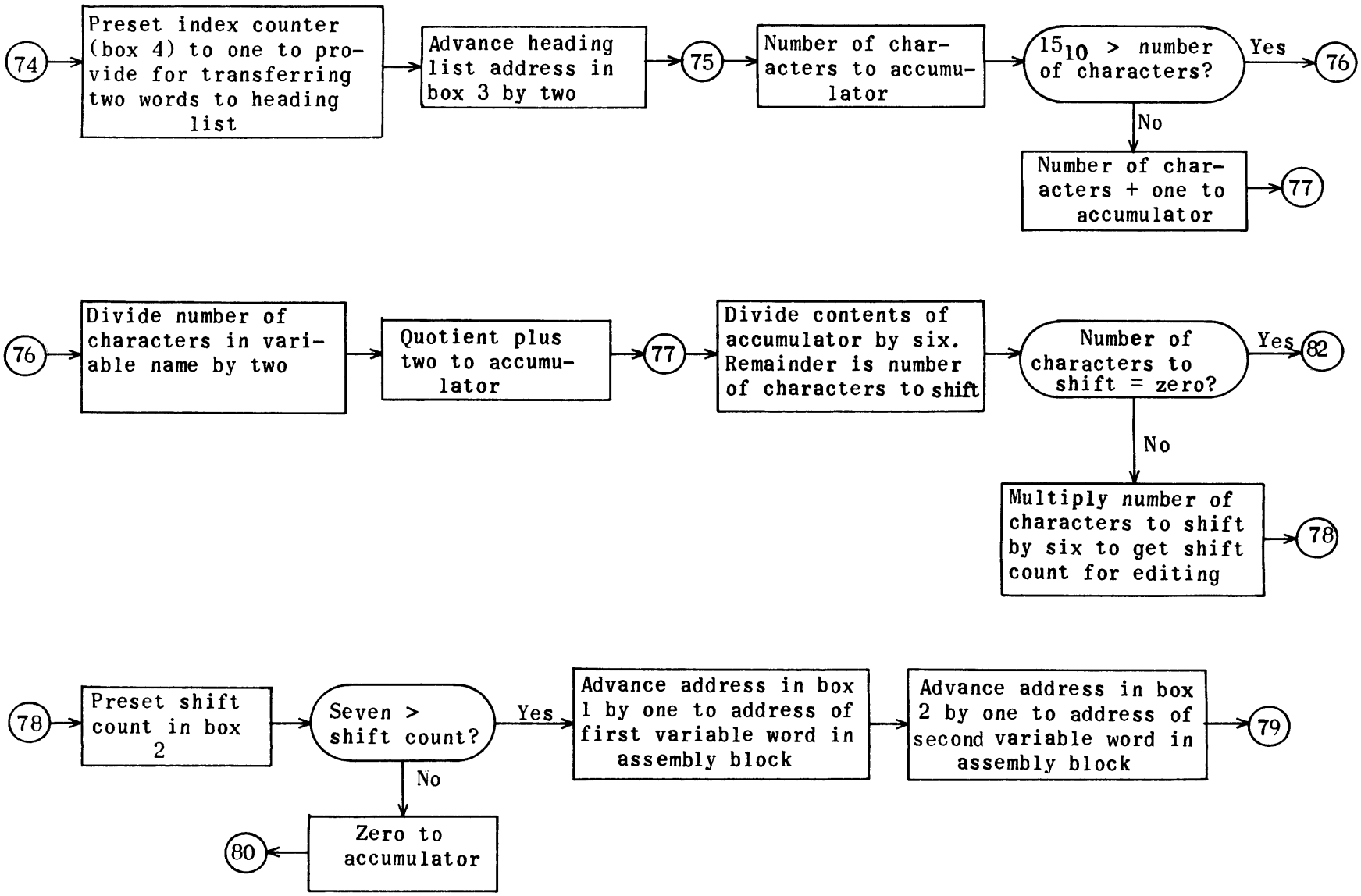

67
Two > number of characters to shift?
No
Yes
Preset box 1 to address of first word in assembly block
Preset box 2 to address of second word in assembly block
Preset box 1 to address of second word in assembly block
Preset box 2 to address of third word in assembly block
Multiply number of characters to shift by six to get shift count for editing
Preset shift count in box 2
68

List Subroutine to Edit Title, Cont.




69
Number of characters in variable = 23_{10}?
No
Is this floating point variable?
Yes
13_{10} > number of characters?
No
Preset index counter (box 4) to three to provide for transferring four words to heading list
71
Yes
82
No
72
Yes
70
70
Preset index counter (box 4) to one to provide for transferring two words to heading list
Advance heading list address in box 3 by one
71
Number of characters in variable name (or column heading) plus three to accumulator
Divide preceding sum by two & send quotient to accumulator
77
72
12_{10} > number of characters?
No
19_{10} > number of characters?
No
Preset index counter (box 4) to three to provide for transferring four words to heading list
75
Yes
74
Yes
73
73
Preset index counter (box 4) to two to provide for transferring three words to heading list
Advance heading list address in box 3 by one
75

List Subroutine to Edit Variable or Column Heading, Cont.



List Subroutine to Edit Variable or Column Heading, Cont.