Y-2013

A POSTPROCESSOR FOR AN AUTOMATIC PROGRAMMED TOOL (APT) PROCESSOR

T. L. Williams

August 1976

UNION CARBIDE

OAK RIDGE Y-12 PLANT
OAK RIDGE, TENNESSEE

prepared for the **U.S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION**
under **U.S. GOVERNMENT** *Contract W-7405 eng 26*

DISCLAIMER

Reference to a company or product name does not imply approval or recommendation of the product by Union Carbide Corporation or the U.S. Energy Research and Development Administration to the exclusion of others that may meet specifications.

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
Price: Printed Copy $5.50; Microfiche $2.25

Date of Issue: **August 24, 1976**
Distribution Category: **UC-32**

Report Number: **Y-2013**

A POSTPROCESSOR FOR AN AUTOMATIC PROGRAMMED TOOL (APT) PROCESSOR

T. L. Williams

Y-12 Laboratory Development Department
Y-12 Development Division

Oak Ridge Y-12 Plant
P. O. Box Y, Oak Ridge, Tennessee 37830

Prepared for the US Energy Research
and Development Administration
Under US Government Contract W-7405-eng-26

ABSTRACT

A postprocessor program was written for converting United Computing Corporation Automatic Programmed Tool (APT) cutter location data to a format compatible with the Ex-Cell-O 921/922 turning machines equipped with DDNC or a Bendix 1800 Control. The program is written in MACRO-11 and FORTRAN IV language for the PDP-11/10 minicomputer and is run under control of the computer's disk operating system software.

CONTENTS

SUMMARY

Automatic Programmed Tool (APT) processor software was purchased for use on a direct numerical control (DNC) system minicomputer. The computer system originally acted as a central part library and was only used when loading part programs in a controller's core memory. The APT processor was added and is used to generate cutter location (CL) data for manufacturing special production parts and is run during computer idle time. The APT-processor output consists of CL data and does not have the correct format for the DNC machine controller. A postprocessor program was written to convert these data to a format compatible with the direct digital numerical controller (DDNC) or the Bendix 1800 Controller used with Ex-Cell-O 921/922 turning machines. The postprocessor is written in minicomputer assembly language (MACRO-11) and in FORTRAN IV and is run under control of the computer's disk operating system.

INTRODUCTION

A DNC system, consisting of a central control computer (CCC) and several machine tool controllers,[a] was installed at the Oak Ridge Y-12 Plant.[b] The computer system (Figure 1) consists of a DEC[c] PDP-11/10 computer with 24 K words of core memory, a 1.25 megaword disk file, four 0.25 megaword magnetic tape units, a reader/punch, a line printer, a cathode ray tube (CRT) display, a teleprinter, and a card reader. The controllers were designed and manufactured in Y-12. Each one contains a magnetic core memory for program storage, a communication package for conversing with the CCC, and circuitry for handling two axes of motion on the Ex-Cell-O 921 and 922 turning machines. The controller can be expanded to five axes via minor modifications and used to control any machine tool requiring continuous-path motion.

The computer acts as a central part program library and is used only when new part programs are loaded into the core memory of the controller. An APT processor was purchased from United Computing Corporation. The processor accepts English-like statements that describe a part shown as an engineering drawing and converts these statements to CL data interspersed with machine-control commands such as those for coolant, spindle, and feedrate. These data are general in nature and cannot be directly used to control any particular machine. They must be further processed into a format compatible with the machine tool/controller combination on which the part is to be machined.

A postprocessor program was written to perform this task. The program is written using PDP-11 MACRO assembly language and FORTRAN IV and is run under control of the computer disk operating system software. The program accesses CL data temporarily stored on the disk by the APT processor and converts it to a format that is compatible with the Ex-Cell-O 921 and 922 turning machines equipped with a Bendix 1800 control unit or a DDNC.[a]

(a) Bowers, G. L., Lay, C. M., and Williams, T. L.; ***A Direct Digital Numerical Controller for Machine Tools***, Y-1966; Union Carbide Corporation-Nuclear Division, Oak Ridge, Tennessee; January, 1975.

(b) Operated by Union Carbide Corporation's Nuclear Division for the US Energy Research and Development Administration.

(c) A Digital Equipment Corporation trademark.

155352

Figure 1. CENTRAL CONTROL COMPUTER FOR DIRECT NUMERICAL CONTROL.

THE POSTPROCESSOR

INTRODUCTION

An APT part program example is presented in Figure 2. When this information is processed, two forms of output result: (1) a CL listing (Figure 3) is generated at the line printer for program verification, and (2) a file is created on the disk (disk CL file) that contains these data along with other commands that must be processed further before the data are in a form that can be used to control a machine tool. An example of a program with a representation of the disk file is given in Figure 4. The first column of numbers is the APT statement number; the second column is the address of the disk area where data are stored; the remaining data are an octal representation of the information as it is stored on the disk and includes all the CL data along with such machine control commands as feedrates, coolant commands, spindle controls, and sequence numbers. The postprocessor massages these data, converting them to a format that is compatible with the Bendix 1800 NC or DDNC interfaced to an Ex-Cell-O 922 or 921 turning machine. Figure 5 outlines the different processes and paths that a part program takes before it can be used to machine a part.

POSTPROCESSOR REQUIREMENTS

The postprocessor's basic function is to convert the CL data to properly formatted incremental movements with feed numbers and spindle speeds along with other coded control parameters. CL data are actually cartesian coordinates of the end point of each tool movement. Incremental movements are calculated by subtracting successive CL data points and rounding off to the nearest machine pulse value [20×10^{-6} in (0.5 μm)]. Errors induced by this operation are always tagged onto the next machine movement. The required data format for an incremental movement assumes that a decimal point exists five places to the left of the least significant digit and that the sign is positive unless otherwise indicated, thus:

X	Y	Feed Number	Spindle Code
X-1234568	Y-1234568	F12345	S12

The feed number can contain from one to five digits and is calculated as the inverse of time required to travel the maximum incremental movement multiplied by 10:

$$\text{Time (s)} = \frac{\text{Maximum Incremental Movement}}{\text{Feedrate (in/min)} \times 1/60} \text{ (in);}$$

$$\text{Feed Number} = 10\left(\frac{1}{\text{Time}}\right) = \frac{\text{Feedrate} \times 600}{\text{Maximum Incremental Movement}}.$$

A spindle code is derived from calculated RPM (revolutions per minute) values via a curve-fitting technique (further details are given in Appendix A). An example of a typical postprocessor output is given in Figure 6. Other data are interspersed with the incremental movements (feedrates and spindle speeds). Table 1 contains a list of these data along with a brief explanation of their effect on the postprocessor.

```
PARTNO TEST FOR PT DEFINED SURF COMPATIBILITY -ELLIPSE-**UNCLASSIFIED** 9201-5W
CHARGE=1863
        RESERV/P,46,L,46,LN,46,PN,25                                    0104
        NOPOST                                                          0105
        INTOL/.00001                                                    0107
        OUTTOL/.00001                                                   0109
        CUTTER/.030                                                     0111
        CLPRNT                                                          0113
PO      =POINT/0,0,0                                                    0115
SETPT   =POINT/-1.,-1.,0                                                0117
LX      =LINE/PO,ATANGL,0                                               0119
LY      =LINE/PO,ATANGL,90                                              0121
PN(1)   =POINT/RTHETA,XYPLAN,4.000000,0                                  01
PN(2)   =POINT/RTHETA,XYPLAN,3.992452,4                                  03
PN(3)   =POINT/RTHETA,XYPLAN,3.970206,8                                  05
PN(4)   =POINT/RTHETA,XYPLAN,3.934407,12                                 07
PN(5)   =POINT/RTHETA,XYPLAN,3.886808,16                                 09
PN(6)   =POINT/RTHETA,XYPLAN,3.829579,20                                 11
PN(7)   =POINT/RTHETA,XYPLAN,3.765098,24                                 13
PN(8)   =POINT/RTHETA,XYPLAN,3.695751,28                                 15
PN(9)   =POINT/RTHETA,XYPLAN,3.623790,32                                 17
PN(10)  =POINT/RTHETA,XYPLAN,3.551222,36                                 19
PN(11)  =POINT/RTHETA,XYPLAN,3.479762,40                                 21
PN(12)  =POINT/RTHETA,XYPLAN,3.410818,44                                 23
PN(13)  =POINT/RTHETA,XYPLAN,3.345308,48                                 01
PN(14)  =POINT/RTHETA,XYPLAN,3.284686,52                                 03
PN(15)  =POINT/RTHETA,XYPLAN,3.228990,56                                 05
PN(16)  =POINT/RTHETA,XYPLAN,3.178878,60                                 07
PN(17)  =POINT/RTHETA,XYPLAN,3.134665,64                                 09
PN(18)  =POINT/RTHETA,XYPLAN,3.096561,68                                 11
PN(19)  =POINT/RTHETA,XYPLAN,3.064701,72                                 13
PN(20)  =POINT/RTHETA,XYPLAN,3.039162,76                                 15
PN(21)  =POINT/RTHETA,XYPLAN,3.019986,80                                 17
PN(22)  =POINT/RTHETA,XYPLAN,3.007196,84                                 19
PN(23)  =POINT/RTHETA,XYPLAN,3.000800,88                                 21
PN(24)  =POINT/RTHETA,XYPLAN,3.000000,90                                 23
PN(25)  =POINT/RTHETA,XYPLAN,3.000000,90                                 24
T1      =TABCYL/THETAR,SPLINE,$                                         0123
        0,4,4.3.992452,8,3.970206,12,3.934407,16,3.886808,20,3.829579,$  0125
        24,3.765098,28,3.695751,32,3.623790,36,3.551222,40,3.479762,$    0201
        44,3.410818,48,3.345308,52,3.284686,56,3.228990,60,3.178878,$    0203
        64,3.134665,68,3.096561,72,3.064701,76,3.039162,80,3.019986,$    0205
        84,3.007196,88,3.000800,90,3,SLOPE,0                            0207
PT      =POINT/4.0,0                                                    0209
L(1)    =LINE/PO,ATANGL,0                                               0211
P(1)    =POINT/INTOF,L(1),T1,PT                                         0213
LN(1)   =LINE/P(1),ATANGL,0                                             0215
        K=-2                                                            0217
        I=1                                                             0219
        LOOPST                                                          0221
10)     I=I+1                                                           0223
        K=K+4                                                           0225
        L(I)=LINE/PO,ATANGL,K                                           0301
        P(I)=POINT/INTOF,L(I),T1,PN(I)                                  0303
        LN(I)=LINE/P(I),PERPTO,T1,PN(I)                                 0305
        IF(K-88)10,11,11                                                0307
11)     LOOPND                                                          0309
        PRINT/3,ALL                                                     0311
        FROM/SETPT                                                      0313
        INDIRV/0,1,0                                                    0315
        GO/TO,T1                                                        0317
        TLRGT,GORGT/T1,TO,LY                                            0319
        I=25                                                            0321
        LOOPST                                                          0323
13)     I=I-1                                                           0325
        GOFWD/T1,ON,LN(I)                                               0401
        IF(I-1)14,14,13                                                 0403
14)     LOOPND                                                          0405
        TLON,GORGT/L(I),PAST,LY                                         0407
        GOTO/SETPT                                                      0409
        FINI                                                            0411
```

Figure 2. EXAMPLE OF A PART PROGRAM.

POSTPROCESSOR OPERATION

Postprocessor operation is initiated via the disk operating system (DOS) RUN POS922 command. This system loads the postprocessor program (stored on disk) in the computer and initiates its operation. The input to the postprocessor is from a United Programming Language (UPL) file that was created when APT processed the part program in question. When this file is located by the postprocessor, processing begins.

The disk CL file is separated into records of which there are five types: FINI, postprocessor commands, cutter centerline data, circle canonical form, and multax cutter centerline data. (Refer to Appendix B for further details on the disk's CL format.) Each record is sequentially processed by the postprocessor. Appropriate action is taken according to the record type. (A detailed analysis of the postprocessor is given in Appendix A.)

Type 0 Record (FINI)

This record only occurs one time. It is generated by an APT FINI statement that signifies that the end of the program is at hand. When it is encountered, the disk file being generated by the postprocessor is closed and all final software housekeeping is performed before program control returns to the DOS.

Type 1 Record (postprocessor commands)

These commands are listed in Table 1. When a Type 1 record is encountered, a software polling is made for each of these commands until one is found.

```
REC0056
    -1.000000    -1.000000     .000000
REC0057
    -1.000000    -1.000000     3.500000
REC0060
     3.000000     4.000000     3.500000
REC0062
     4.149999     4.850000     2.750000
     3.850000     4.850000     2.750000
     3.725000     4.725000     2.750000
     4.274999     4.725000     2.750000
     4.274999     4.933891     2.750000
     4.000000     5.071391     2.750000
     3.725000     4.933891     2.750000
     3.725000     4.725000     2.750000
     3.475000     4.475000     2.750000
     4.524999     4.475000     2.750000
     4.524999     5.038399     2.750000
     4.000000     5.350900     2.750000
     3.475000     5.088399     2.750000
     3.475000     4.475000     2.750000
     3.225000     4.225000     2.750000
     4.774999     4.225000     2.750000
     4.774999     5.242908     2.750000
     4.000000     5.630408     2.750000
     3.225000     5.242908     2.750000
     3.225000     4.225000     2.750000
     2.975000     3.975000     2.750000
     5.024999     3.975000     2.750000
     5.024999     5.397416     2.750000
     4.000000     5.909916     2.750000
     2.975000     5.397416     2.750000
     2.975000     3.975000     2.750000
     2.725000     3.725000     2.750000
     5.274999     3.725000     2.750000
     5.274999     5.551925     2.750000
     4.000000     6.189424     2.750000
     2.725000     5.551925     2.750000
     2.725000     3.725000     2.750000
     2.475000     3.475000     2.750000
     5.524999     3.475000     2.750000
     5.524999     5.706434     2.750000
     4.000000     6.468933     2.750000
     2.475000     5.706434     2.750000
     2.475000     3.475000     2.750000
     2.225000     3.225000     2.750000
     5.774999     3.225000     2.750000
     5.774999     5.860942     2.750000
     4.000000     6.748441     2.750000
     2.225000     5.860942     2.750000
     2.225000     3.225000     2.750000
     2.125000     3.125000     2.750000
     5.875000     3.125000     2.750000
     5.875000     5.922746     2.750000
     4.000000     6.860246     2.750000
     2.124999     5.922746     2.750000
     2.125000     3.125000     2.750000
REC0064
     2.125000     3.125000     6.250000
REC0065
    -3.875000     3.125000     6.250000
REC0066
    -1.000000    -1.000000     .000000
```

Figure 3. EXAMPLE OF A CUTTER LOCATION LISTING.

```
      UNITED COMPUTING CORPORATION * UNIAPT  * V4 R13 03/15/73
      0001   MACHIN/41
      0002   PARTNO TEST CASE FOR NULL
      0003   GOTO/1,2,3
      0004   FEDRAT/50
      0005   STOP
      0006   END
      0007   FINI
      END PAS1   00 ERRORS.

0001 = 00600000  0000  0005  0001  0001  0033  1767  0002  0000  0006  2440  0000  0000
0002 = 00600006  0000  0044  0001  0002  0033  2025  4015  0007  0240  0324  0305  0323  0324  0240  0303  0301
       00600016  0323  0305  0240  0306  0317  0322  0240  0316  0325  0314  0314  0240  0240  0240  0240  0240
       00600026  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240
       00600036  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240
       00600046  0240  0240  0240  0240  0240  0240  0240  0240  0240  0240
0003 = 00600053  0000  0007  0002  0003  0001  2000  0000  0000  0002  2000  0000  0000  0002  3000  0000  0000
0004 = 00600063  0000  0005  0001  0004  0033  1761  0002  0000  0006  3100  0000  0000
0005 = 00600071  0000  0002  0001  0005  0033  0002
0006 = 00600074  0000  0002  0001  0006  0033  0001
0007 = 00600077  0000  0001  0000  0007
```

0001 Length=5; Type 1 (one physical record), Logical Record 1; Subtype 3, Vocabulary Word Class 1, Vocabulary Word Subclass 1767; Subtype 2; Floating 41.

0002 Length=44; Type 1 (one physical record), Logical Record 2; Subtype 3, Vocabulary Word Class 1, Vocabulary Word Subclass 2025; Subtype 1; Character String 33 words long.

0003 Length=5; Type 2, Logical Record 3; Floating 1, Floating 2, Floating 3.

0004 Length=5; Type 1 (one physical record), Logical Record 4; Subtype 3, Vocabulary Word Class 1, Vocabulary Word Subclass 1761; Subtype 2; Floating 50.

0005 Length=2; Type 1 (one physical record), Logical Record 5; Subtype 3, Vocabulary Word Class 1, Vocabulary Word Subclass 2.

0006 Length=2; Type 1 (one physical record), Logical Record 6; Subtype 3, Vocabulary Word Class 1, Vocabulary Word Subclass 1.

0007 Length=1; Type 0; Logical Record 6.

Floating point numbers take two UPL words or four DEC words to represent.

; (semicolon) represent end of a word
. (period) end of a record

Figure 4. EXAMPLE PROGRAM WITH AN OCTAL DUMP OF A DISK FILE.

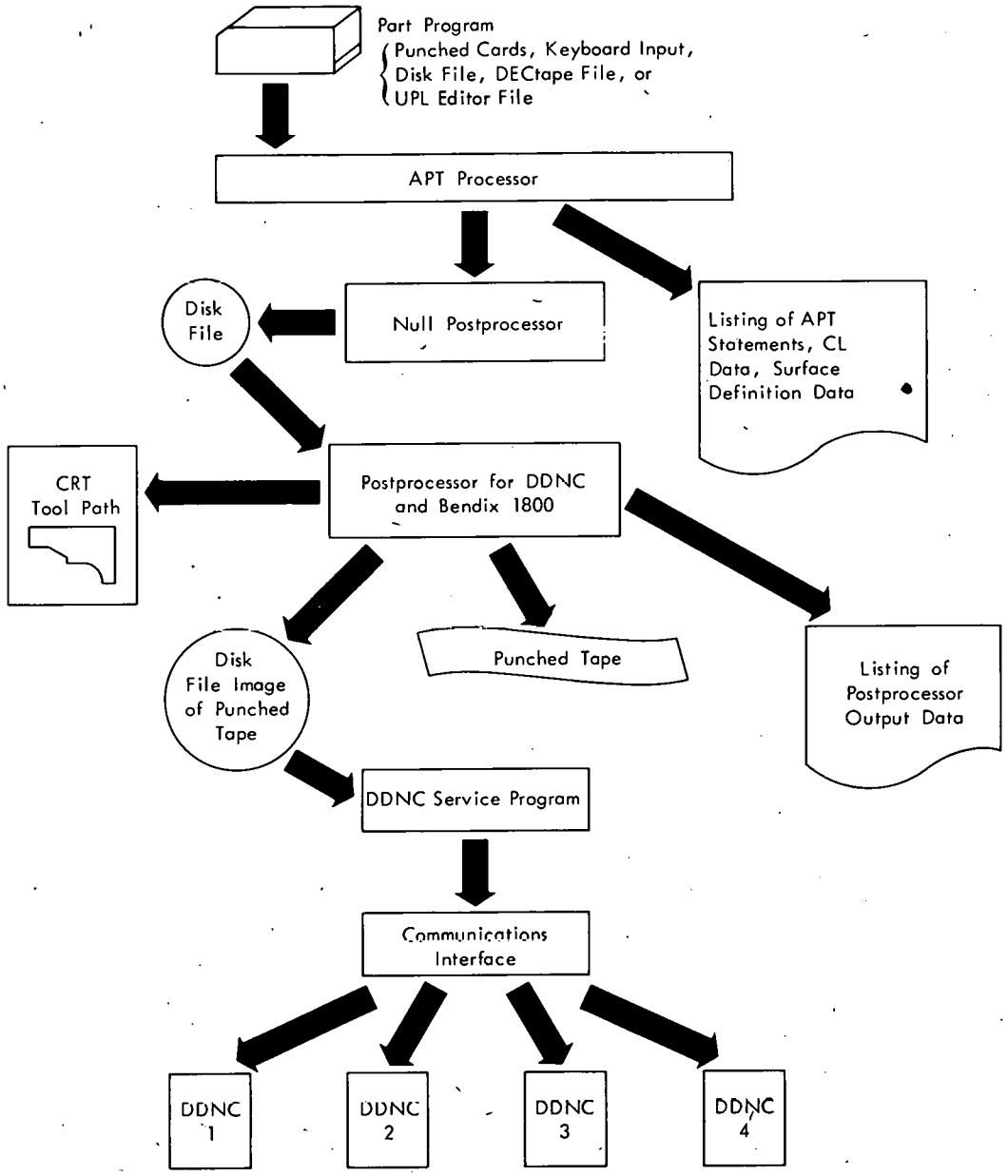


Figure 5. PART PROGRAM GENERATION.

Further processor action is determined by the type of command that is being processed. (Refer to Appendix A for further details.) For example, when the APT statement "SPINDL" is encountered, the software must then look for: (1) a following floating point number, (2) is this number RPM or SFM (surface feet per minute), (3) is there a gear range, (4) if so what is it, and (5) is the direction of the spindle clockwise or counterclockwise. Each of these causes the processor to perform a different function.

```
000053  000055  000415   ************** CONTACT TYPE TOOL SET ***************
000170  00000000000000000000000000000000000000000000000000000000000000
000174  N021
000175  F6000
000176  00000000000000000000000000000000000000000000000000000000000000
000417  M18
000432  000433  X-10000F1000
000434  000435  Y-62100F160
000436  X62100F160
000437  000440  Y10000F1000
000441  M00
000442  Y-10000F1000
000443  000444  X-62100F160
000445  Y62100F160
000446  000447  X10000F1000
000450  M00
000451  X-21900F456
000454  000456  000463  Y-100000F300
000467  Y-464450F64
000470  X-926000F32
000471  M00
000556   *****  END OF ATLSET MACRO  *****
000563   ***** STRTUP MACRO *****
000570  00000000000000000000000000000000000000000000000000000000000000
000571  N070
000572  00000000000000000000000000000000000000000000000000000000000000
000573  000576  000577  M17
000603  S60
M04
000613  F6000
000614   ***** END OF STRTUP MACRO *****
000746  Y137350F218861
000747  X-77900F385
000750  000751  000752  M08
000754  X-64696F51576
X-38244F95
000756  Y12650F288
XXXXXXPP7XXXXXX 000035
000757  X-2Y1828F1994
000757  X-32Y3658F996
000757  X-64Y3658F996
000757  X-96Y3656F997
000757  X-128Y3656F997
000757  X-158Y3654F998
000757  X-192Y3654F999
000757  X-224Y3650F1000
000757  X-254Y3650F1001
000757  X-288Y3648F1003
000757  X-318Y3644F1004
000757  X-352Y3642F1005
000757  X-382Y3636F1008
000757  X-414Y3636F1009
000757  X-446Y3630F1012
000757  X-476Y3626F1014
```

Figure 6. A TYPICAL POSTPROCESSOR OUTPUT.

Type 2 Record (cutter centerline data)

These data signal the postprocessor that this is either an APT "FROM" statement or tool movement statement. The "FROM" statement is an initialization command. When it is encountered, the cartesian coordinates contained in this record are stored as if they are the position from which the tool will start. Tool-movement records require more extensive

Table 1

POSTPROCESSOR COMMANDS

	Postprocessor Command	Code for Controller	Function
(1)	AUXFUN/17	M17	Insert selected offset.
(2)	AUXFUN/18	M18	Remove selected offset.
(3)	COOLNT/ON	M08	Turn coolant on.
(4)	COOLNT/OFF	M09	Turn coolant off.
(5)	DELAY/s, REV	Fnnnnn	Causes a controller dwell.
(6)	DISPLY/ON		Enables the postprocessor for drawing the tool path on a cathode ray tube (CRT).
(7)	DISPLY/OFF		Erases the CRT and disables the postprocessor output to the CRT.
(8)	FEDRAT/{IPM, IPR}, f, MAXIPM		Inputs f in ipm or ipr along with maximum ipm to the postprocessor.
(9)	LEADER/n	Generates n x 10 EIA 0s	Inserts n inches of leader on tape and stores n x 10 zero on the disk file.
(10)	MACHIN/XX, i, j, k, l		Postprocessor medium control (see Appendix B).
(11)	MCHTOL/n		Postprocessor control parameter (machine tolerance).
(12)	OPSTOP	M01	Causes a stop at the controller if optional stop enabled.
(13)	PARTNO/--66 Characters---		Used for part identification (see Appendix B).
(14)	PPRINT		Indicates that the following comment is to be printed.
(15)	SEQNO/{ON, OFF}		Enables or disables postprocessor generation of sequence numbers.
	SEQNO/iii, INCR j, k		Sequence number control parameters (see Appendix B).
(16)	SPINDL/iii, {RPM, SFM}, RANGE, {HIGH, MEDIUM, LOW, n}, {CLW, CCLW}	{M03 CLW, M04 CCLW, M05 OFF, SXX Spindle speed}	Spindle control parameters (see Appendix B).
(17)	STOP	M00	Causes an unconditional stop at the controller.
(18)	RAPID		Postprocessor control. Forces maximum feedrate.
(19)	REWIND	M30	Causes a memory address reset or tape rewind.
(20)	END	M02	Notifies controller of the end of part program.

processing (see Appendix A). The cartesian coordinate information within the record is first stored as this tool movement's end point, and calculations are made to determine incremental movement of each axis, feed number, and spindle speed (see Appendix A). The movement is then plotted on a CRT, and properly formatted data are outputted to a previously defined disk file, line printer for program verification, and punched tape when required.

Types 3 and 4 Records (circle, canonical form, and multax)

These data types are used for multiaxes controllers and controllers with circular interpolators and are not used by this postprocessor.

SYSTEM OPERATION

System Hardware

Figure 7 is a block diagram of the DNC hardware system. This system is considered to be of minimal size for efficient operation.

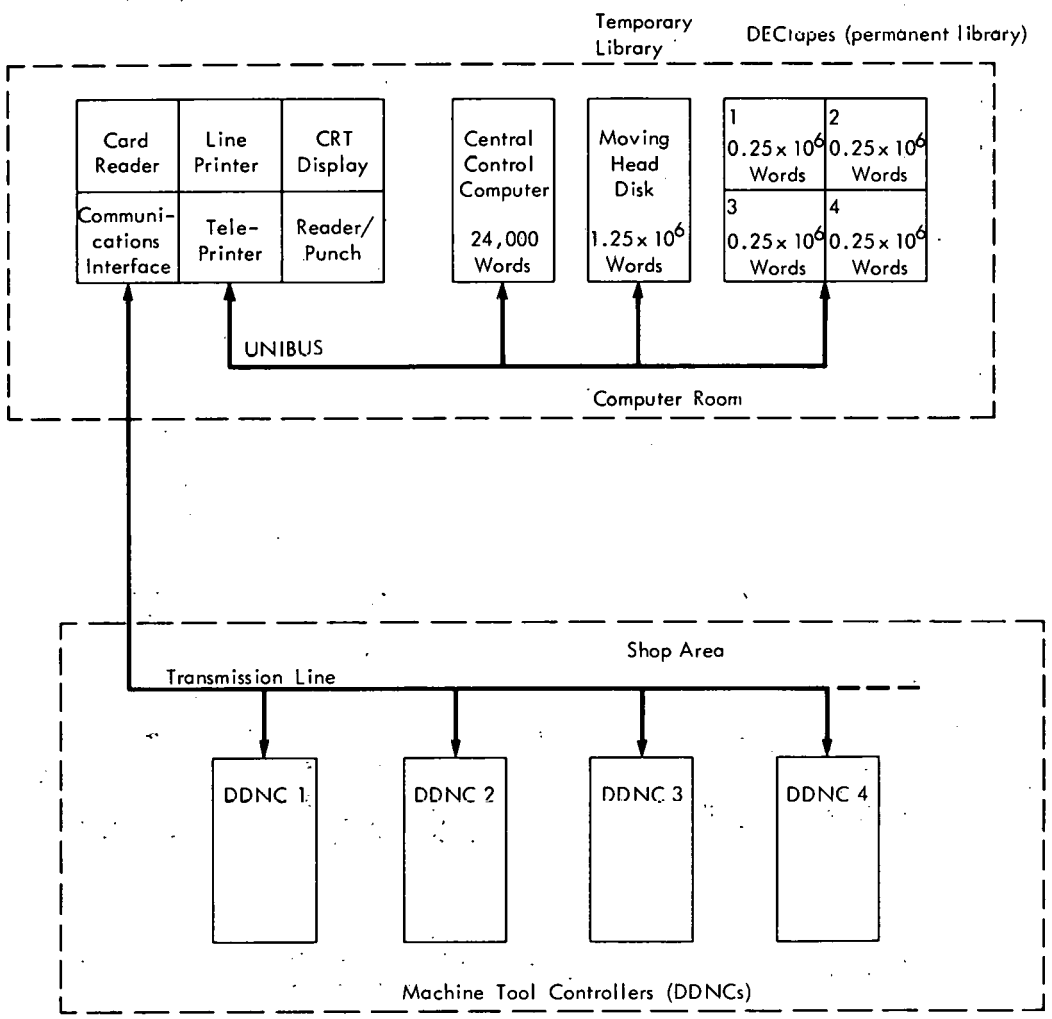


Figure 7. DIRECT NUMERICAL CONTROL HARDWARE SYSTEM.

Central Control Computer - The computer is a DEC PDP-11/10 with 24,000 sixteen-bit words of core memory, read-only-memory (ROM) boot loaders for DEC tape and disk, electronic extended arithmetic element (EAE), auto power fail, and real time clock.

Moving Head Disk - The disk is a 1.25-million-word device with removable cartridge. It is used for permanent storage of system software and temporary storage for APT source programs and postprocessor object (part programs) storage.

Magnetic Tape Units - Four 0.256×10^6 word DECtapes are used for permanent storage of part programs and for creation of permanent records of APT source programs.

Card Reader - A 300-card-per-minute unit is used for the input of APT source card decks. It is generally used for input of very long source programs.

Lineprinter - The lineprinter is generally used for APT CL listings, postprocessor listings, and printout of sequence numbers with their corresponding memory address in the DDNC's core memory. The printer is a 300-character-per-second, 132-character-line-length, dual-head matrix device.

CRT Display - A 16.2 by 21-cm CRT is used to display the tool path as it is being generated by the postprocessor. It is controlled by an in-house designed interface through software interpolation of tool path data.

Reader/Punch - These two devices are generally used for system maintenance, input of EIA part descriptionss for conversion to ASCII and storage in the program library, and creation of EIA punched tapes for standard NC controllers. The device reads at 300 characters per second and punches at 50 characters per second.

Teleprinter - The teleprinter serves as an input/output terminal between the user and computer. It may also be used in place of the line printer for CL and postprocessor listings if printer failures occur.

Communications Interface - This electronic device serves as a communication link between the computer and DDNCs which are connected in party-line arrangement. The interface receives serial request data from the DDNCs, stacks these data, transfers them to the computer, serializes the part program data, and transmits these data to the requesting DDNC.

Direct Digital Numerical Controllers (DDNCs) - The controller consists of a memory for part program storage, a communications package, and a controller designed around digital differential analyzers. More detailed information on DDNCs may be found in the report by Bowers, et al.[a]

System Software - There are two separate software operating packages. A more detailed description of their operation can be found in Appendix C.

DEC Disk Operating System[d] - This software supports the PDP-11 user to provide convenient access to such system programs as FORTRAN, MACRO assembler, debugger,

---

(d) DEC disk operating system which is called DOS and the United disk system called UDS.

editor, and file utility package. Keyboard commands enable the operator to load and run programs. More detailed information can be found by referring to the appropriate DEC manuals listed in Table 2.

Table 2

DIGITAL EQUIPMENT CORPORATION'S SOFTWARE MANUALS

Manual	DEC Item Number
PDP 11/10	DEC-11-405AA-B-D
DOS Monitor	DEC-11-OMONA-A-D
FORTRAN	DEC-11-LFIVA-A-D
MACRO Assembler	DEC-11-OMACA-B-D
File Utility Package	DEC-11-UPUPA-B-D
Editor	DEC-11-EEDA-D
Debugger	DEC-11-OODA-D
Link	DEC-11-ULLMA-A-D

United Computing Corporation's United Disk System (UDS) - UDS is called under control of DEC DOS; but, when entered, it acts as an independent monitor. This capability exists because the software initially commandeers a large portion of the disk, preventing access by any other software package except its own. UDS allows the user to process APT source programs stored on any DOS support peripheral, allows transfer of source programs from any DOS-supported device into its own library, presents an editor for making changes to source programs stored in library, and allows processing of any programs or combinations of programs stored within the UDS library.

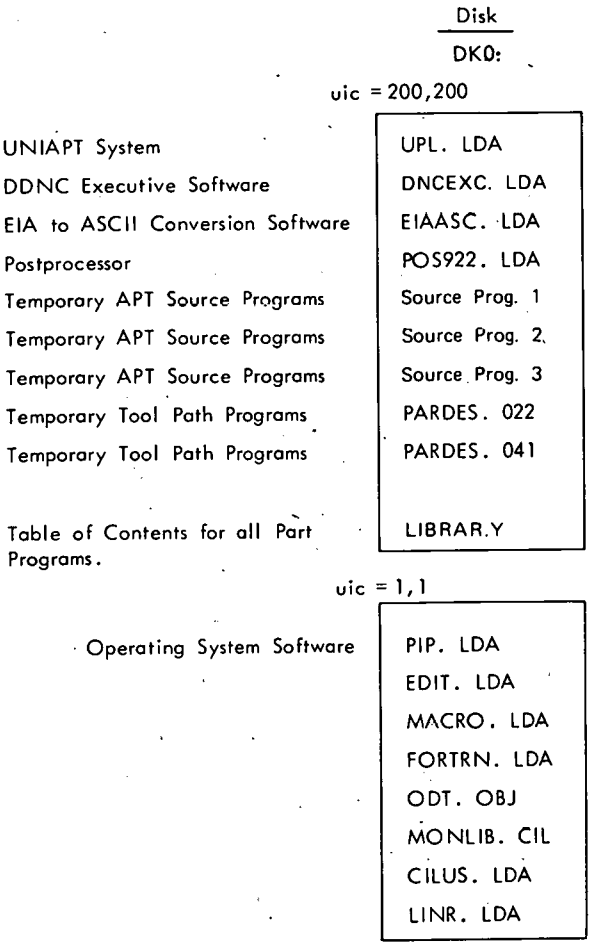


Figure 8. DISK FILE ORGANIZATION.

File Organization

Disk File - The disk serves as a permanent storage device (Figure 8) for the DOS operating system under user identification code (uic) 1,1. All other users have different access codes and cannot modify any program stored under this uic. Other programs pertaining to DNC operation are stored under uic 200,200. These programs include the UNIAPT software system, executive software for servicing DDNCs, postprocessor for the United APT processor, and various other programs connected with this operation. Disk files are also used as a scractch area (temporary) for the APT processor and various other programs connected with this operation. Disk files are also used as a scratch area (temporary) for APT source programs which are being developed and postprocessed object files (part programs) that required frequent access. Generally, these object files are

stored on one of the DECtapes where a significant amount of time (up to three minutes) may be required to search out a program for transmission to the DDNCs.

DECtape Files - The DECtapes are used to create permanent files for records and as a storage device for the part program libraries (Figure 9) accessed by the DDNCs, leaving the

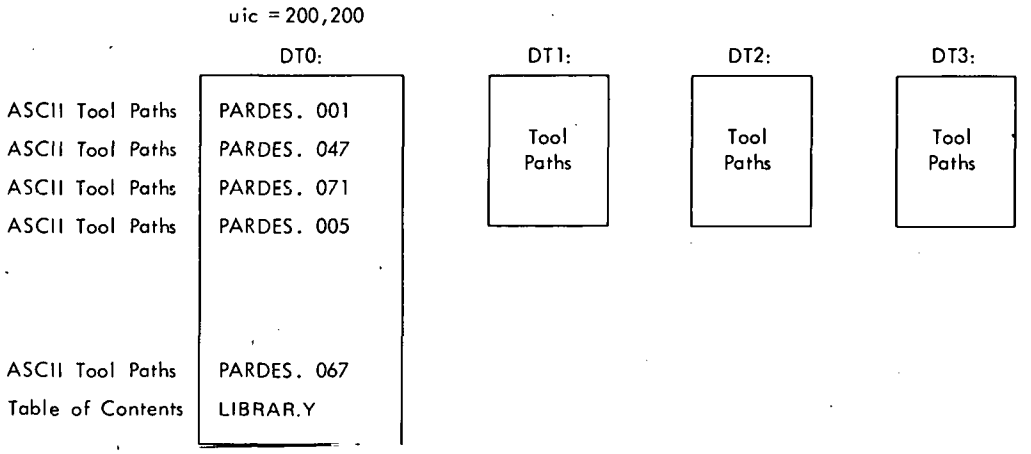


Figure 9. DECTAPE PART PROGRAM LIBRARY.

disk available for temporary storage of APT source programs and frequently used part programs. Each of the four units will store approximately 30 part programs which are equivalent in length to 125 feet of punched tape. All programs are stored with a fixed file name and a variable numerical extension (PARDES.nnn, where nnn can vary between 1 and 999). Each tape also has a file called LIBRAR.Y which contains a table of contents listing each part program existing on the particular tape.

United Disk System Disk Storage - Three storage areas within this portion of the disk are important to the reader (Figure 10): the extended disk, the editor library, and the editor scratch area. As stated earlier, the APT processor will under most conditions create a temporary CL file on the disk. This file is located in the UPL extended disk area for access by the postprocessor. CL data will remain in this disk area until another APT source program is processed. The editor library is used as permanent storage of APT MACROS and temporary storage of APT source programs that are being developed. These files can originate from any logical DEC peripheral and can be accessed by the UDS editor and APT processor. The editor scratch area is set aside for temporary storage of any APT source or EIA object program which is being edited by the UNIAPT software system.

Processing an APT Program

Processing Using the United Disk System Editor - An outline of the necessary steps in processing an APT program is: (A detailed step-by-step procedure is given in Appendix C.)

1. Input the source program from the teleprinter keyboard or card reader. Use PIP (peripheral interchange programs), when inputting from keyboard, to create a disk file which is then transferred to the UDS editor library under control of the UDS operating

system. Cards are directly transferred to the editor library without creating a disk file.

2. Run the APT processor under control of the UDS editor.
3. Correct any errors using the UDS editor.
4. Reprocess the corrected APT source program.
5. Repeat Steps 3 and 4 until all errors have been eliminated.
6. Postprocess the CL data in the UDS extended disk area. If errors show up, go back to Step 3.
7. Store the APT source program on DECtape for permanent records using the UDS editor.
8. Give the postprocessed object a file name that can be accessed by the DDNCs and store on DECtape using PIP.
9. Update the table of contents (LIBRAR.Y) on the DECtape where the object program was stored and also on the disk. Disk file LIBRAR.Y contains the table of contents for part programs on the four DECtapes and disk.

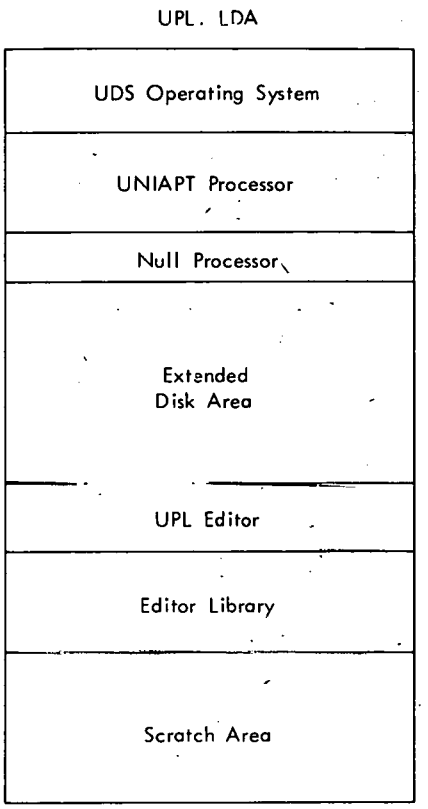


Figure 10. UNITED DISK SYSTEM DISK STORAGE.

Processing Using the UDS Operating System - This method is optional and can be used by the operator that prefers using the DOS editor (which is more powerful) instead of UDS, and where no APT MACROS are needed. Steps in the method are:

1. Input the APT source programs from the teleprinter keyboards or card reader under control of PIP and create an APT source file on the disk.
2. Process this APT file under control of the UDS operating system.
3. Edit out any errors using the DOS editor.
4. Repeat Steps 2 and 3 until no errors are observed.
5. Postprocess the CL data stored in the UDS extended disk area.
6. Go back to Step 2 if any errors occur.
7. Store the APT source program on DECtape using PIP.

8. Rename object for access by DDNCs and store on DECtape using PIP.

9. Update table of contents on this DECtape and on the disk.

Additional information concerning system operations is provided in Appendixes D and E.

CONCLUSIONS

The postprocessor has been used to successfully generate object programs for several series of special production parts. These parts are typical of those manufactured at this facility which involve close-tolerance turning operations on point and mathematically defined shapes. The postprocessor is continually being updated and improved (along with the entire DNC system) and additional features are being added even though it is routinely used in the manufacturing of special production parts.

The postprocessor along with the UCC APT processor and DNC system offer an efficient and relatively inexpensive method for acquiring quick turn arounds in the shops. Even though the computer processing time is considered lengthy when compared to the much larger and more expensive computer systems, as many as five or six turn arounds per shift can be made on programs that include TABCYLs. (Point-defined contours, or TABCYLs, require more processor time than any other type of contour.) The number of turn arounds becomes increasingly larger when parts having mathematically defined contours are scheduled, inasmuch as the program processing becomes less complicated.

ACKNOWLEDGEMENTS

The author wishes to express his thanks to C. M. Davenport and L. E. Bearden of the Y-12 Laboratory Development Department for their consultations concerning the FORTRAN portions of the programming.

APPENDIX A

SOFTWARE FOR POSTPROCESSING APT CL DATA GENERATED BY UNITED COMPUTING CORPORATION'S APT PROCESSOR FOR THE PDP-11/10 MINI-COMPUTER

Software Routine For Startup and Record-Type Interrogation

This routine (Figure A-1) is entered via the DOS command "RUN POS922". When entered, the software is initialized via subroutine "INITIA" for the Ex-Cell-O 921/922 turning machine with a DDNC or Bendix 1800 Controller. The software then looks for DOS file "DISK.UPL" on Disk Unit 0 which is the starting location of the entire UNIAPT software system. This procedure is used to find where the CL data resides on the disk (CL data starts 600,000 (8) words beyond the start of DISK.UPL). At this time, the first 512-word block of information is read from the disk into the computer memory. Hereafter, this 512-word area will be automatically refilled when all data have been processed (data are retrieved from the 512-word buffer by subroutine "GETUPL" which checks for the end of this area each time a UPL word, two DEC words, is retrieved). After the 512-word block is transferred, a check is made for the record type. This information is always at the beginning of a record and is stored in "FSTUPL" by subroutine "GETUPL". Record type can be 0, 1, 2, 3, or 4 and determines the direction of program flow.

Type 0 Record Processing Software - Type 0 record (Figure A-2) is always the APT FINI statement signifying the end of the APT program. When this statement is detected, the record number and "FINI" are outputted to the teleprinter, an ASCII "$" is stored on the last disk record, a leader is punched on tape, all open files are closed, all DOS drivers are released, and the postprocessor is exited with program control returning to DOS.

Type 1 Record Processing Software - Type 1 records are always postprocessor commands. This routine polls each record for every legal command until the one residing in the present record is found. The order of polling is listed in Table A-1. Action taken when a subclass is recognized is dependent on that subclass. (Refer to Figure A-3 in the following discussions.)

Table A-1

CLASS ONE VOCABULARY WORDS

Word	Subclass 1
AUXFUN	1022.
COOLNT	1030.
DELAY	1010.
DISPLY	1021.
END	1.
FEDRAT	1009.
LEADER	1013.
MACHIN	1015.
MACTOL	1016.
OPSTOP	3.
PARTNO	1045.
INSERT	1046.
PPRINT	1044.
RAPID	5.
REWIND	1006.
SEQNO	1017.
SPINDL	1031.
STOP	2.
TURROT	1033.

AUXFUN Subclass 1022 - Auxiliary functions can be AUXFUN/12, 13, 17, or 18. These functions are processed and converted to M12, M13, M17, or M18 and outputted by subroutines for processing M numbers (LSTMNU and PUNMNU). The correct M number is stored on disk using subroutine DKIN. Routine starts at "JJ" of Figure A-3.

COOLNT Subclass 1030 - This statement can be followed by either Subclass 8 statements "ON" or "OFF" (EXP. COOLNT/OFF). A list of these

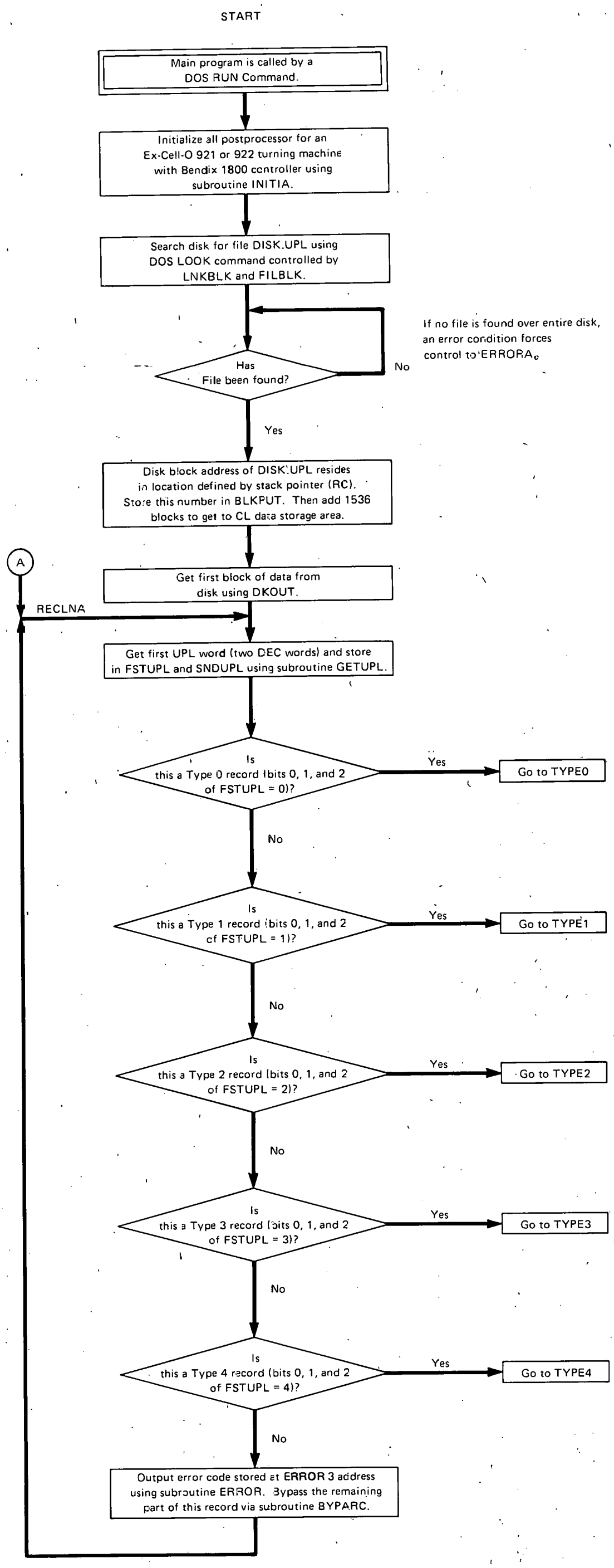


Figure A-1. FLOWGRAPH FOR STARTUP AND RECORD-TYPE INTERROGATION.

TYPE 0

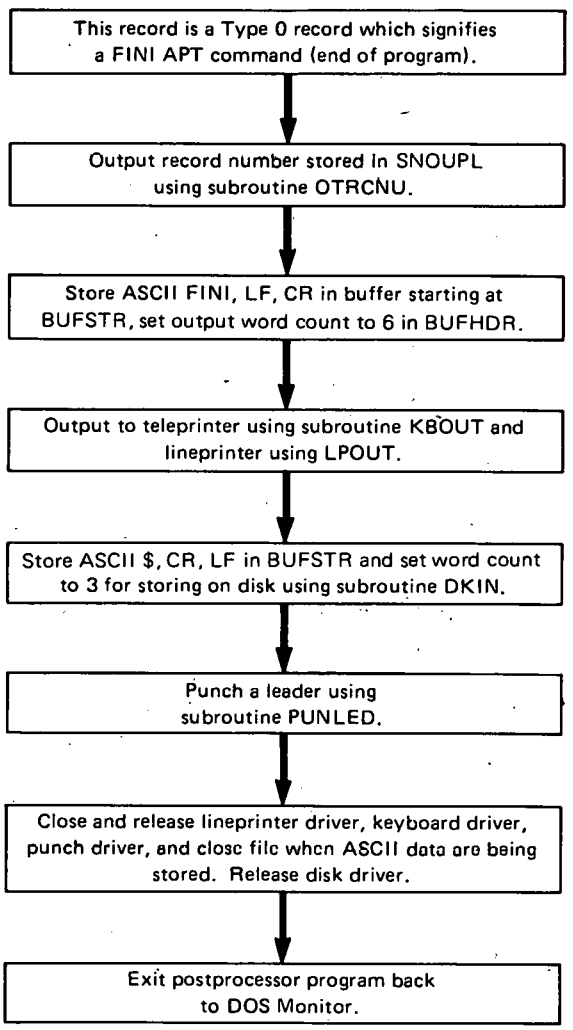


Figure A-2. TYPE 0 RECORD PROCESSING SOFTWARE FLOWGRAPH.

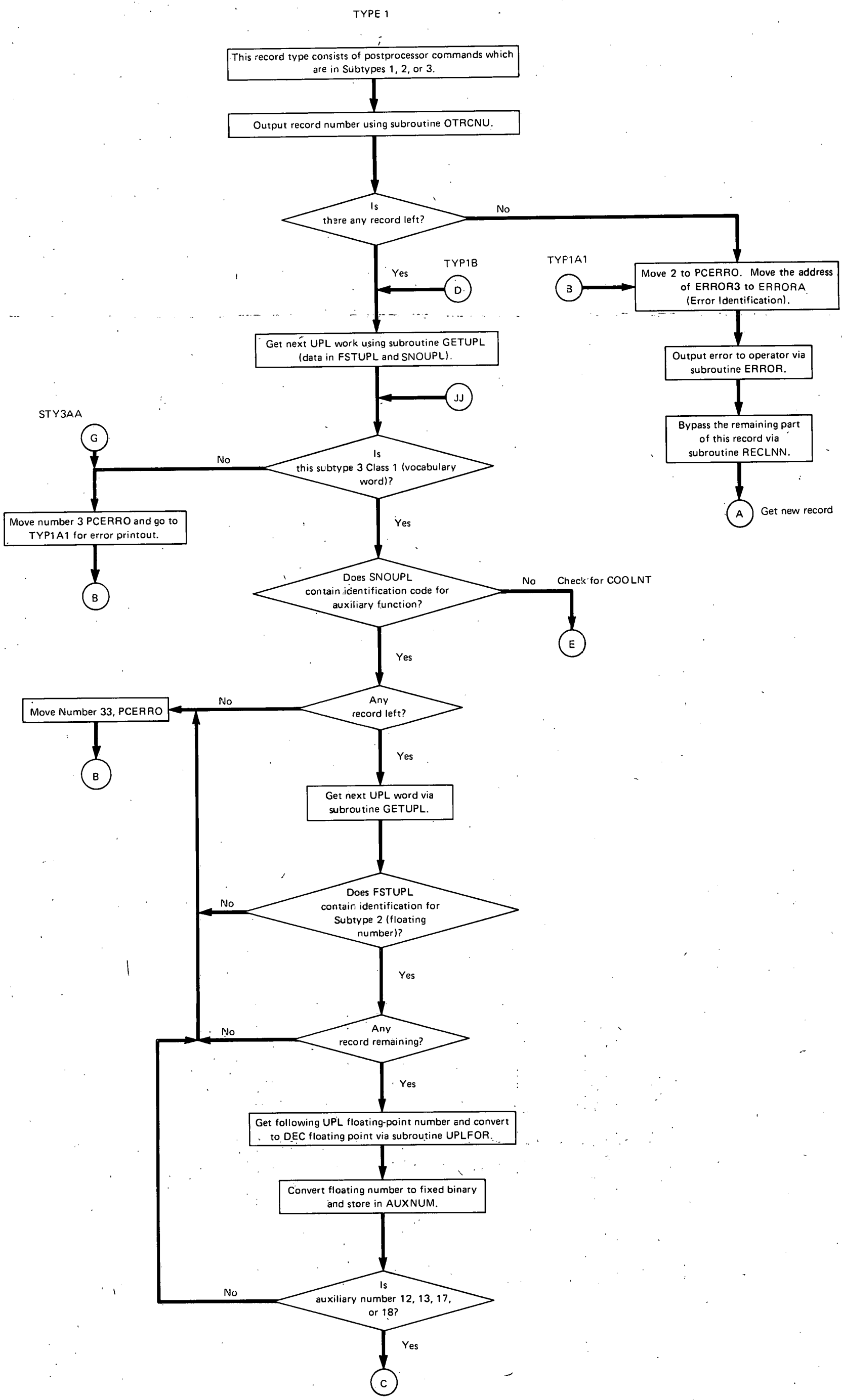


Figure A-3. TYPE 1 RECORD PROCESSING SOFTWARE FLOWGRAPH. (Section 1)

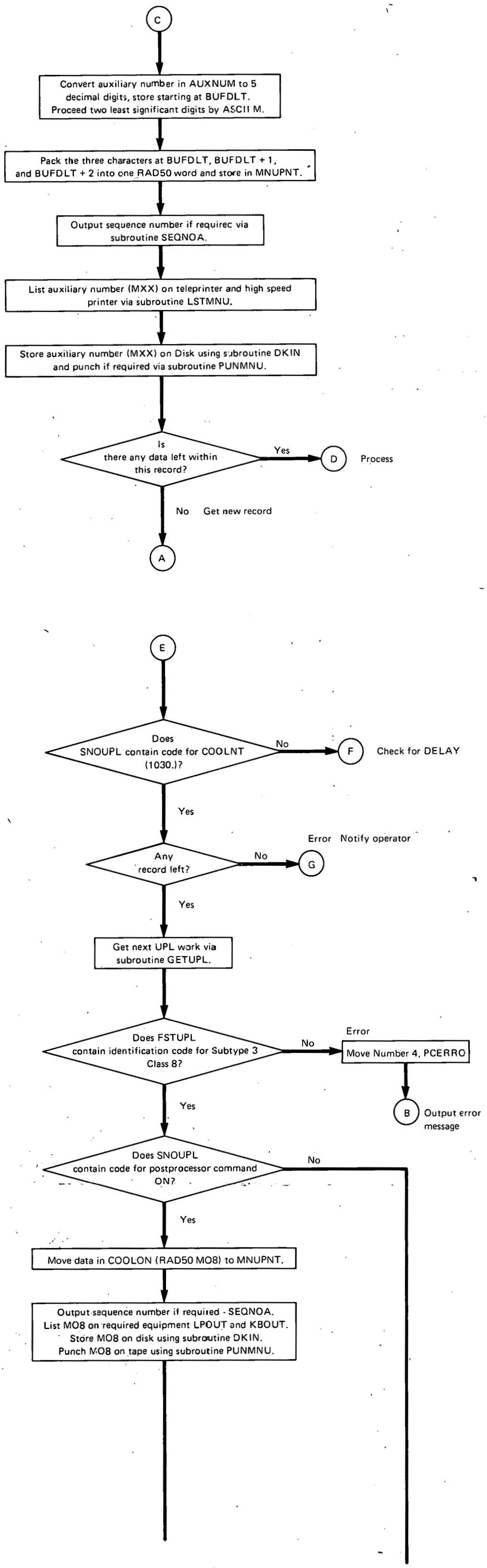


Figure A-3. (Section 2)

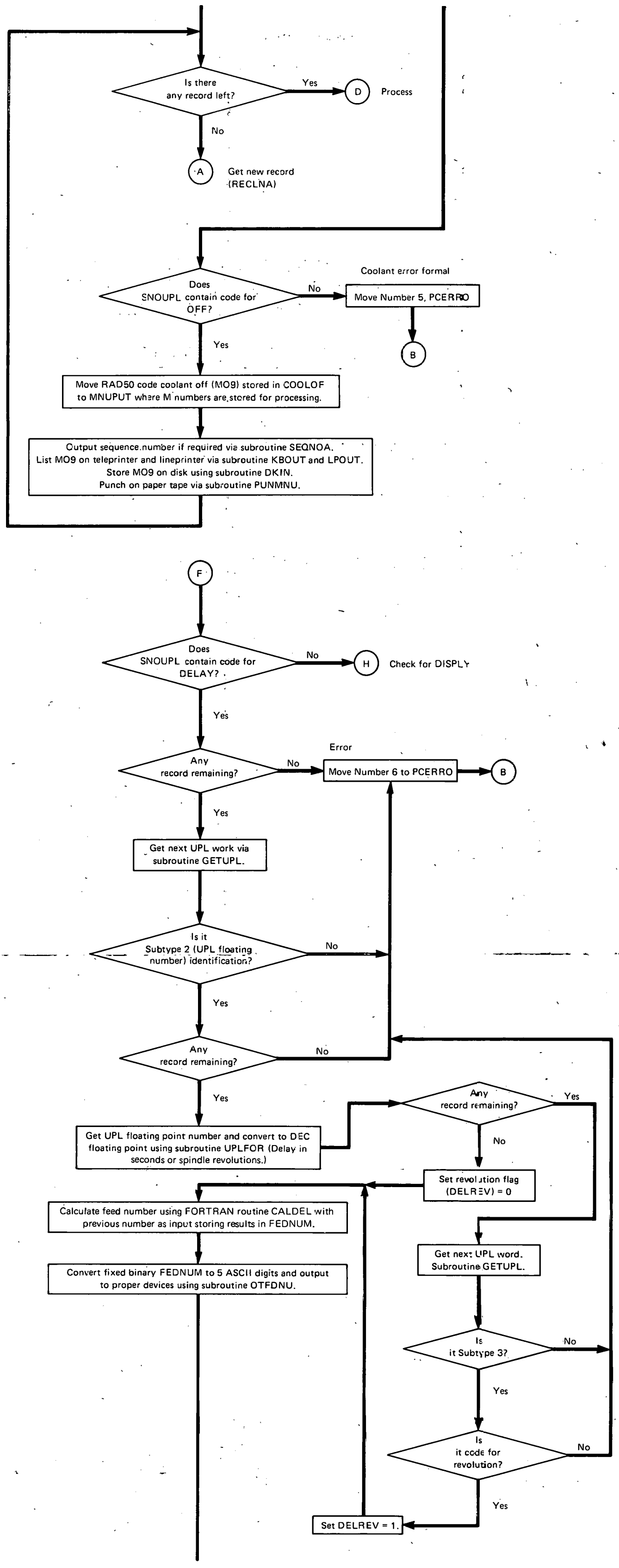


Figure A-3. (Section 3)

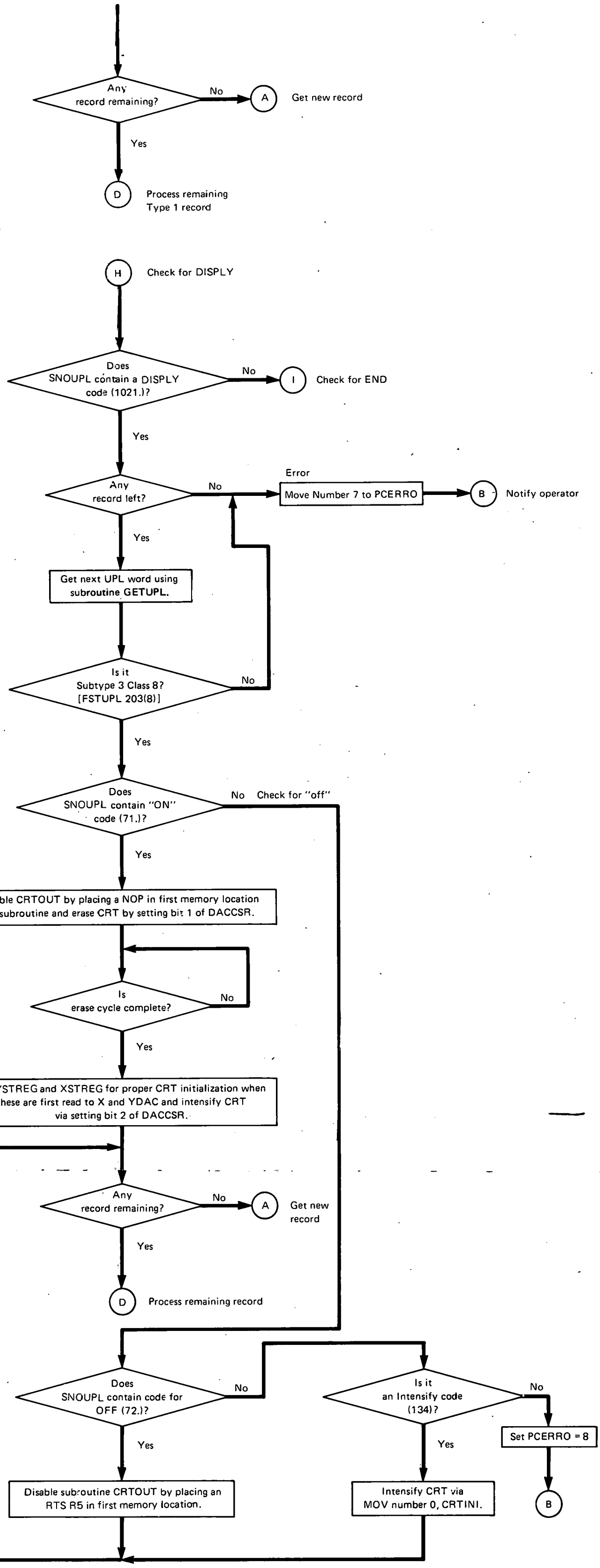


Figure A-3. (Section 4)

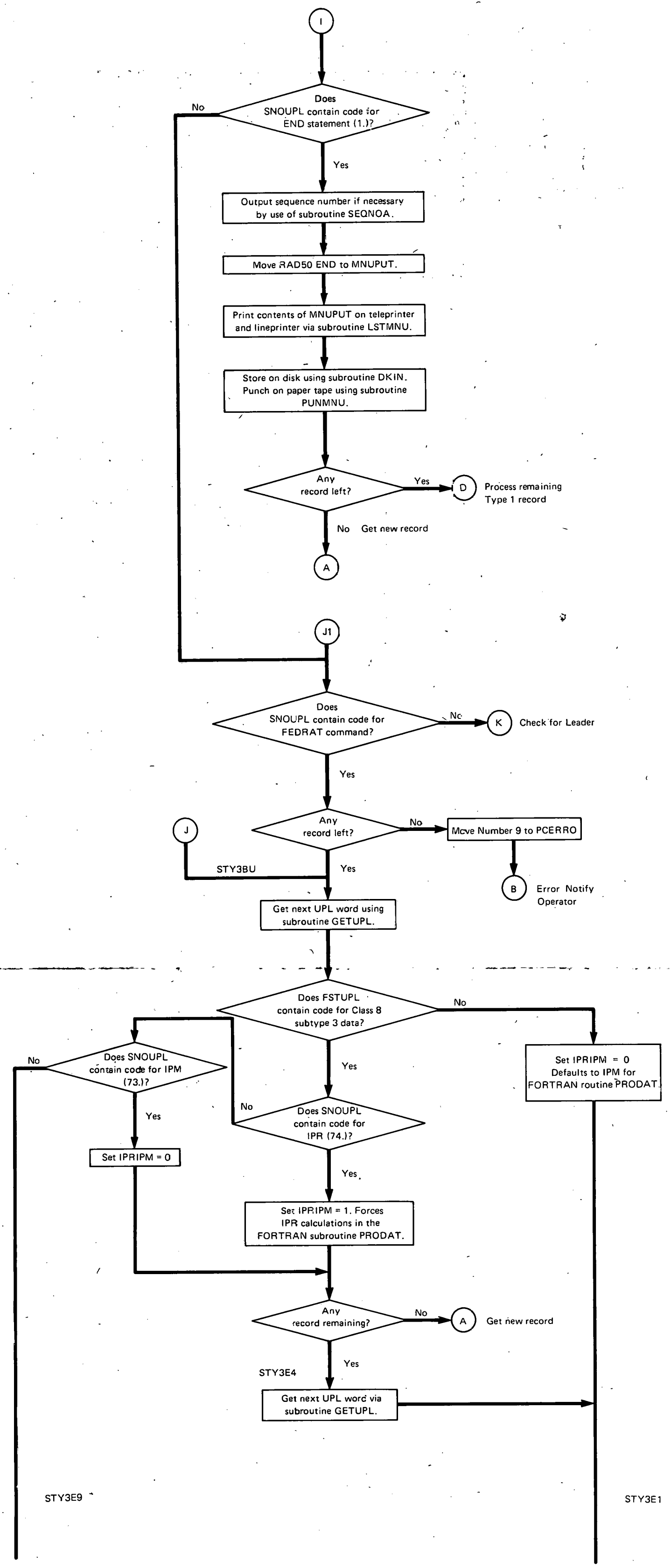


Figure A-3. (Section 5)

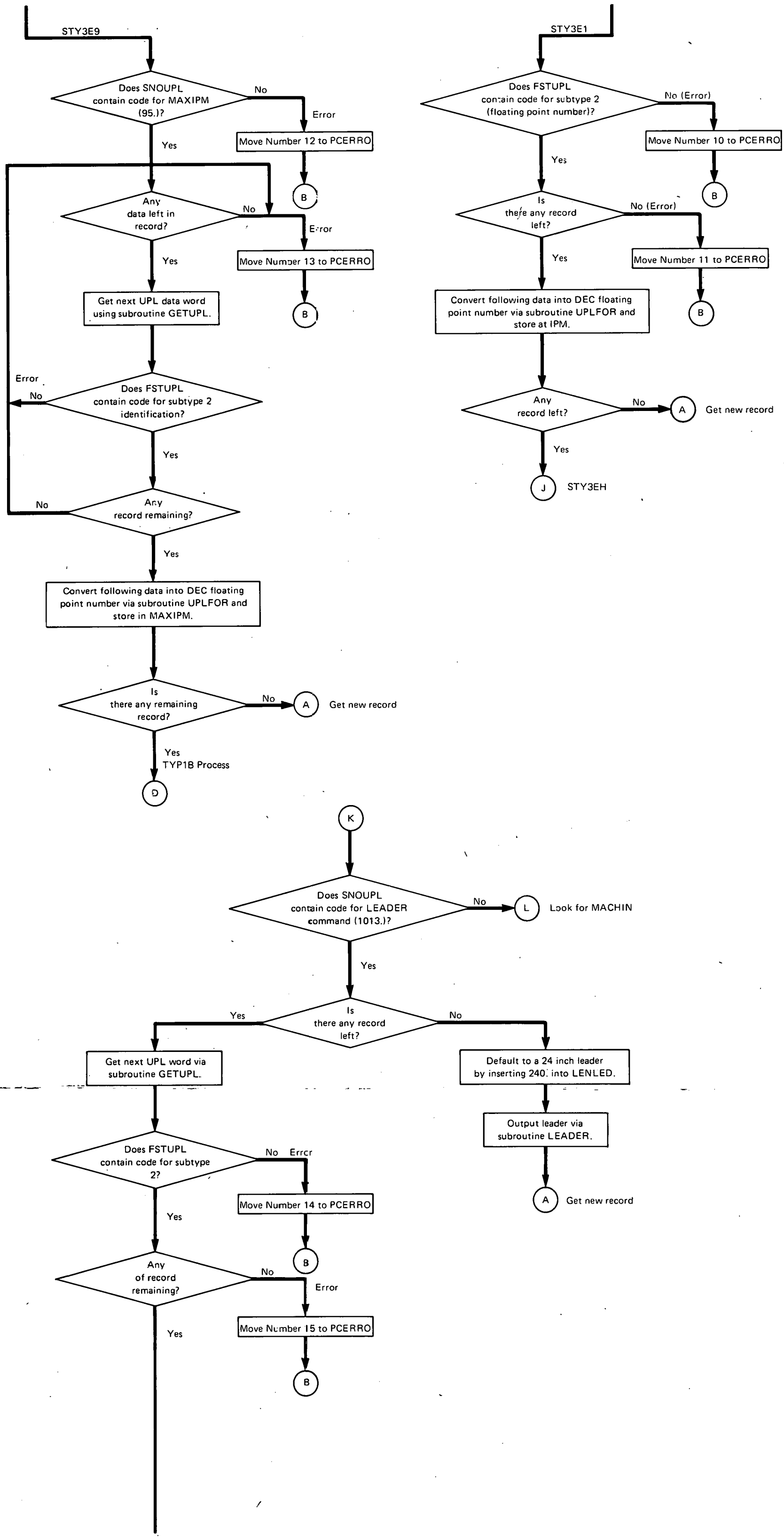


Figure A-3. (Section 6)

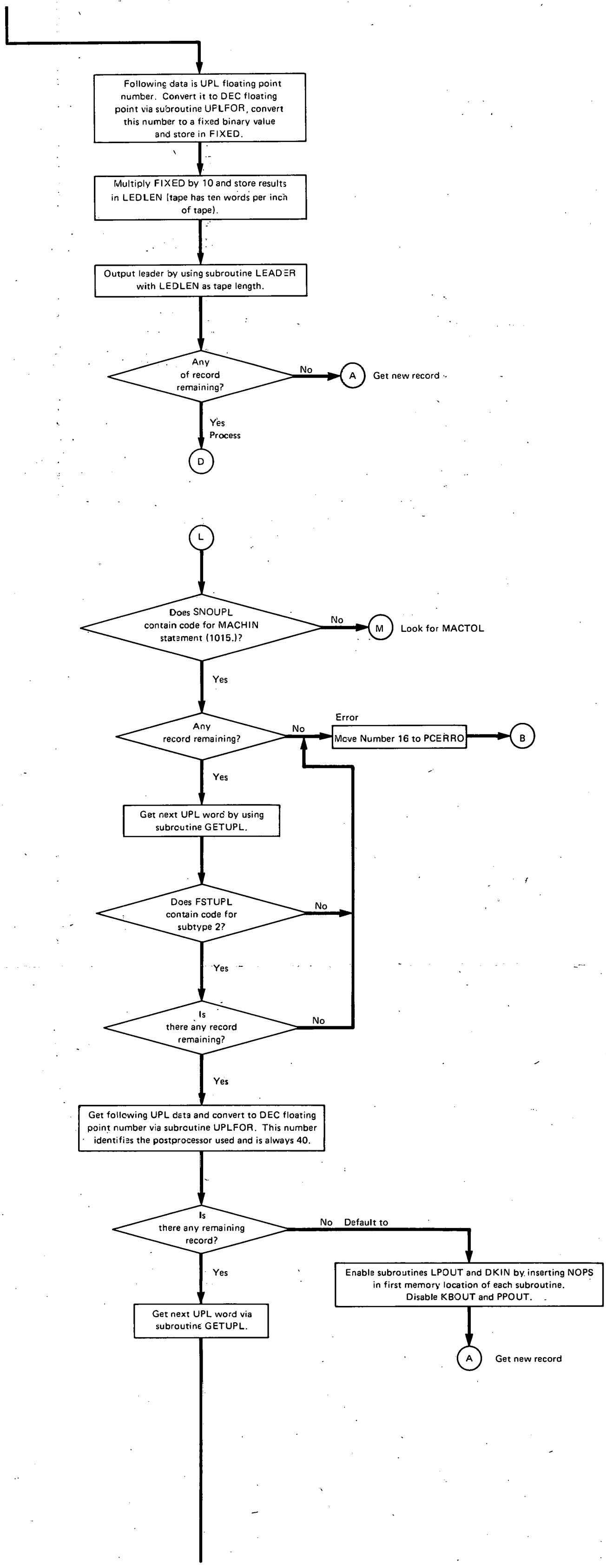


Figure A-3. (Section 7)

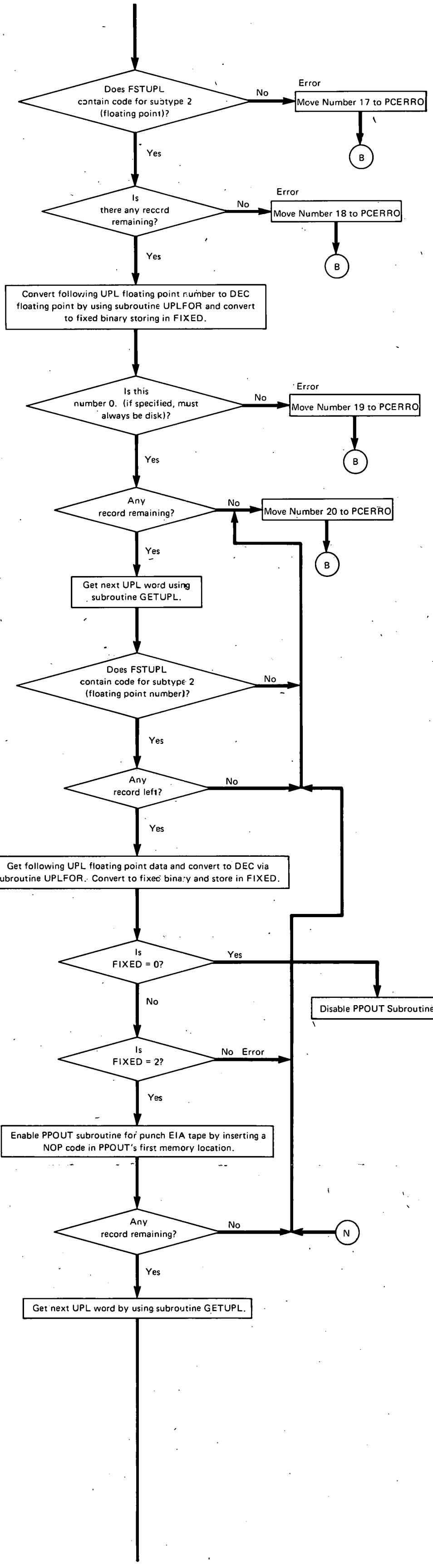


Figure A-3. (Section 8)

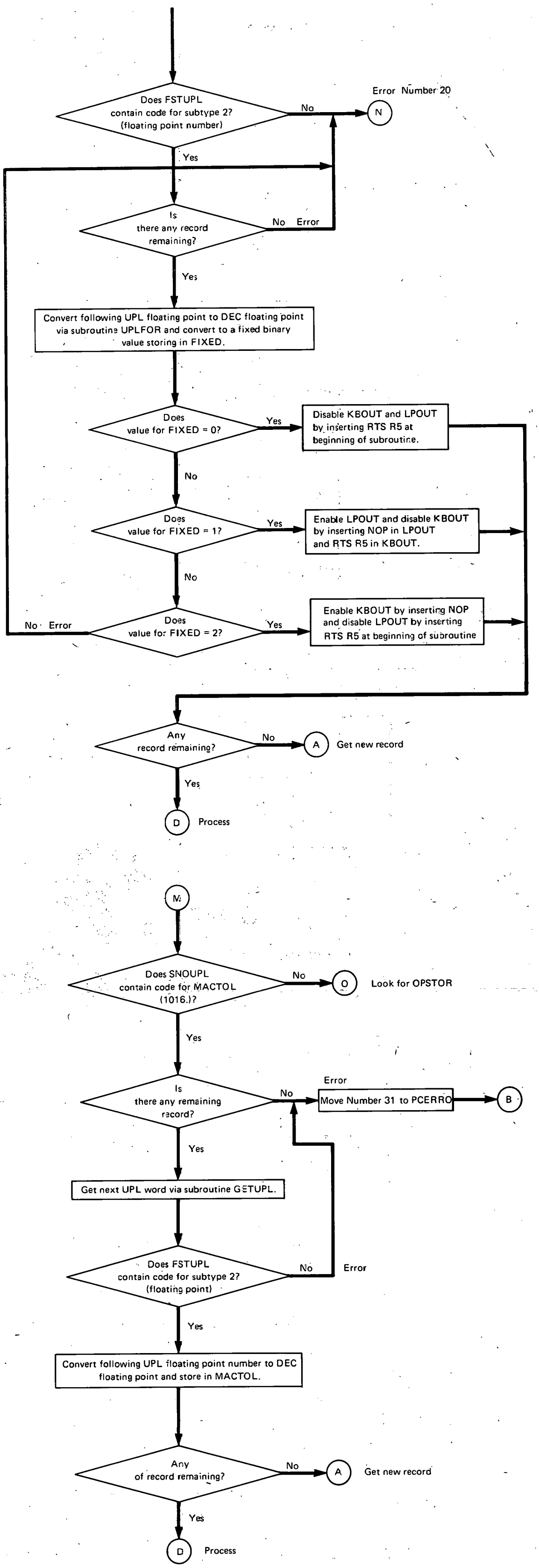


Figure A-3. (Section 9)

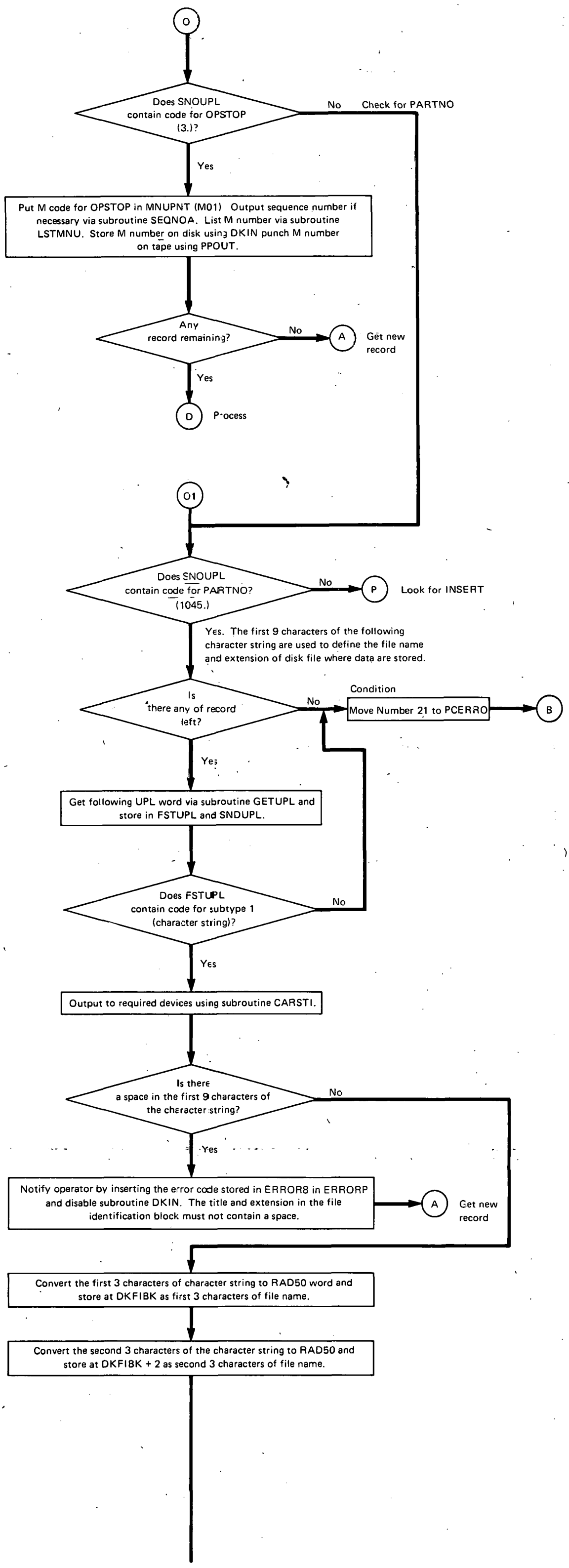
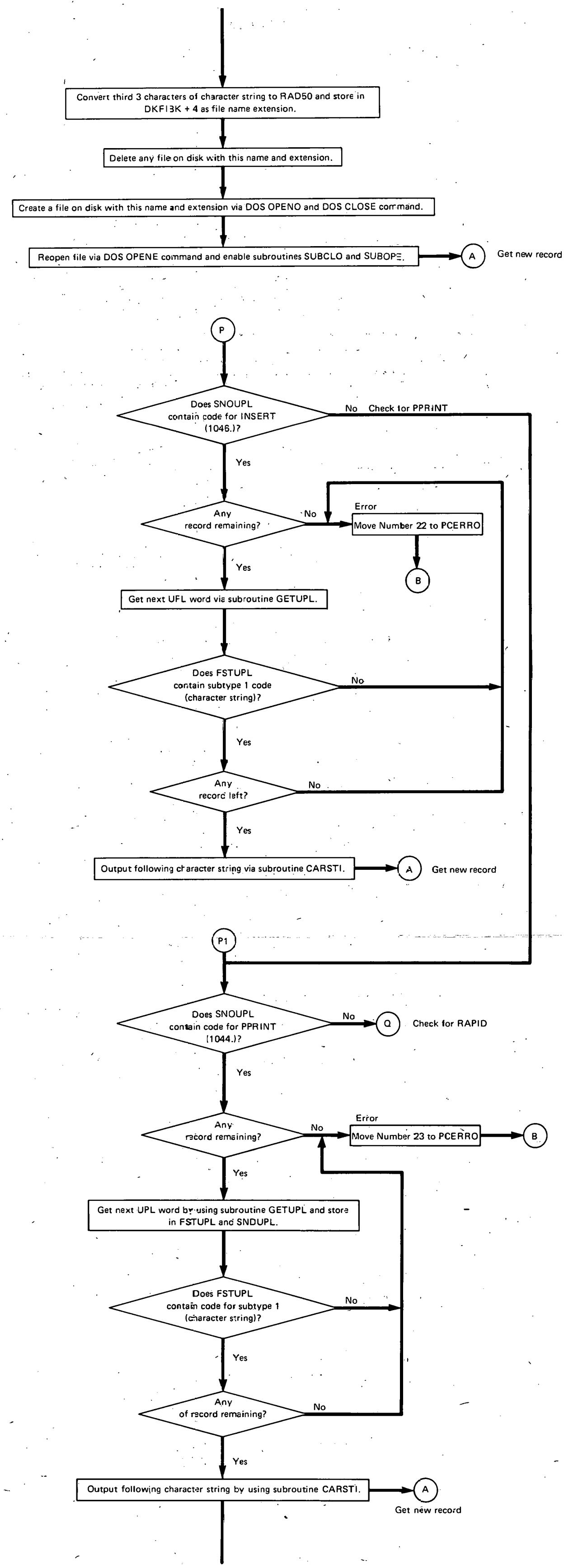


Figure A-3. (Section 10)


Convert third 3 characters of character string to RAD50 and store in DKF13K + 4 as file name extension.
Delete any file on disk with this name and extension.
Create a file on disk with this name and extension via DOS OPENO and DOS CLOSE command.
Reopen file via DOS OPENE command and enable subroutines SUBCLO and SUBOPE.
A
Get new record
P
Does SNOUPL contain code for INSERT (1046.)?
No
Check for PPRINT
Yes
Any record remaining?
No
Error
Move Number 22 to PCERRO
B
Yes
Get next UFL word via subroutine GETUPL.
Does FSTUPL contain subtype 1 code (character string)?
No
Yes
Any record left?
No
Yes
Output following character string via subroutine CARSTI.
A
Get new record
P1
Does SNOUPL contain code for PPRINT (1044.)?
No
Q
Check for RAPID
Yes
Any record remaining?
No
Error
Move Number 23 to PCERRO
B
Yes
Get next UPL word by using subroutine GETUPL and store in FSTUPL and SNDUPL.
Does FSTUPL contain code for subtype 1 (character string)?
No
Yes
Any of record remaining?
No
Yes
Output following character string by using subroutine CARSTI.
A
Get new record

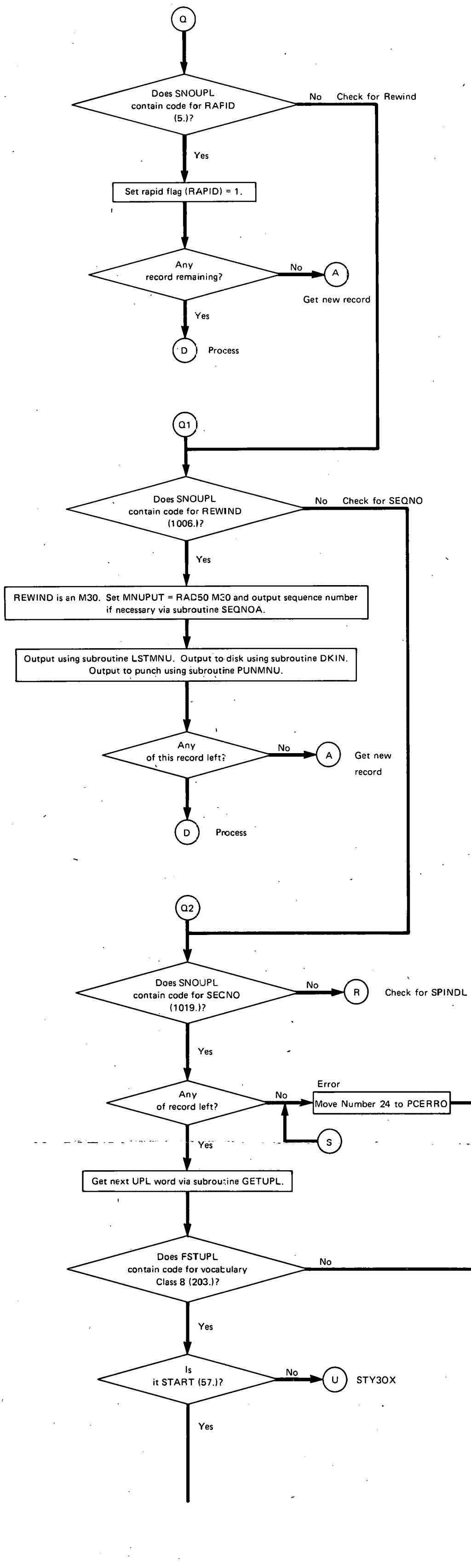


Figure A-3. (Section 12)

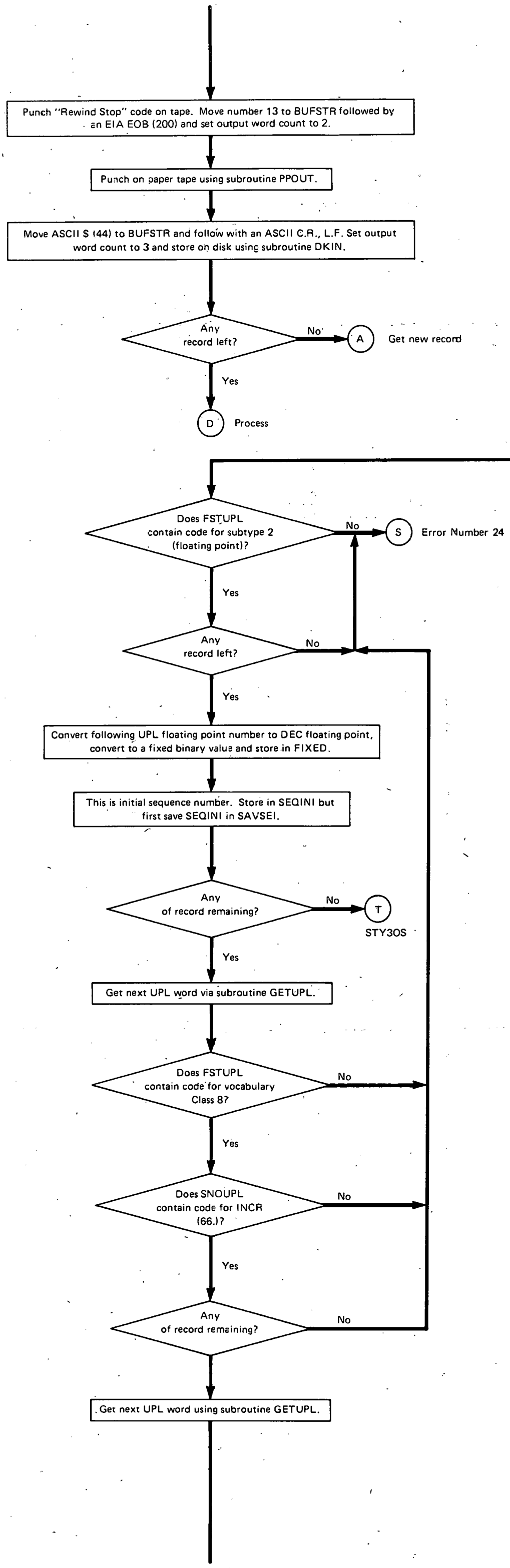


Figure A-3. (Section 13)

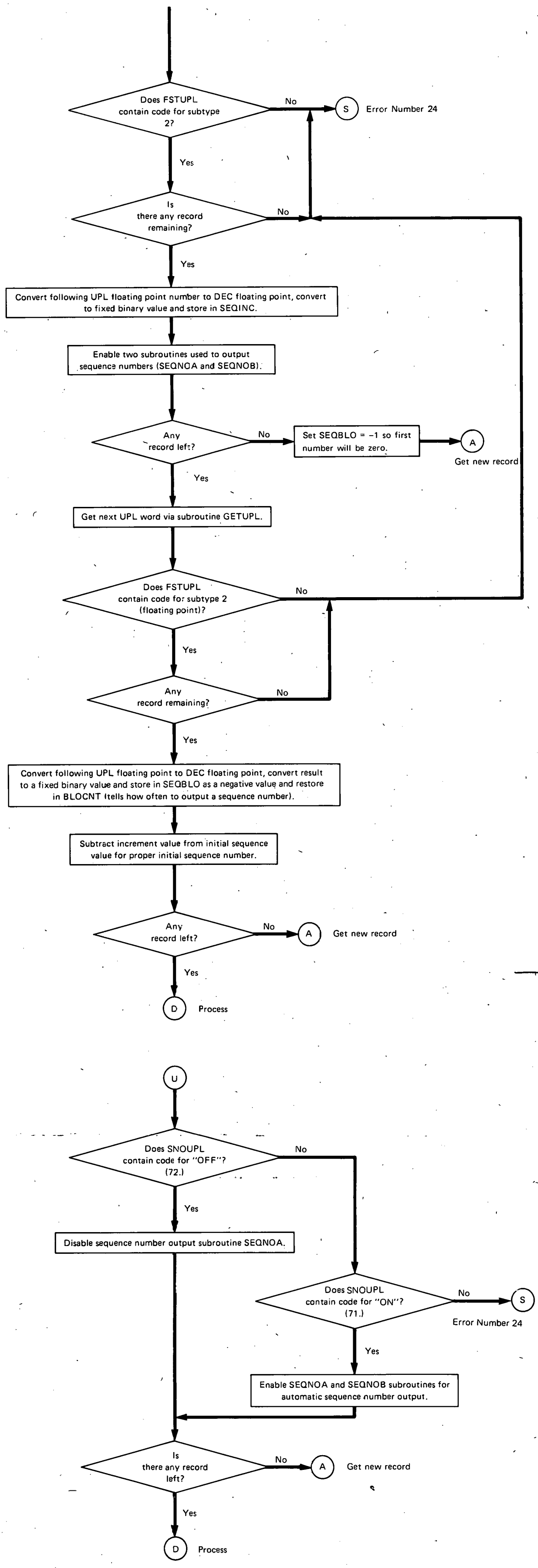


Figure A-3. (Section 14)

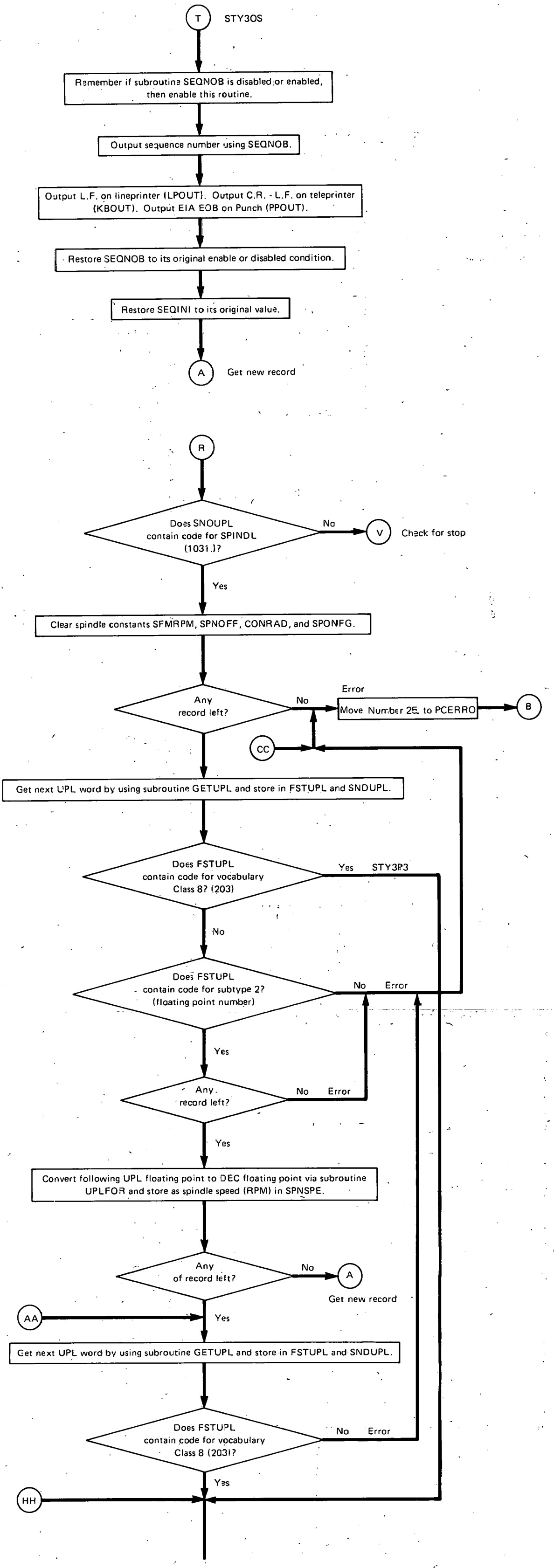


Figure A-3. (Section 15)

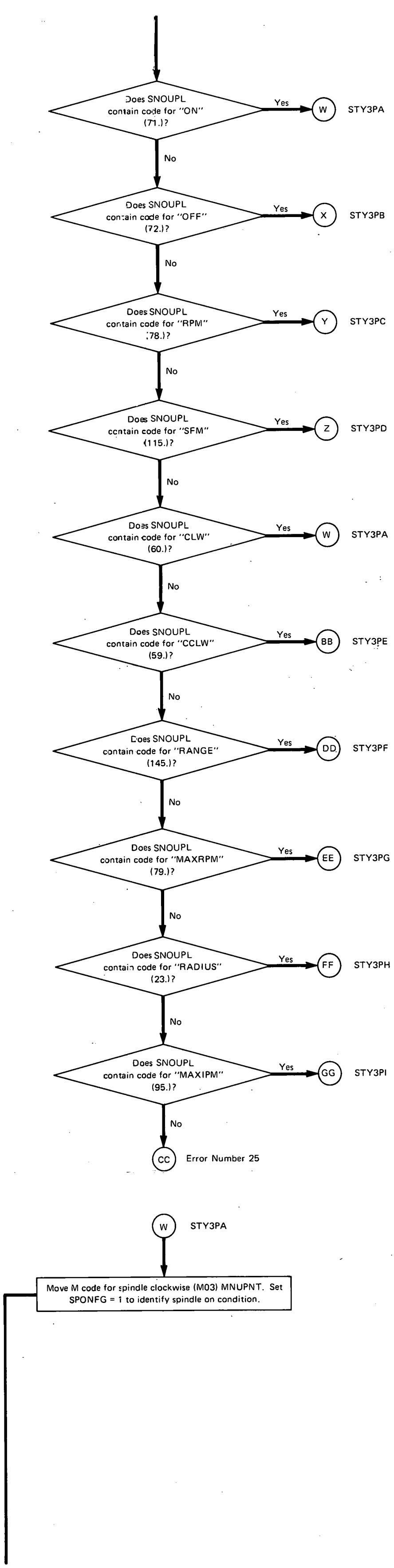


Figure A-3. (Section 16)

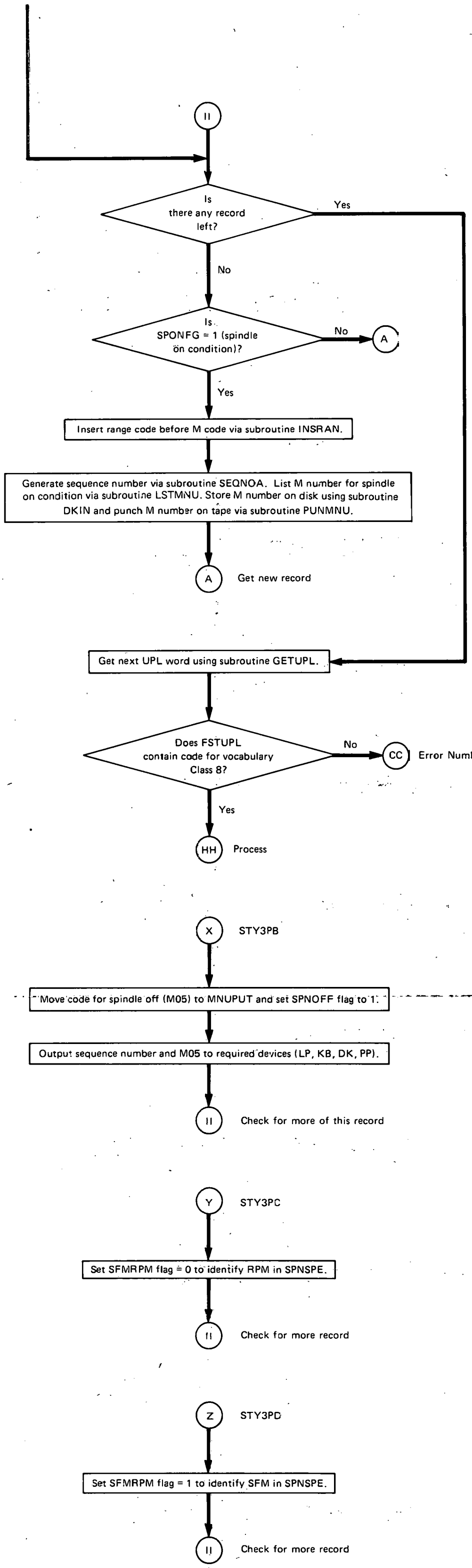


Figure A-3. (Section 17)

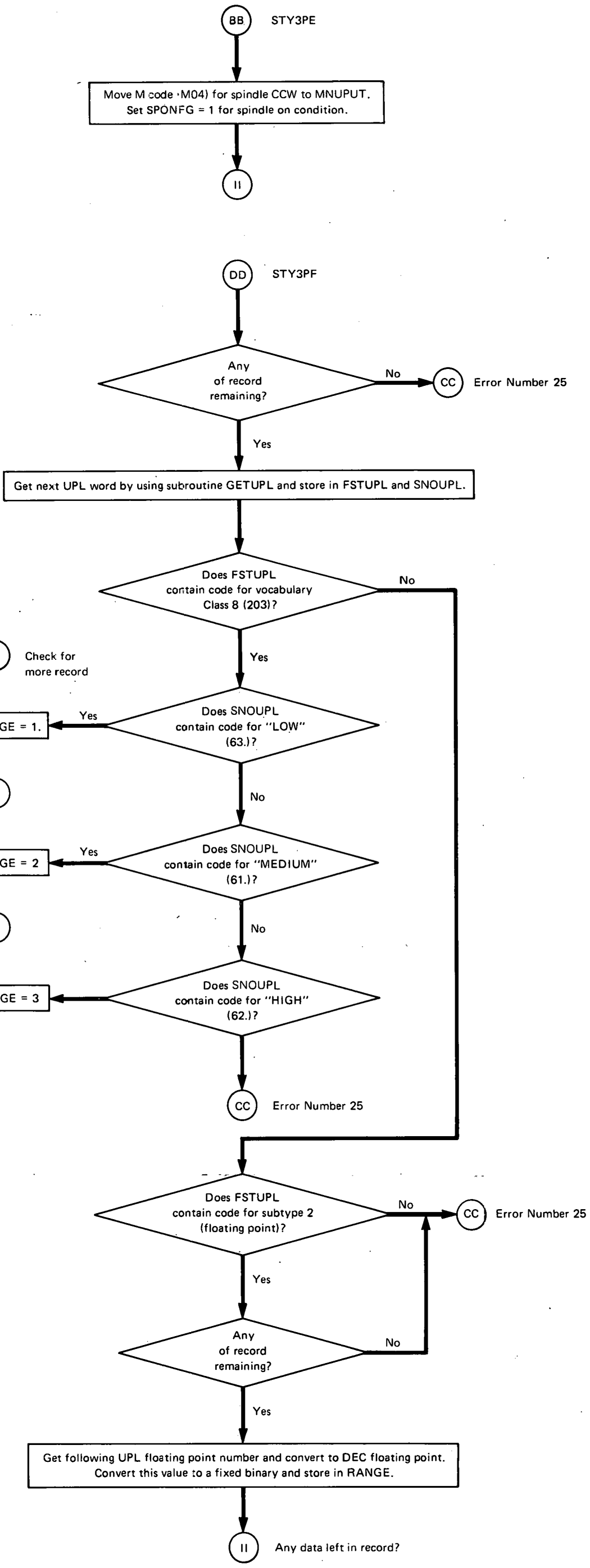


Figure A-3. (Section 18)

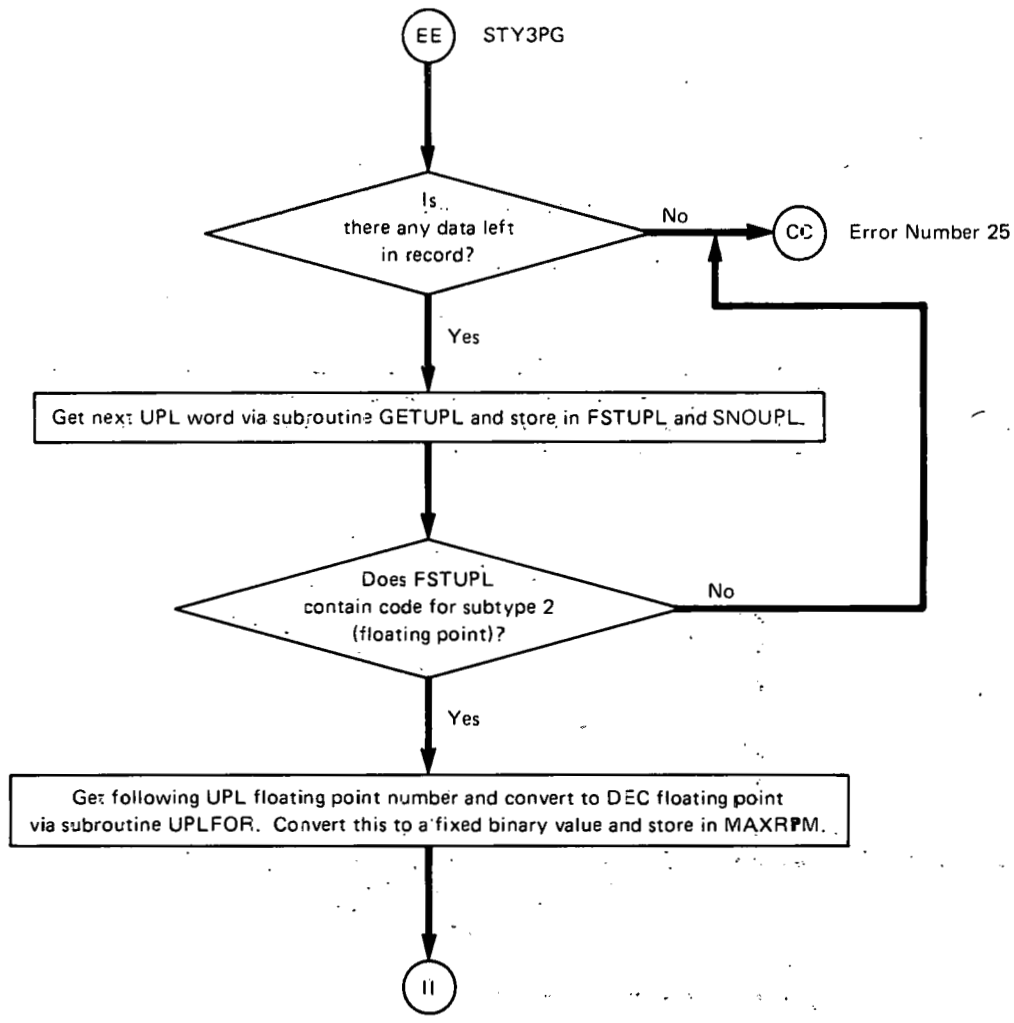


Figure A-3. (Section 19)

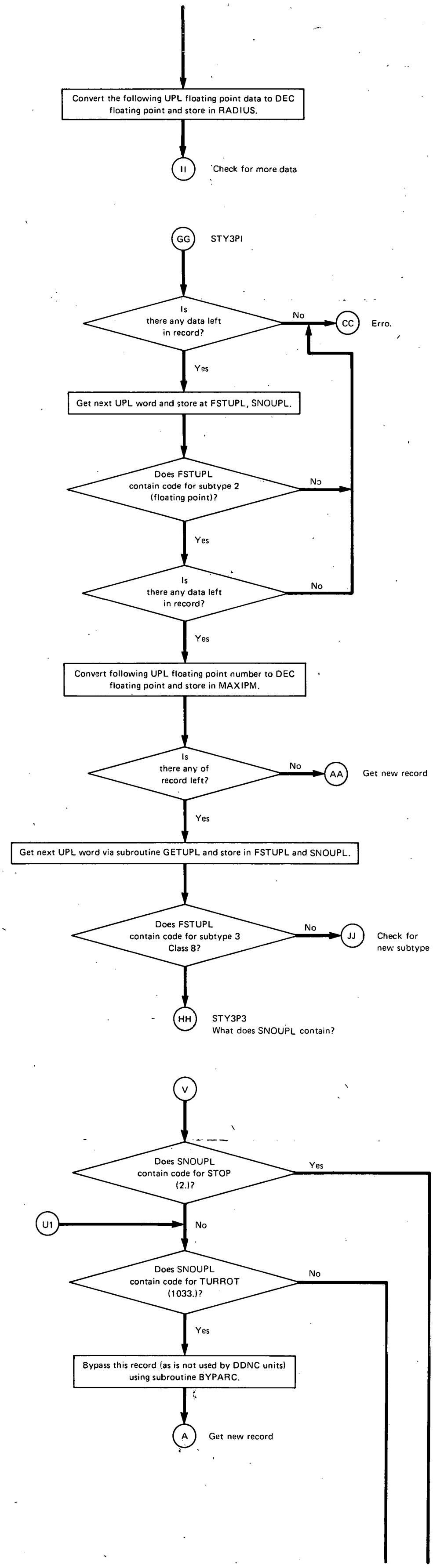


Figure A-3. (Section 20)

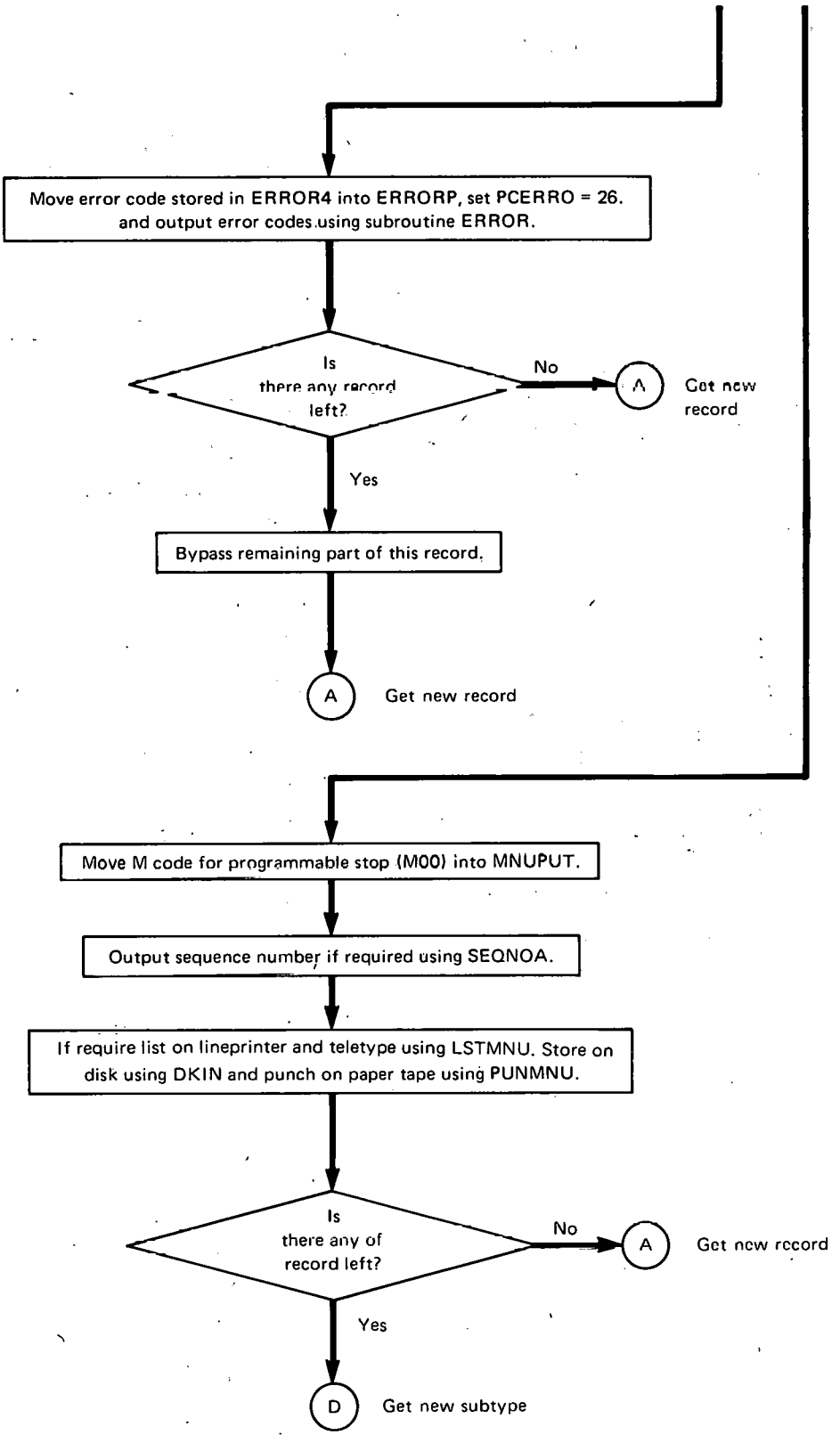


Figure A-3. (Section 21)

types of modifiers is given in Table A-2. "Coolant on" is treated as M08 and "coolant off" as M09. These are outputted as any other M number. Routine starts at E of Figure A-3.

Table A-2

CLASS 8 VOCABULARY WORDS

Word	Subclass 8	Function
CCLW	59.	Counterclockwise
CLW	60.	Clockwise
INTENS	134.	Intensify CRT
HIGH	62.	Highest Gear Range
IPM	73.	Inches per Minute
IPR	74.	Inches per Revolution
LOW	63.	Lowest Gear Range
MAXIPM	95.	Maximum Inches per Minute
MAXRPM	79.	Maximum Revolutions per Minute
MEDIUM	61.	Medium Gear Range
OFF	72.	Off
ON	71.	On
RANGE	145.	Gear Range Follows
RADIUS	23.	Coordinate for Feed-Number Calculation Follows
REV	96.	Reverse
RPM	78.	Revolution per Minute
SFM	115.	Surface Feed per Minute
XCOORD	116.	Use X coordinate for Feed Calculation
YCOORD	117.	Use Y Coordinate for Feed Calculation
ZCOORD	118.	Use Z Coordinate for Feed Calculation

DELAY Subclass 1010 - The delay statement format is DELAY/n,REV. "n" is normally in seconds unless it is followed by REV which defaults n to revolutions of the spindle. The delay statement is outputted as FNNNN, where NNNN is calculated by FORTRAN subroutine CALDEL, via subroutine OPTDNU.

DISPLY Subclass 1021 - The display statement is used to enable use of the CRT display peripheral and has a format of DISPLY/ON, DISPLY/OFF, or INTENS. "OFF" causes subroutine "CRTOUT" to be disabled (disables output to CRT), "INTENS" causes an intensity command to be sent to the CRT, and "ON" enables subroutine "CRTOUT", erases CRT, initializes digital-to-analog converts for driving the CRT, and intensifies the CRT. The routine starts at (H) of Figure A-3 (Section 4).

END Subclass 1 - END is processed as M02 (if sequence number output is required, refer to subroutine "SEQNOA") and is outputted on enabled devices using subroutines LSTMNU, DKIN, and PUNMNU. Routine starts at (I) of Figure A-3 (Section 5).

FEDRAT Subclass 1009 - Format for the feedrate statement is "FEDRAT/$\begin{Bmatrix}IPM\\IPR\end{Bmatrix}$, nn, MAXIPM, ii". If IPM or IPR is not included, a default flag is set for IPM (IPRIPM = 0). This flag is used by subroutine "PRODA1" where feedrate is

calculated (IPM also causes this flag to be set to 0; IPR causes "IPRIPM" to be set to 1). Again, this is used as a conditional argument in "PRODA1". "nn" is converted to a DEC floating number by subroutine "UPLFOR" and stored at "IPM" for both IPM and IPR. When MAXIPM is included, ii is converted to DEC floating point and stored in MAXIPM for use by subroutine PRODA1. This routine starts at (J1) of Figure A-3 (Section 5).

LEADER Subclass 1013 - This statement is used to punch the leader on the EIA tape. Its format is LEADER/n where "n" is in inches of tape. If "n" is not included, the default length is 24 inches. When "LEADER" is followed by "n", "n" is converted to DEC floating point by subroutine "UPLFOR" and stored as "n x 10" (10 characters per inch of punched tape) in "LEDLEN". In either case, the length of leader is outputted using subroutine "LEADER". This routine starts at (K) of Figure A-3 (Section 6).

MACHIN Subclass 1015 - The machine statement is used by both the processor and postprocessor and has a format of MACHIN/xx, i, j, k. "xx" is used by the processor for defining any UNIAPT postprocessor (must be 40 for the Y-12 system for calling the null postprocessor written by UCC). The following characters define postprocessor input/output options and are given in Table A-3. When i, j, and k are not included, the software defaults to i = 0, j = 0, and k = 1 for storage on disk, no punched tape, and output to lineprinter. This routine starts at (L) of Figure A-3 (Section 7).

Table A-3

POSTPROCESSOR INPUT/OUTPUT OPTIONS

i	Always disk	
j	0	Disable punched tape output.
	2	Enable punched tape output.
k	0	Disable putput to teleprinter and lineprinter.
	1	Enable lineprinter - disable teleprinter.
	2	Disable lineprinter - Enable teleprinter.

MACTOL Subclass 1016 - The format of the machine tolerance statement is MACTOL/n, where "n" is in inches. This number is stored in "MACTOL" for use by subroutine "PRODA1".

OPSTOP Subclass 3 - When OPSTOP (optional stop) is detected, an M01 is outputted as other M numbers using subroutines LSTMNU, DKIN, and PPOUT.

PARTNO Subclass 1045 - The part number statement is followed by 66 characters which are always outputted to the teleprinter. The first nine characters are used to define and open a file on disk where the postprocessor's output is stored. These nine characters should conform with restrictions imposed by DOS. Any legal file with this name and extension will be automatically deleted before an open attempt is made. This action can lead to problems if this file was not previously closed properly and will be identified with a DOS fatal error message "FXXX". This routine starts at (O) of Figure A-3 (Section 10).

INSERT Subclass 1046 - INSERT, as was PARTNO, is followed by an ASCII 66 character string. These characters are outputted to the enabled output device via subroutine CARSTI. This routine starts at (P) of Figure A-3 (Section 11).

PPRINT Subclass 1044 - PPRINT is handled as was INSERT and starts at (P1) of Figure A-3 (Section 11).

RAPID Subclass 5 - The rapid statement sets RAPID = 1 which overrides feedrate calculations in subroutine "PRODA1", forcing a feedrate that is equal to the maximum of the machine tool's configuration. Routine starts at (Q) of Figure A-3 (Section 12).

REWIND Subclass 1006 - This routine starts at (Q1) of Figure A-3 (Section 12). It is generally inserted at the end of the tool path to rewind the tape or force the DDNC to address its initial core location. When encountered, an M30 is generated and outputted as other "M" numbers using subroutine "LSTMNU", "DKIN", and "PUNMNU".

SEQNO Subclass 1019 - The format of the sequence number statement is "SEQNO/START", SEQNO/ON", "SEQNO/OFF", or "SEQNO/i, INCR, j, k". "SEQNO/START' causes a rewind-stop code to be punched and stored at the beginning of the tool path; "SEQNO/ON" enables subroutines "SEQNO" and "SEQNOB" for output of sequence numbers; "SEQNO/OFF" disables these subroutines. The last format causes sequence numbers to be outputted starting with an initial value of "i", incremented by "j", every "k" tool path blocks by enabling "SEQNOA" and "SEQNOB" after properly initializing these subroutines. This routine starts at (Q2) of Figure A-3 (Section 12).

SPINDL Subclass 1031 - The spindle control statement format is given as follows: This routine checks the

$$\left\{\begin{array}{l} \\ \text{SPINDL/}\begin{matrix}\text{S}\\ \text{ON}\\ \text{OFF}\end{matrix}\ \begin{matrix}\\ \text{RPM}\\ \text{SFM}\end{matrix}\ \begin{matrix}\\ \text{CLW}\\ \text{CCLW}\end{matrix}, \text{RANGE}, \begin{matrix}\text{HIGH}\\ \text{MEDIUM}\\ \text{LOW}\\ \text{n}\end{matrix}, \$\text{MAXRPM, M,} \\ \\ \end{array}\right.$$

$$\left.\begin{array}{l} \\ \begin{matrix}\text{XCOORD}\\ \text{RADIUS, YCOORD, MAXIPM, u}\\ \text{ZCOORD}\\ \text{r}\end{matrix} \\ \\ \end{array}\right\}$$

spindle statement for all possible entries and stores appropriate data for use by subroutine PRODA1 where feeds and spindle speeds are calculated. It also controls the spindle (ON, OFF, clockwise or counterclockwise direction) through "M" numbers in the part program. These numbers are outputted as previous "M" numbers via subroutines "LSTMNU", "DKIN", and "PUNMNU". Control parameters that are transferred to subroutine "PRODA1" are given in the discussion that follows.

RPM: Integer SFMRPM is set to 0 as a flag to tell "PRODA1" that "SPNSPE" contains a value for revolutions per minute for the spindle.

SFM: Integer SFMRPM is set to 1 as a flag to tell "PRODA1" that "SPNSPE" contains surface feet per minute for the spindle.

RANGE: Integer RANGE is set to 0, 1, or 2 for following "LOW", "MEDIUM", or "HIGH", or is set to n if a floating point number follows. This information is used by the spindle code output routines.

MAXRPM: Real MAXRPM is set equal to "M" which always follows this identifier. This number is used as a limiting value in "PRODA1".

RADIUS: Notifies of data to follow for "PRODA1". Integer "CONRAD" is set to 0, 1, or 2 for following "XCOORD", "YCOORD", or "ZCOORD", and to 4 if the data following "RADIUS" is real (this real value is stored in "REAL"). This information is used in feed calculations when it is desired to machine at constant SFM values.

MAXIPM: The real number following this identifier is stored in MAXIPM and is used by subroutine PRODA1 as a limiting value.

STOP Subclass 2 - When this command is detected, an "M00" is inserted in the tool path as was other "M" codes using the "LSTMNU", "DKIN", and "PUNMNU" subroutines. This routine starts at (U) of Figure A-3 (Section 14).

TURROT Subclass 1033 - This statement is recognized, but not processed by the software. Data in this record are bypassed using subroutine "BYPARC". Refer to (U1) of Figure A-3 (Section 20).

Subclass Not Recognized - When all possible Subclass 1 statements have been processed and none are recognized, an error message "XXXPP5XXX" is listed on the printing output device, the record is bypassed, and the next record is processed.

Type 2 Record Processing - Type 2 records (Figure A-4) contain cutter centerline data. Their format is shown in Appendix B. These data are processed by first outputting a record number to the printing device using subroutine "OTRCNU". A check is made to determine if the following data are from an APT "FROM" statement symbolizing the tool starting point. When this point is detected, three UPL real numbers that follow are converted to DEC's floating format (subroutine "UPLFOR") for double-precision real numbers and stored in "FRMXCO", "FRMYCO", and "FRMZCO". These are used by "PRODA1" for calculation of incremental movements. If the data are not "FROM" information, they are treated as tool-path data and stored in "DSTXCO", "DSTYCO", and "DSTZCO" as double-precision real numbers. Subroutine PRODA1 is now entered where ΔX, ΔY, and ΔZ (DLTXCO, DLTYCO, and DLTZCO), spindle speed, and feedrate are calculated from previously inputted control parameters. The incremental movements of ΔX and ΔY are plotted on a CRT using software interpolation in subroutine "CRTOUT". Incremental data are converted to the controller format by first converting delta to ASCII strings via

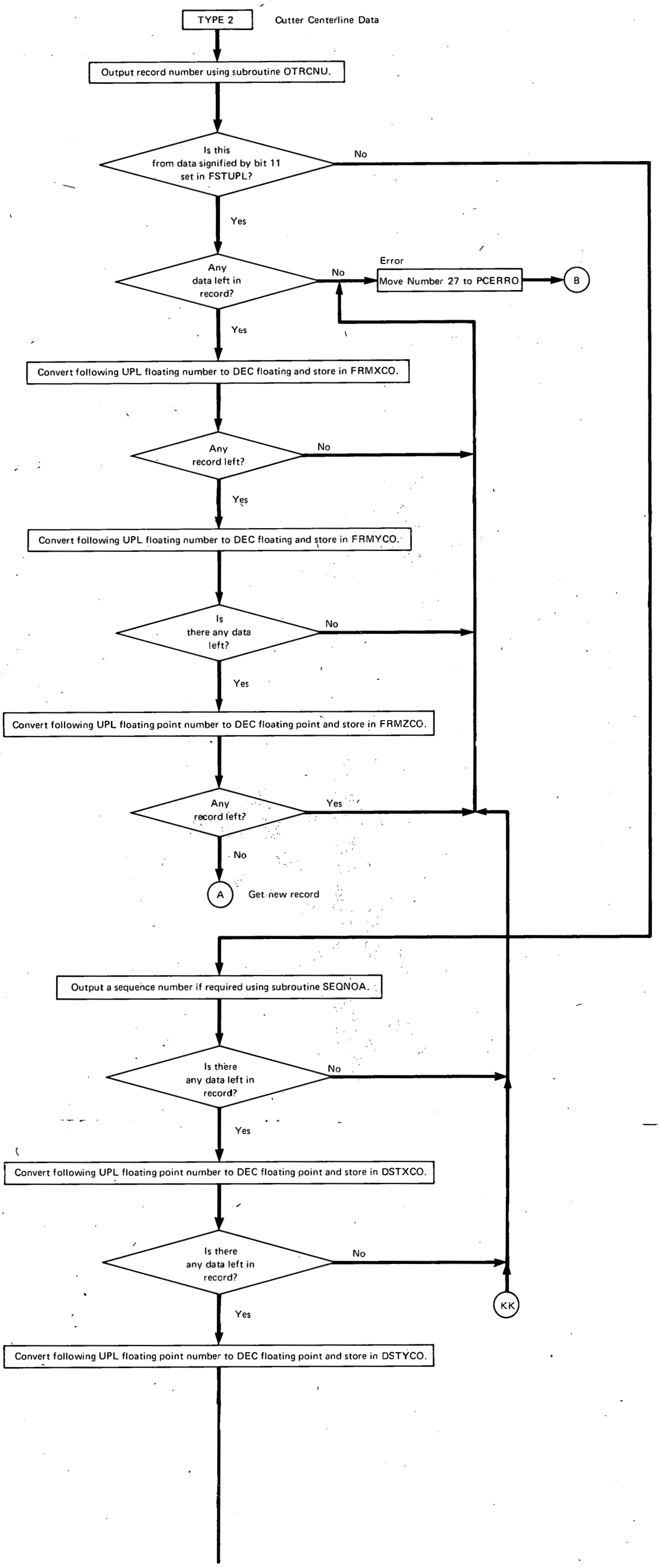


Figure A-4. TYPE 2 RECORD PROCESSING SOFTWARE FLOWGRAPH. (Section 1)

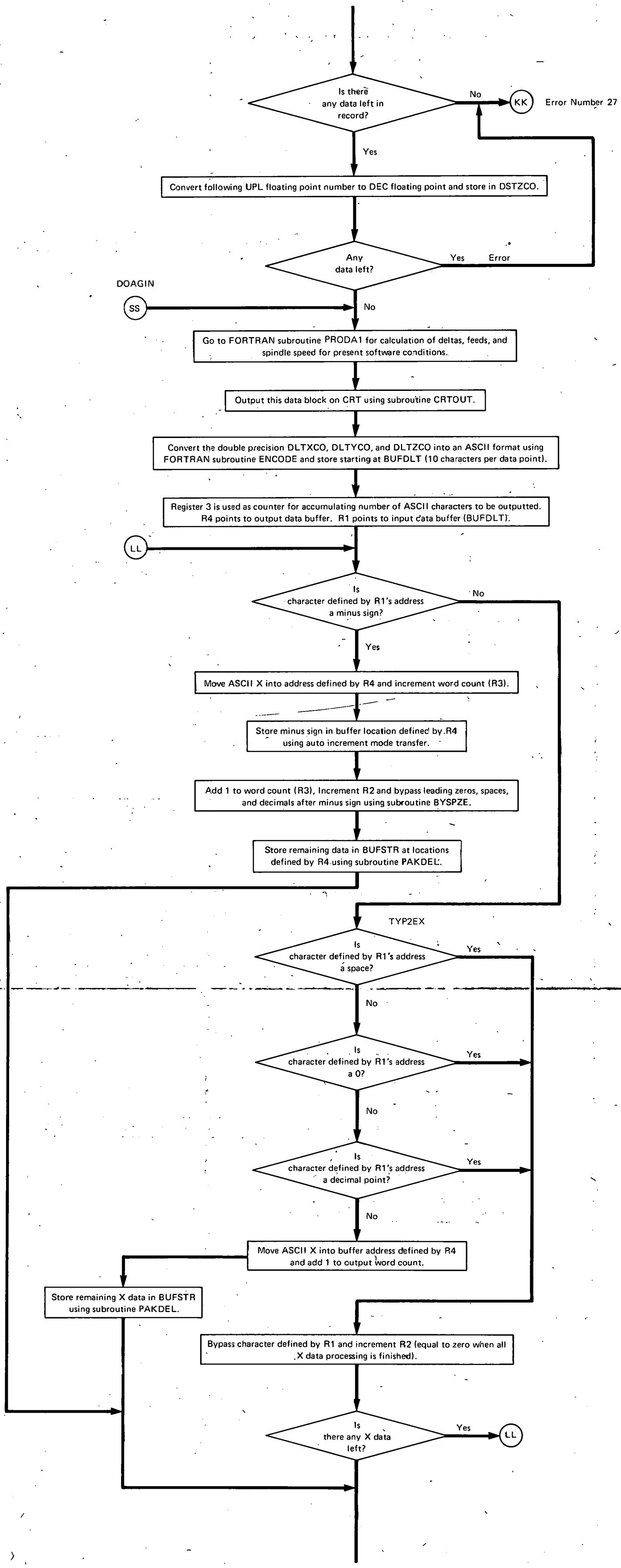


Figure A-4. (Section 2)

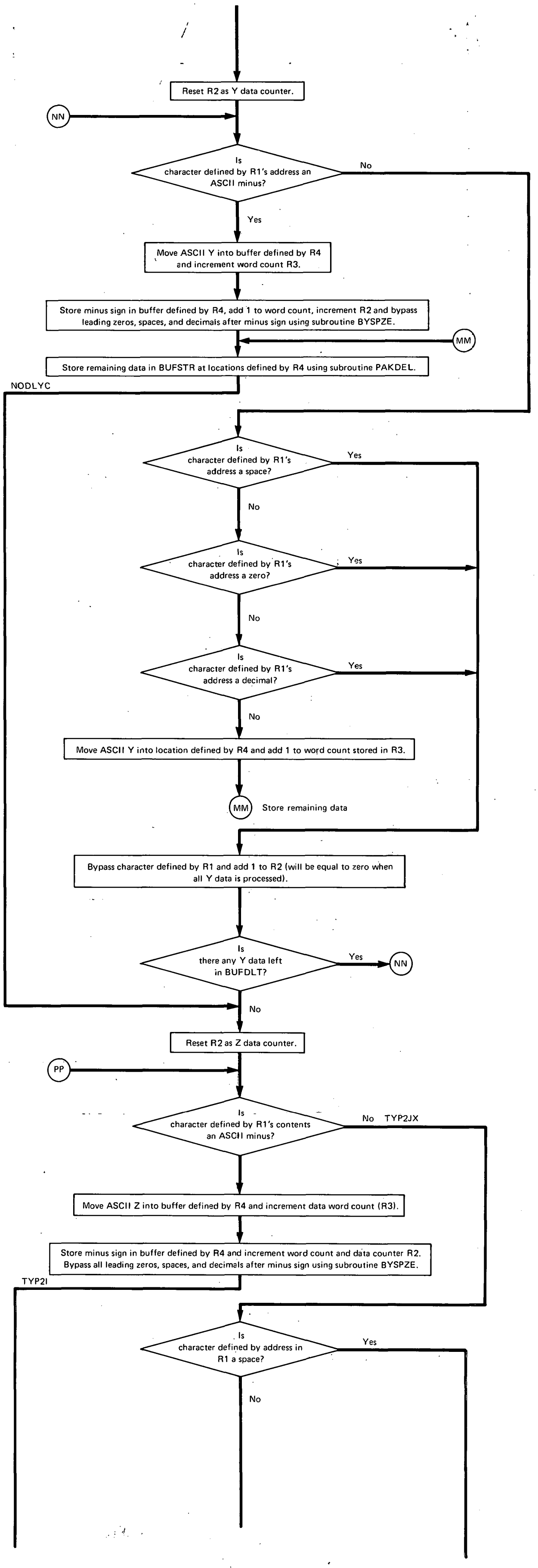


Figure A-4. (Section 3)

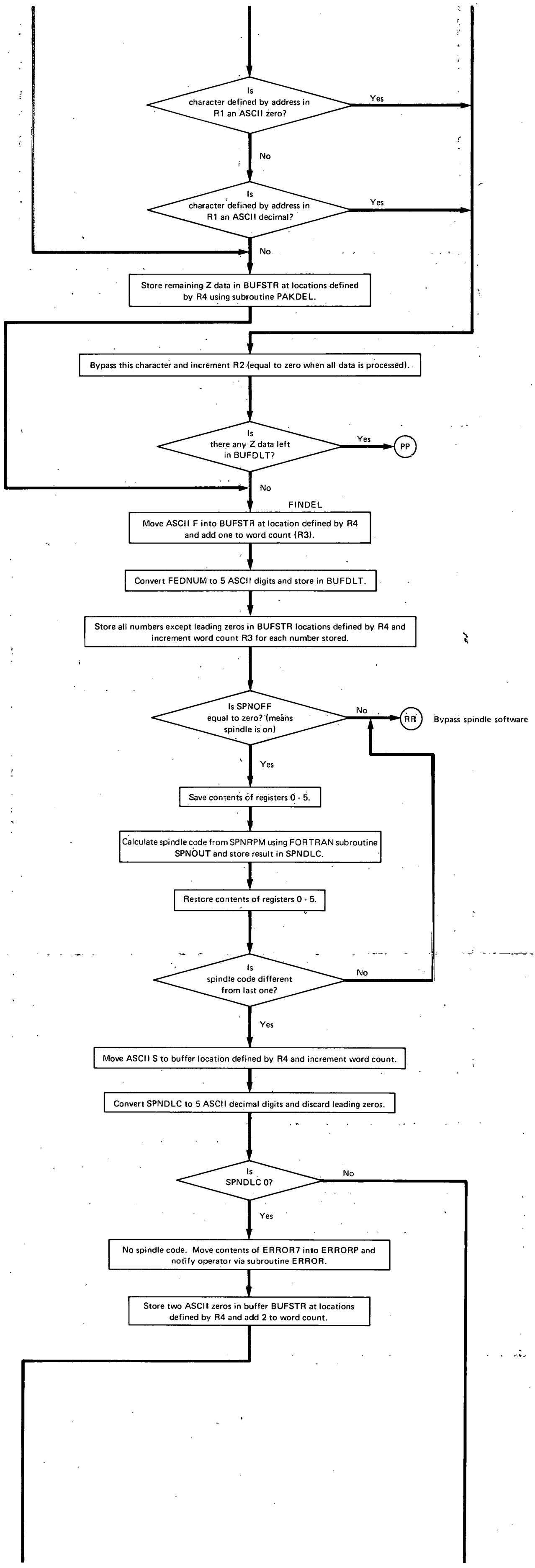


Figure A-4. (Section 4)

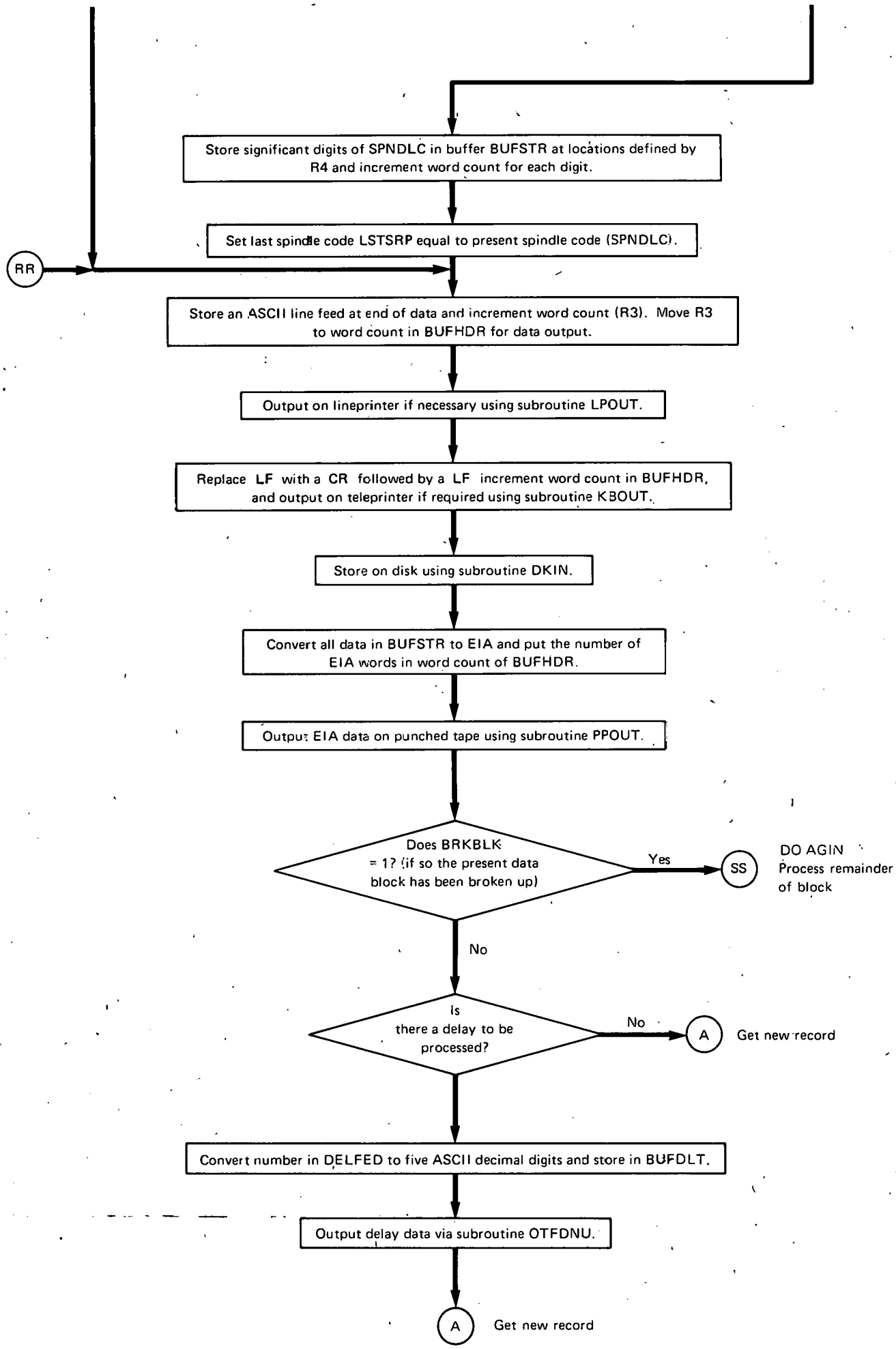


Figure A-4. (Section 5)

FORTRAN subroutine "ENCODE", removing spaces and periods, bypassing leading zeros, and inserting dimensional preamble characters (X, Y, or Z). The preamble character for feednumber is inserted after these data followed by the ASCII value of "FEDNUM" (minus leading zeros) which was calculated by "PRODA1". If the spindle is on, a spindle code is calculated using FORTRAN subroutine "SPNOUT". If the spindle code is different from the last one outputted, its ASCII value is inserted in the data stream following feednumbers. The data are then outputted to enabled devices (lineprinter, teleprinter, disk, or punched tape). Subroutine "PRODA1" will break an incremental movement into several blocks if a feedrate variation greater than 5% is detected from starting to end point. When this situation occurs, a flag is set (BRKBLK = 1) that tells the postprocessor to reenter "PRODA1" to continue processing this record, otherwise a new record is retrieved.

Type 3 and Type 4 Records - No Type 3 and 4 records exist for the DDNC or Bendix 1800 without circular interpolation. When detected, an error "XXXPP9XXX" is outputted to the listing device, and the record is bypassed using the "BYPARC" subroutine. A flowgraph is given in Figure A-5.

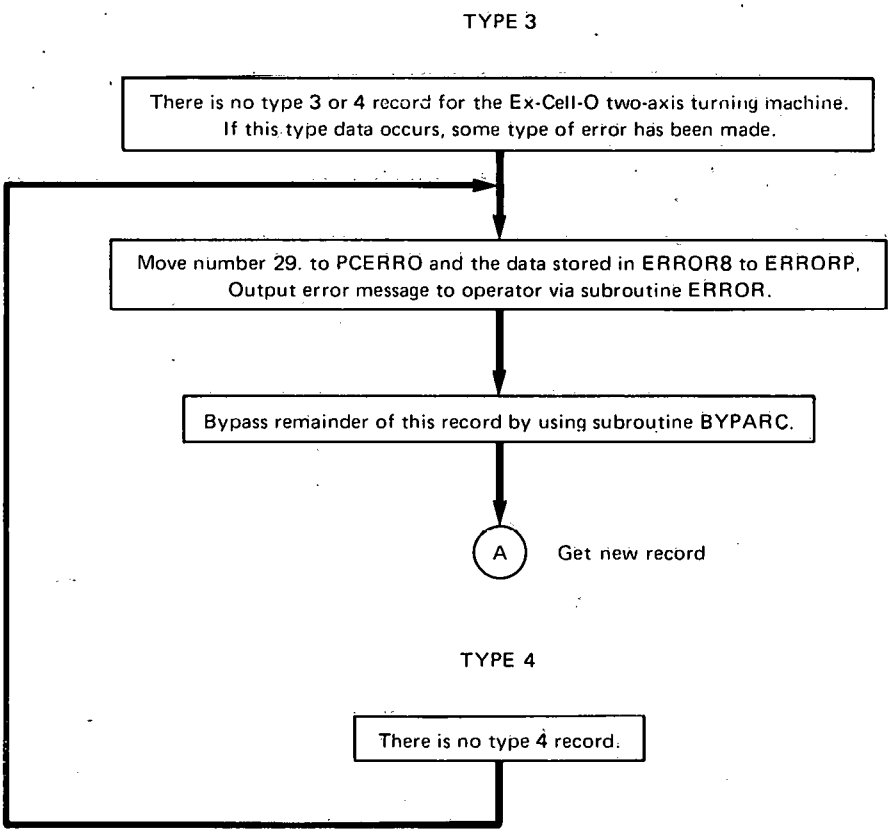


Figure A-5. FLOWGRAPH FOR TYPES 3 AND 4 RECORDS.

Subroutines Used by the Postprocessor

Subroutine BYPARC - This subroutine (Figure A-6) is used to bypass all or part of a record that is not recognized or contains some type of error.

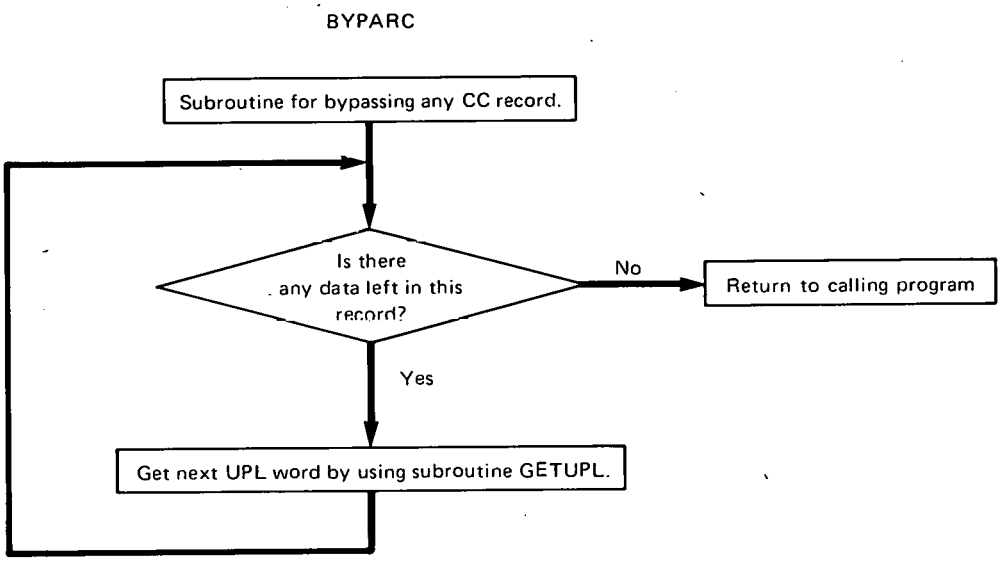


Figure A-6. SUBROUTINE FOR BYPASSING A UNITED PROGRAMMING LANGUAGE RECORD.

Subroutine BYSPZE - ASCII character strings from subroutine ENCODE can include spaces, leading zeros, and decimals. These characters are not allowed in the object data. This subroutine (Figure A-7) bypasses these characters (R1's contents is the address of data in question) when the object data block is being formatted.

Subroutine OTFDNU - This subroutine (Figure A-8) removes leading zeros from the ASCII data (feednumber) in buffer "BUFDLT", transfers the data preceded by an ASCII F into "BUFSTR" buffer, and outputs the data to the enabled peripheral devices.

Subroutines ERROR and ERRORA - RAD50 data stored in "ASTERR" and "ERRORP" are unpacked and stored in buffer "BUFSTR" in ASCII code as ******PPn******. These data are followed by a space and the ASCII octal equivalent of the number stored in "PCERRO" (a listing of errors and their description is given in Table A-4 and outputted to the lineprinter. Subroutine ERRORA is used to notify the operator of DOS-related errors where hardware errors are detected when transferring data from disk, DECtape, or CPU. (Refer to Figure A-9 for the flow diagram).

Subroutine LPOUT - This routine (Figure A-10) lists the contents of buffer BUFSTR on the lineprinter. It is enabled by inserting a "NOP" at "LPOUT" or disabled by a "RTS R5" at this location. The amount and type of data to be transferred are defined by a header block starting at "BUFHDR".

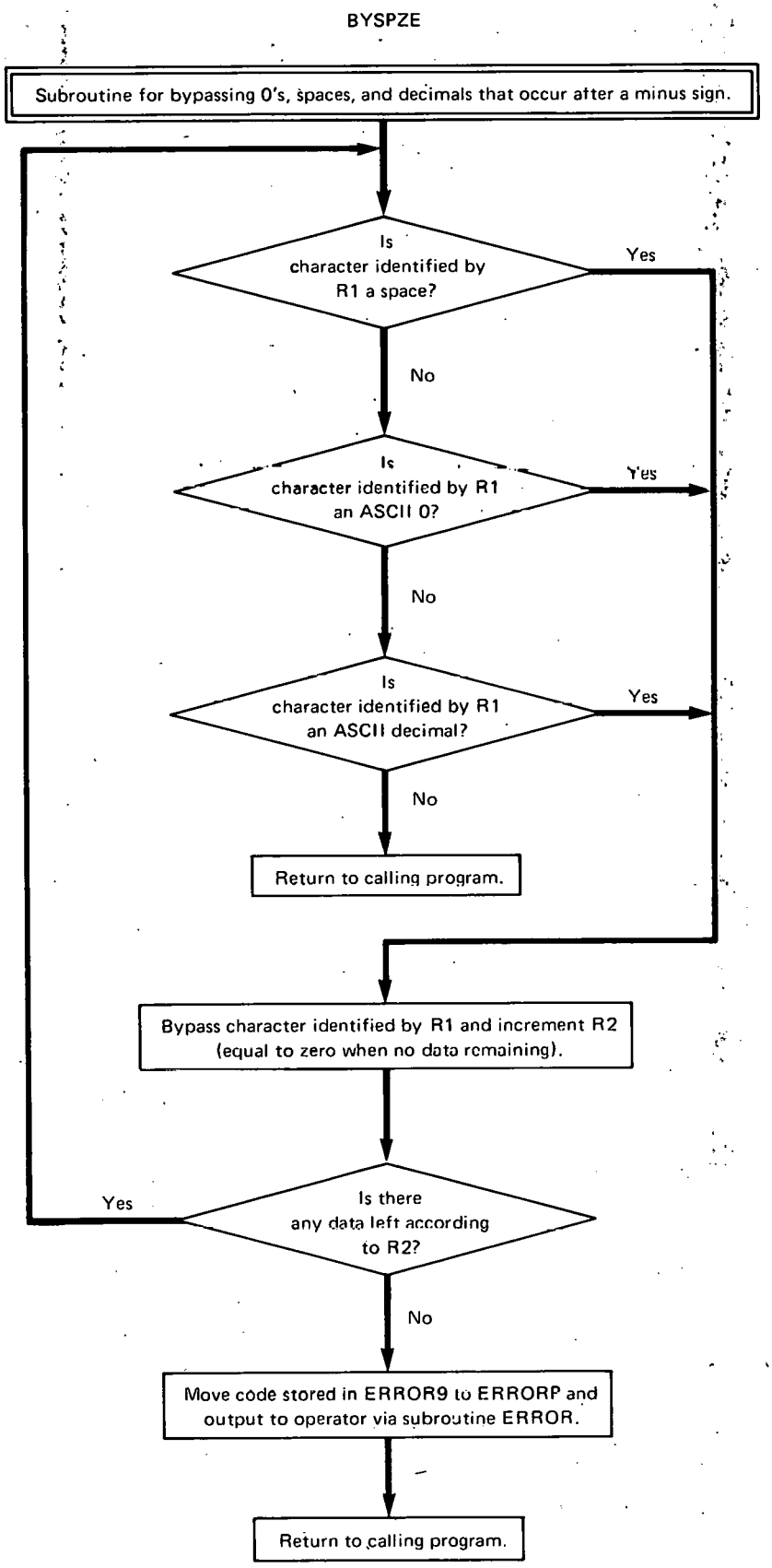


Figure A-7. SUBROUTINE FOR BYPASSING SPACES, ZEROS, AND DECIMALS.

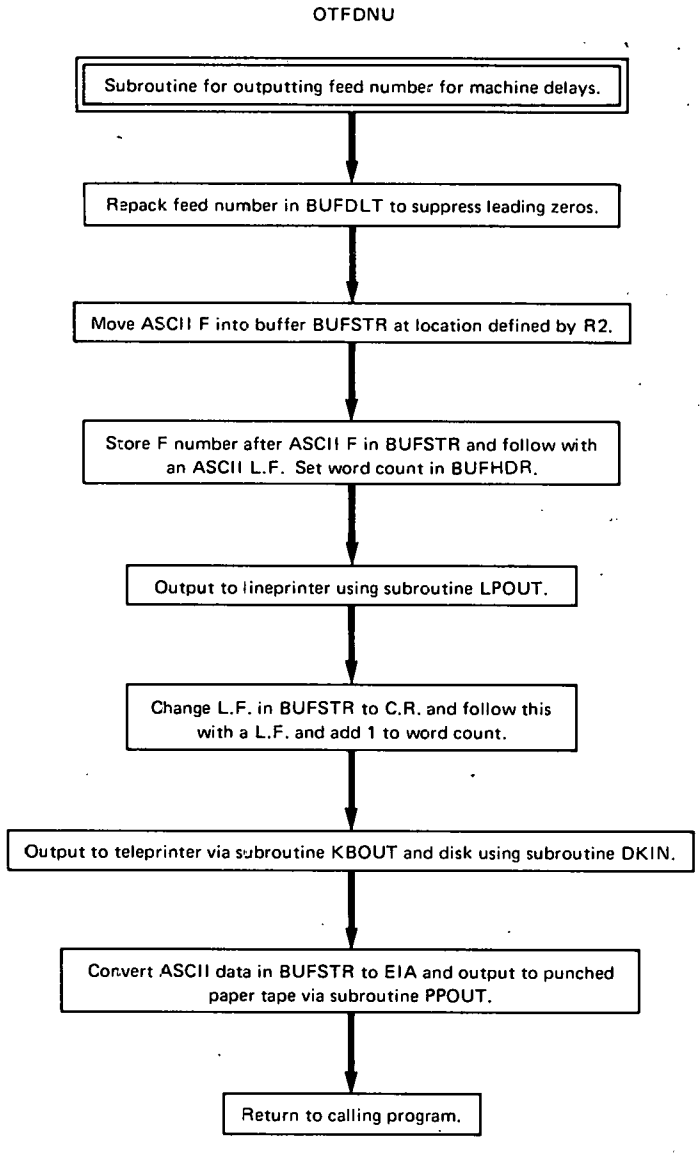


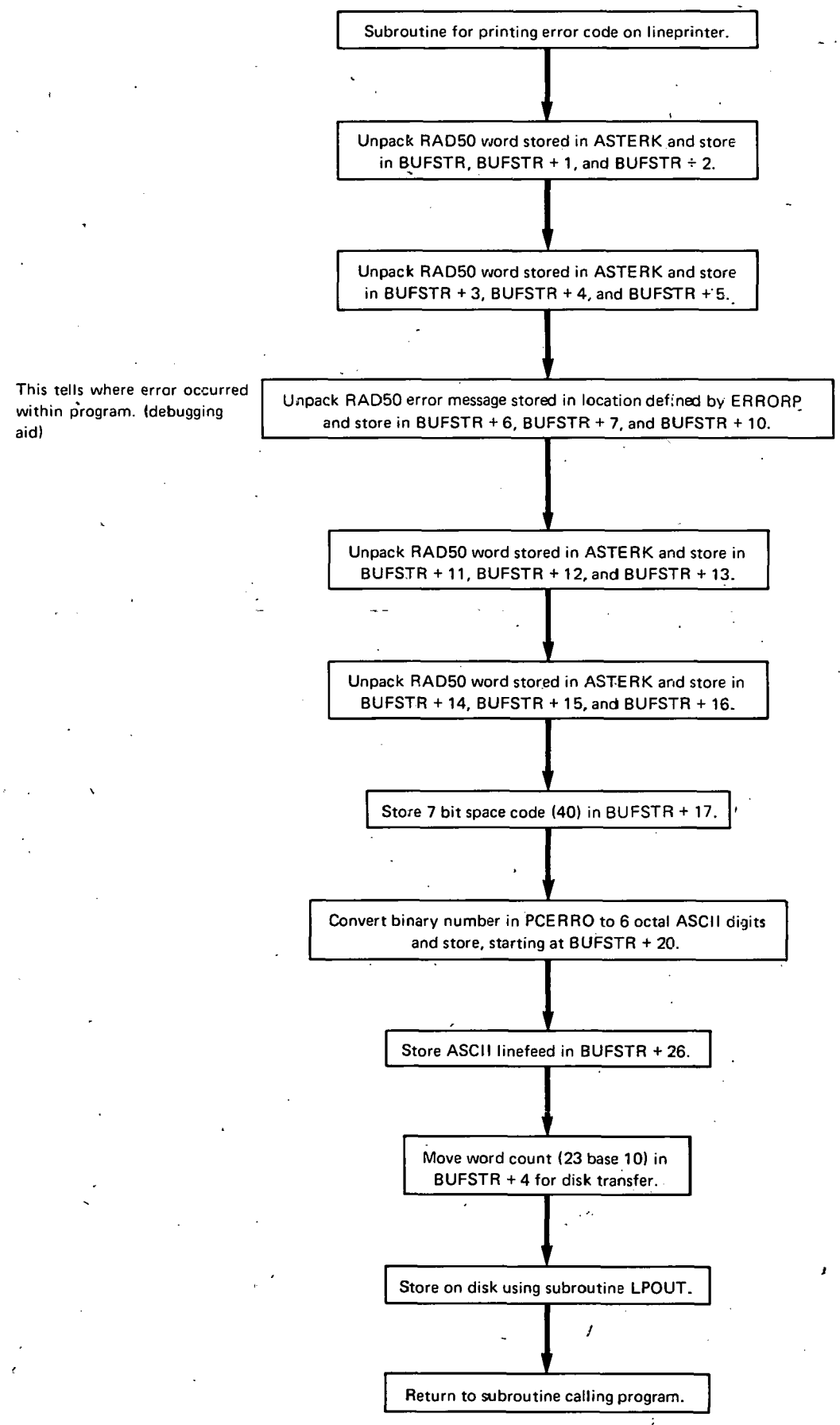
Figure A-8. SUBROUTINE FOR OUTPUTTING THE DELAY FEED NUMBER.

Table A-4

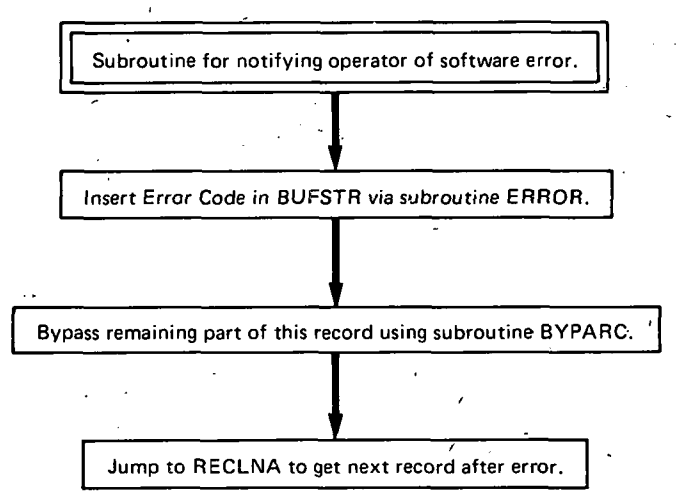
ERRORS DETECTED BY THE POSTPROCESSOR
******PPn****** ii

Error PPn	ii	Description
PP0	-	Software error code (DOS related).
PP1	-	Record too long.
PP2	2	Format error at TYP1A
	3	Format error at STY3AA.
	4	Format error in COOLNT statement at COOLNT.
	5	Format error in COOLNT statement at COOLNT2.
	6	Format error in DELAY at STYBBB.
	7	Format error in DISPLY at STY3CC.
	8	Format error in DISPLY at STY3C5.
	9	Format error in FEDRAT at STY3EY.
	10	Format error in FEDRAT at STY3E3.
	11	Format error in FEDRAT at STY3E8.
	12	Format error in FEDRAT at STY3E9.
	13	Format error in FEDRAT at STY3EX.
	14	Format error in LEADER at STY3F1.
	15	Format error in LEADER at STY3F4.
	16	Format error in MACHIN at NOPYP1.
	17	Format error in MACHIN at STY36X.
	18	Format error in MACHIN at STY362.
	19	Format error in MACHIN at STY3G4.
	20	Format error in MACHIN at STY3G.
	21	Format error in PARTNO at STY3J2.
	22	Format error in INSERT at STY3K1.
	23	Format error in PPRINT at STY3L1.
	24	Format error in SEQNO at STY301.
	25	Format error in SPINDL at STY3P1.
	26	Format error in STOP at STY3Q1.
	27	Format error in departure data at TYPE2A.
	28	Not a recognized record type at NOTANY.
	30	Format error in AUXFUN at STY3Z2.
	31	Format error in PARTNO at STY3J2.
PP3	-	Postprocessor command not recognized.
PP4	-	Incorrect PARTNO statement
PP5	-	Not used.
PP6	-	Spindle statement error.
PP7	-	Canonical form of circle (no error).
PP8	-	Departure data error (major).
PP9	-	Not used.

ERROR



ERRORA



ERRORB

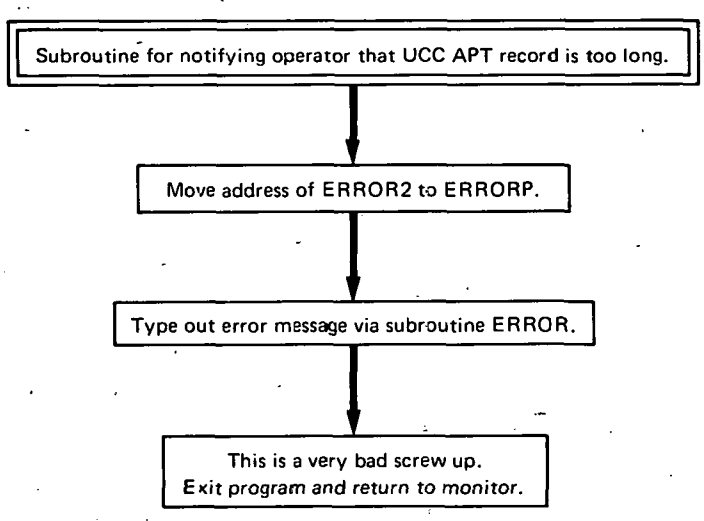


Figure A-9. SUBROUTINES FOR PRINTING ERROR CODES.

Subroutines DKOUT, SUBCLO, and SUBOPE - These routines (Figure A-11) are used to transfer 512 bytes of data from the disk to buffer "BUFFER". Disk control is derived from data blocks at "TRNBLK" and "LNKBLK". DKOUT uses subroutine SUBCLO to close any disk file that might happen to be open (defined by LNKBLK) before data are transferred. This file is reopened by subroutine "SUBOPE" after data transfer (controlled by data block starting at "DKFIBK" and "LNKBLK"). Detailed information on these types of operations can be obtained from DEC manuals on the disk operating system.

Subroutine INITIA - This software (Figure A-12) is entered when the postprocessor is loaded into core memory just prior to the start of data processing. It is used to preset such items as "M" codes and machine parameters, to set subroutines for the teleprinter output for disabling output to line printer, punch, CRT, and disk, and to bring into memory the software drivers for all peripheral devices that can be used.

Subroutine ASCEIA - This subroutine (Figure A-13) converts the ASCII byte stored in "ASCII" to EIA code and stores the converted data in "EIA". This task is accomplished by searching a table starting at "A" for the ASCII character in question. When found, the following byte will be the equivalent EIA code.

Subroutine KBOUT - Routine KBOUT (Figure A-14) outputs the contents of buffer "BUFSTR" to the teleprinter. Data transfer is controlled by control blocks starting at "BUFHDR" and"KBLNBK" and is disabled or enabled by inserting a "RTS RS" or "NOP" at "KBOUT".

Subroutine PPOUT - Routine "PPOUT" (Figure A-15) operates as "KBOUT" but uses data control block "PPLNBK" instead of "KBLNBK" for the "WRITE" statement.

Subroutine PUNLED - This routine punches a two-inch leader on paper tape consisting of 20 EIA zeros. A flowgraph is given in Figure A-16.

Subroutine OTRCNU - This routine (Figure A-17) converts the binary record numbers to octal ASCII data and outputs to either the lineprinter or teleprinter.

Subroutines GETUPL and ENOFBU - A UPL data word stored in the 512-byte buffer (BUFFER) is transferred to "FSTUPL" and "SNDUPL" for interrogation by calling software. When the last word of the buffer is detected, the area is refilled with new data using subroutine "DKOUT". A flowgraph is presented in Figure A-18.

Subroutine LSTMNU - RAD50 data stored in "MNUPNT" are converted to ASCII and stored in buffer "BUFSTR" where it is outputted to either the lineprinter or teleprinter. A flowgraph is given in Figure A-19.

Subroutine PUNMNU - The RAD50 M number stored in "MNUPNT" is first converted to three ASCII characters, then to EIA code using subroutine "ASCEIA", then punched on paper tape when subroutine "PPOUT" is enabled. Figure A-20 is a flowgraph of the software.

BUFSTR

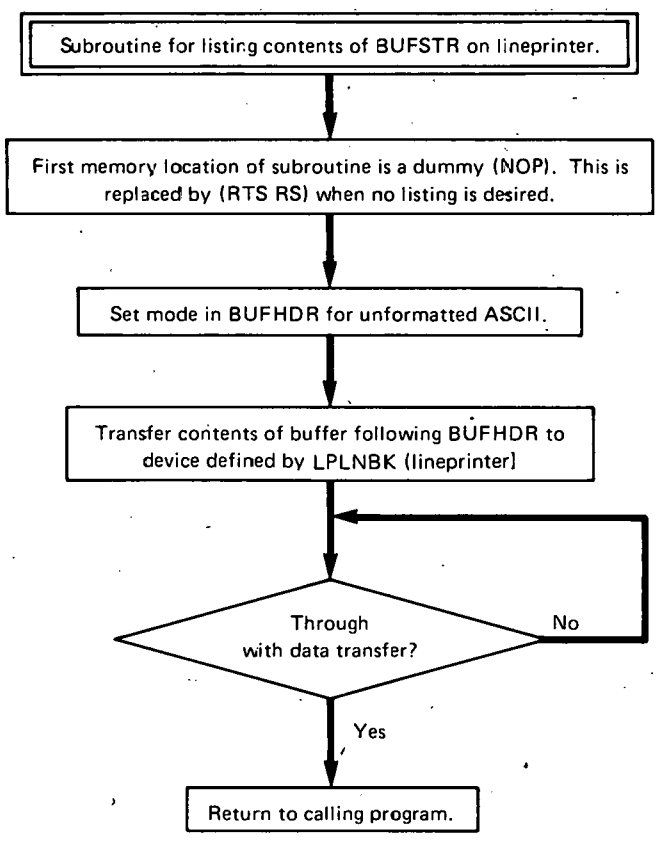
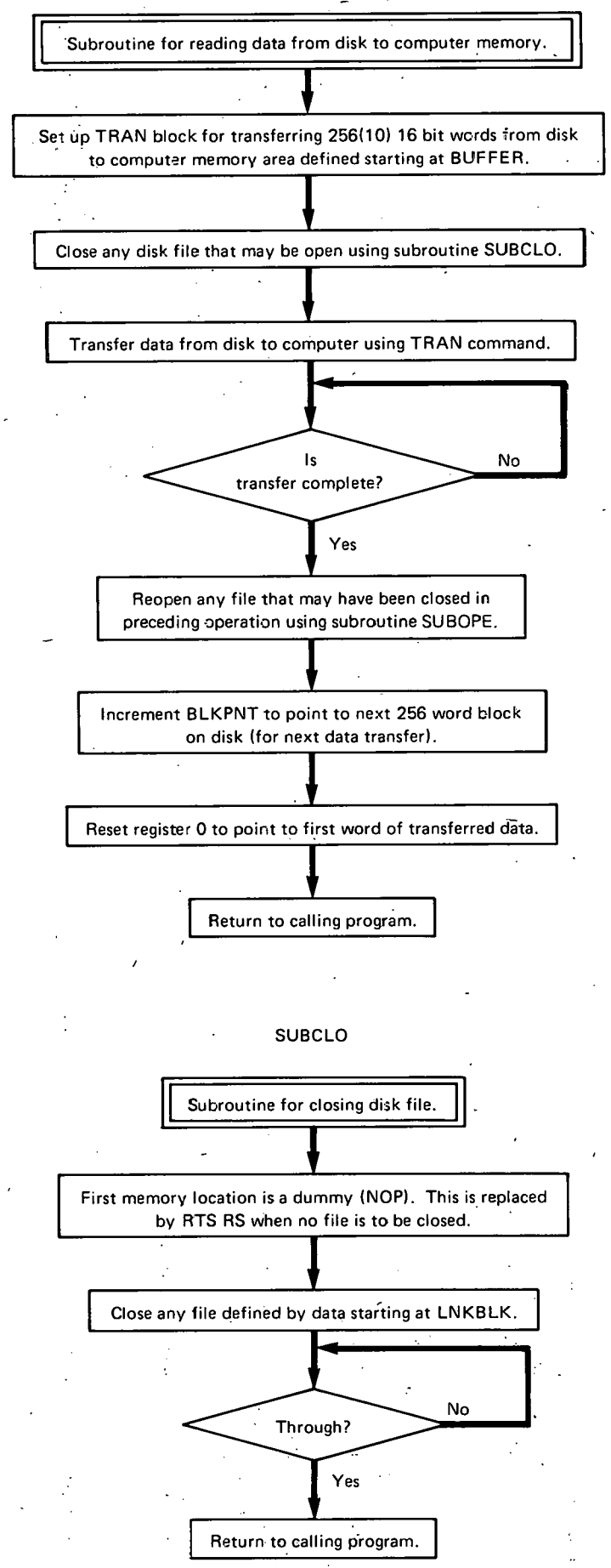


Figure A-10. SUBROUTINE FOR LISTING THE CONTENTS OF BUFSTR ON THE LINEPRINTER.

DKOUT



SUBCLO

SUBOPE

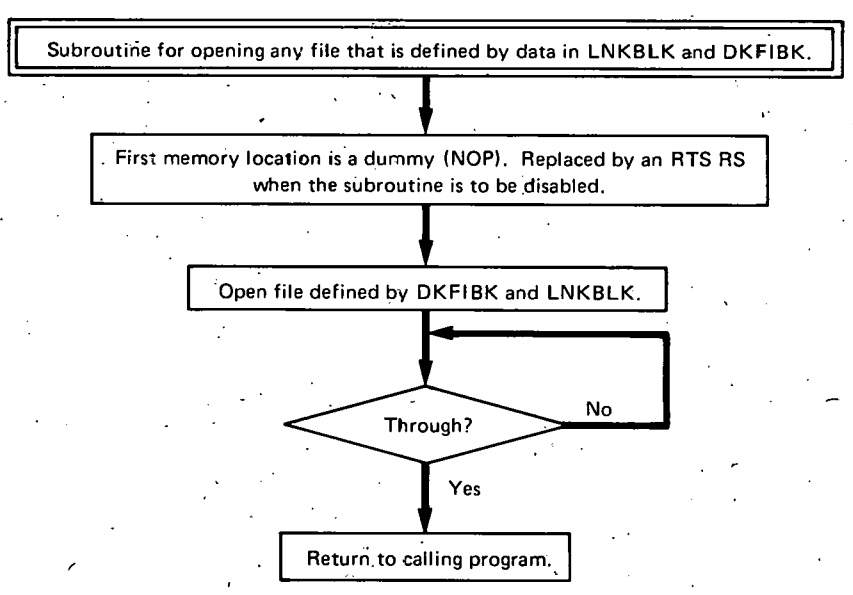


Figure A-11. SUBROUTINES FOR TRANSFERRING DATA FROM THE DISK TO MEMORY.

INITIA

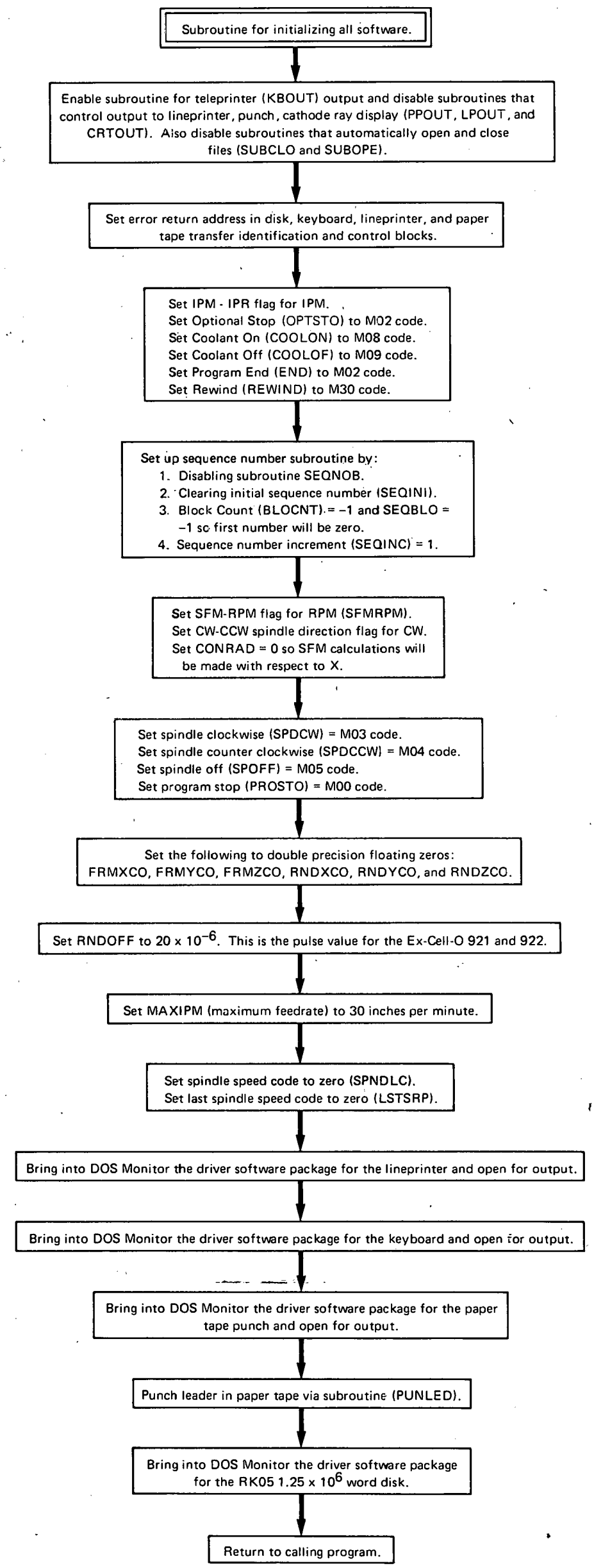


Figure A-12. ROUTINES FOR INITIALIZATION FOR DDNCs OR BENDIX 1800s.

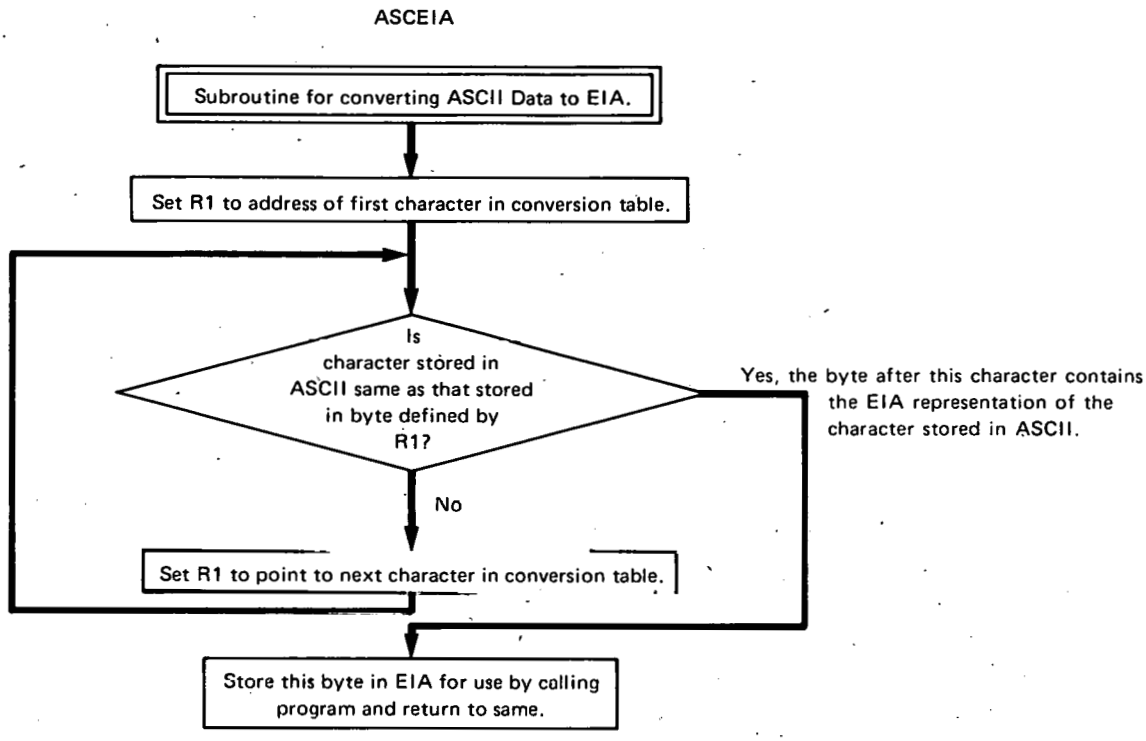


Figure A-13. ASCII-TO-EIA CORE CONVERTER.

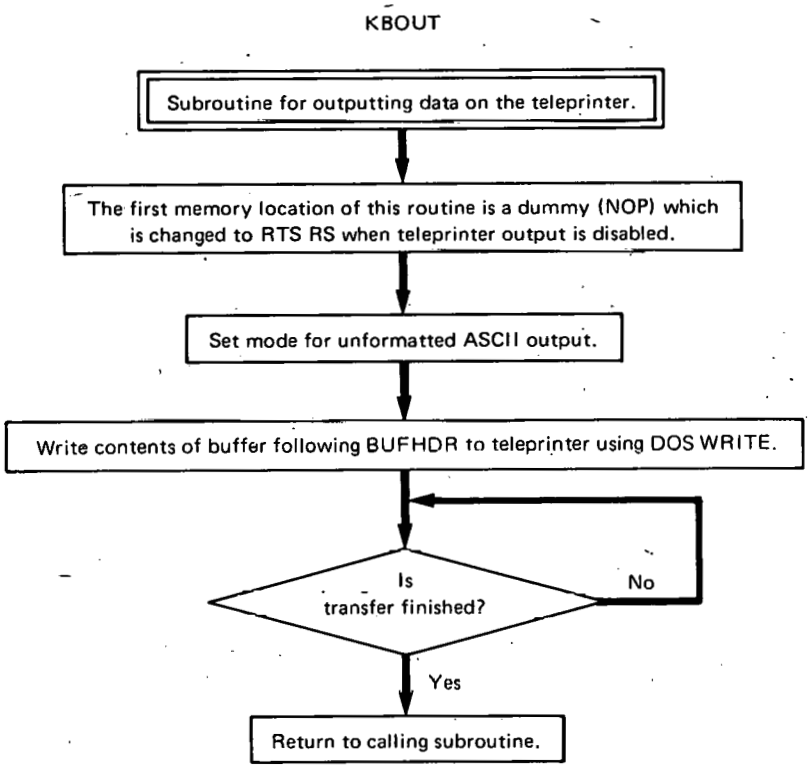


Figure A-14. ROUTINE FOR OUTPUTTING THE CONTENTS OF BUFSTR TO THE TELEPRINTER.

PPOUT

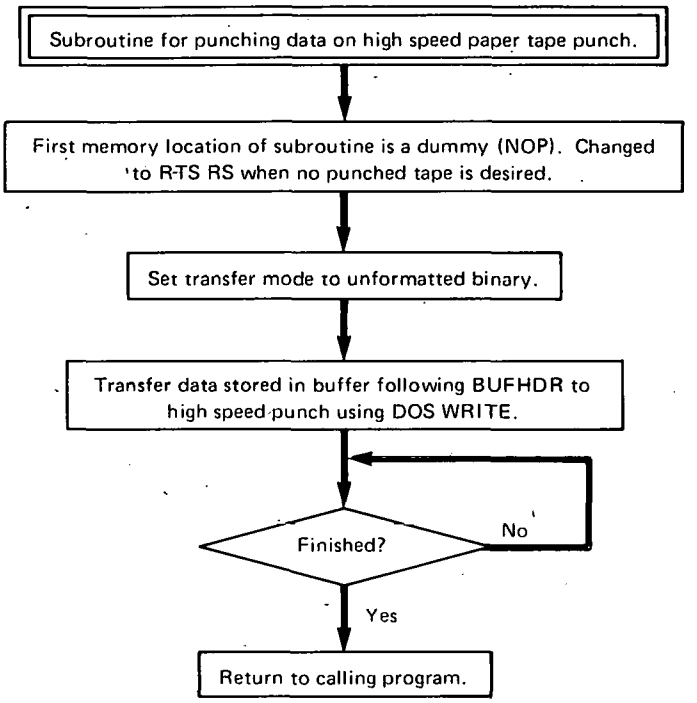


Figure A-15. SOFTWARE FOR OUTPUTTING DATA TO THE PAPER TAPE PUNCH.

PUNLED

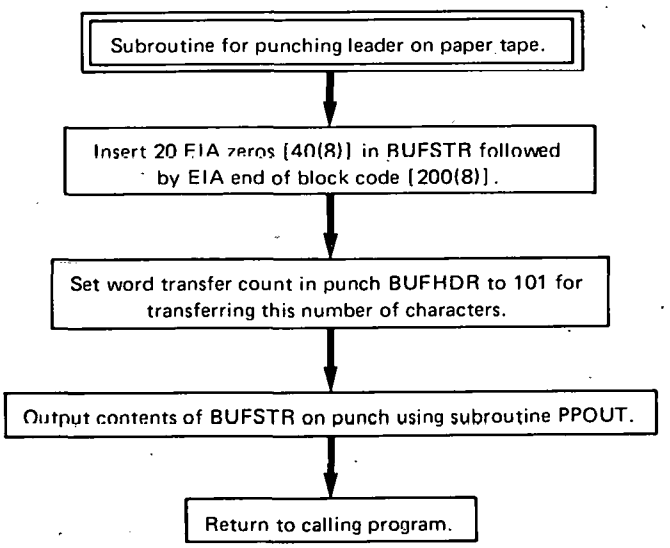


Figure A-16. ROUTINE FOR PUNCHING THE LEADER ON THE PAPER TAPE.

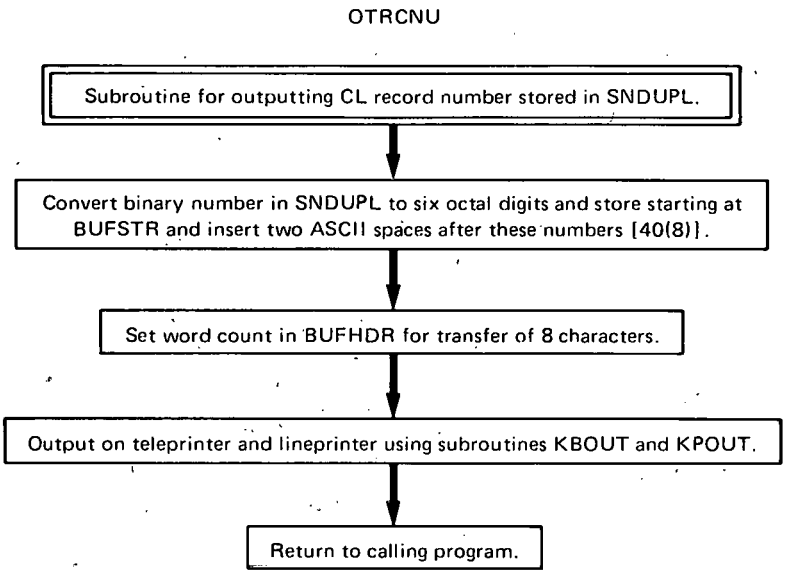
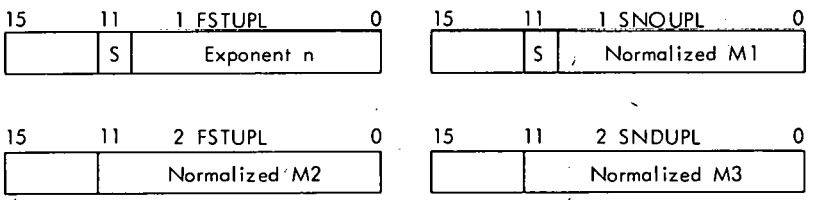
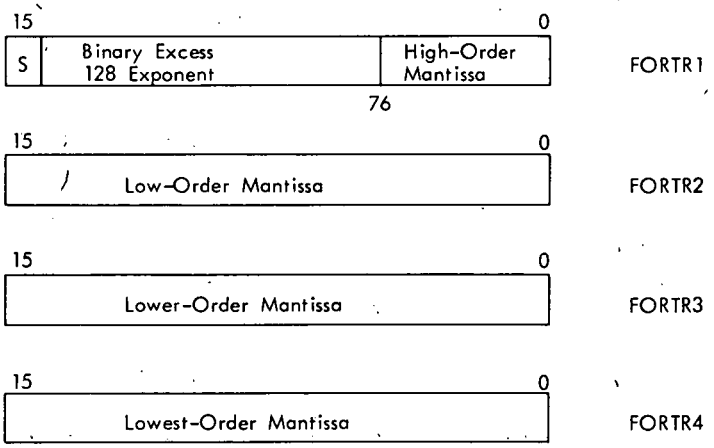


Figure A-17. ROUTINE FOR OUTPUTTING THE RECORD NUMBER.

Subroutine UPLFOR - UPL's floating point format is as follows:



where bits 12 - 15 are not used. This format must be converted to DEC's FORTRAN double-precision floating-point format which is as follows:



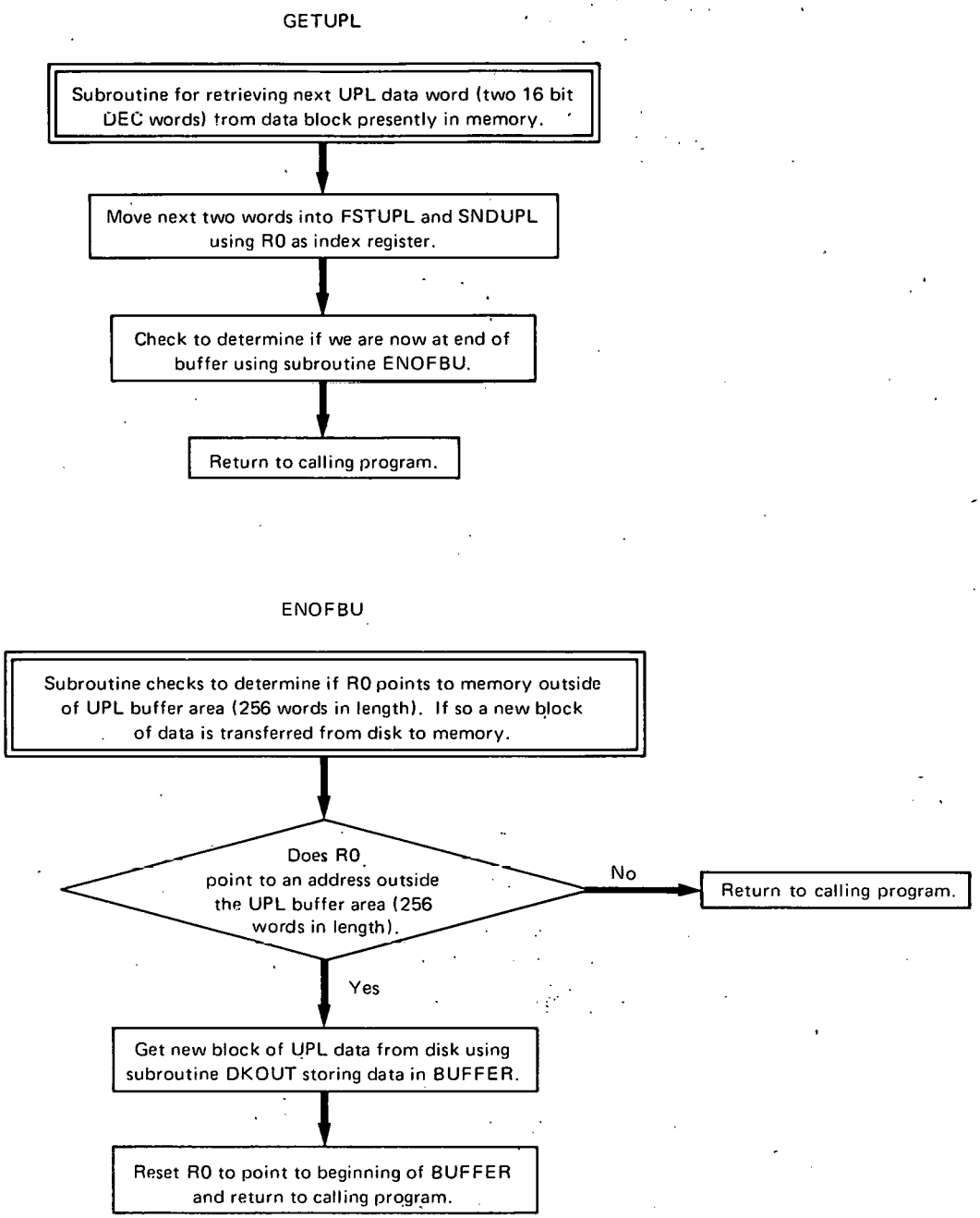


Figure A-18. ROUTINE FOR RETRIEVING TWO DEC WORDS FROM BUFFER.

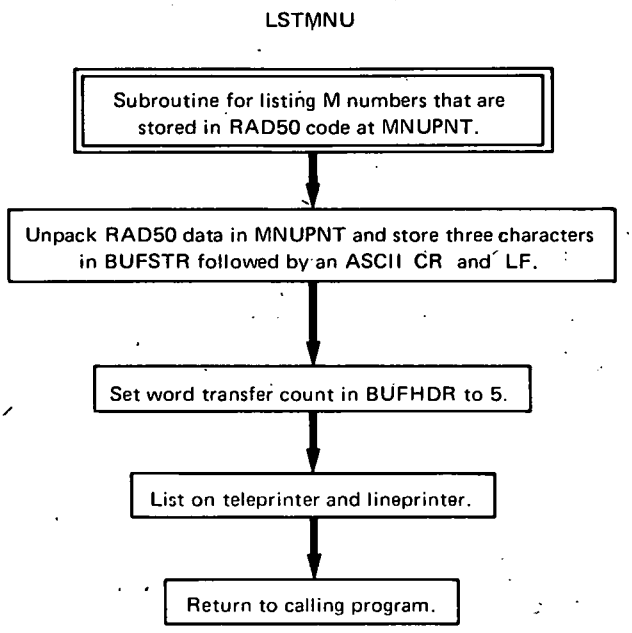


Figure A-19. SOFTWARE FOR LISTING M CODES ON THE LINEPRINTER OR TELEPRINTER.

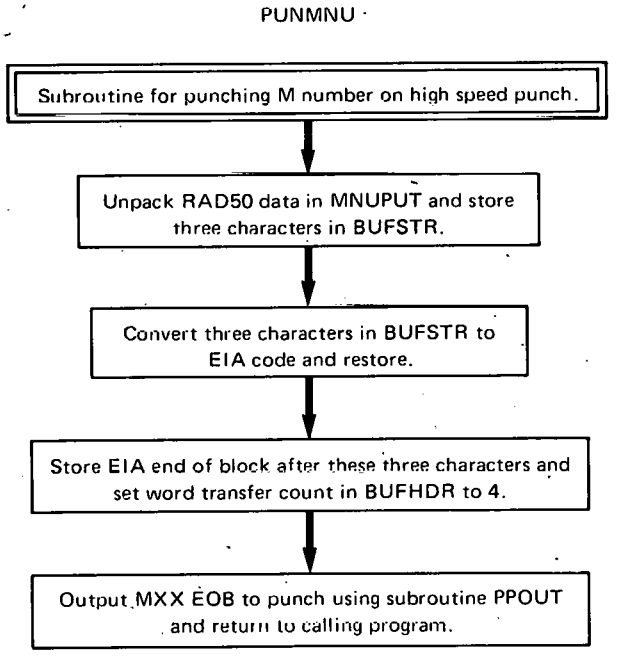


Figure A-20. ROUTINE FOR PUNCHING M CODES ON THE HIGH-SPEED PUNCH.

Since the higher-order bit of the DEC mantissa is always a binary "1", it is not included. Subroutine UPLFOR converts the UPL format to DECs for use by routines generated by DEC's FORTRAN compiler. A flow diagram of the subroutine is seen in Figure A-21.

Subroutine DKIN - Data stored in buffer "BUFSTR" is written on disk using the DOS "WRITE" statement. The "WRITE" statement uses information in "LNKBLK" and "BUFHDR" for controlling data flow and defining the desired output peripheral (DISK). A flow diagram of the routine is shown in Figure A-22.

Subroutine CRTOUT - The postprocessor's tool-path data are drawn on a cathode ray tube (CRT) oscilloscope by interpolating the departure data with software and accumulating control signals in two software registers which are continually transferred to digital-to-analog converters (DAC) whose outputs drive the X and Y axes of the CRT. The CRT is treated as a two-axis machine tool with 5.860×10^{-3} inch of movement per control pulse. Interpolation is accomplished by the following method:

1. The distance of each axis travel is converted to control pulses and stored in two software registers as negative numbers which are counted down to zero.

2. Numbers are calculated and inserted into velocity registers such that when counted down at the same rate with overflows used to increment respective departure registers, the departure registers will reach their destination simultaneously.

3. A loop is entered where the two velocity registers are incremented each time through. As each register reaches zero, its respective departure register is incremented and the original contents of the velocity registers are restored. This process is continued until both departure registers contain zeros.

All math is performed by FORTRAN subroutine "CALCRT". A flow diagram of "CRTOUT" is provided in Figure A-23.

Subroutine LEADER - This routine (Figure A-24) punches lists and stores the number of EIA or ASCII zeros defined by "LENLED". "LENLED" is normally set by multiplying the number following the APT "LEADER" statement by ten (ten characters per inch of punched tape).

Subroutine CARSTI - The APT statements "PPRINT" and "INSERT" are followed by 66 ASCII characters of data which generally contain instructions for manufacturing the part or error messages. This routine transfers these data to buffer "BUFSTR" where it is outputted to the teleprinter or lineprinter. A flow diagram of the subroutine is given in Figure A-25.

Subroutines SEQNOA and SEQNOB - These routines are used to output "N" followed by a three-digit number to the lineprinter, teleprinter, disk, and paper tape punch. The number is controlled by the APT statement "SEQNO" which sets the following constants:

BLOCNT - set equal to the number of blocks between each sequence number.
SEQINI - initial value of the sequence number.
SEQINC - the amount to change the sequence number for each output.

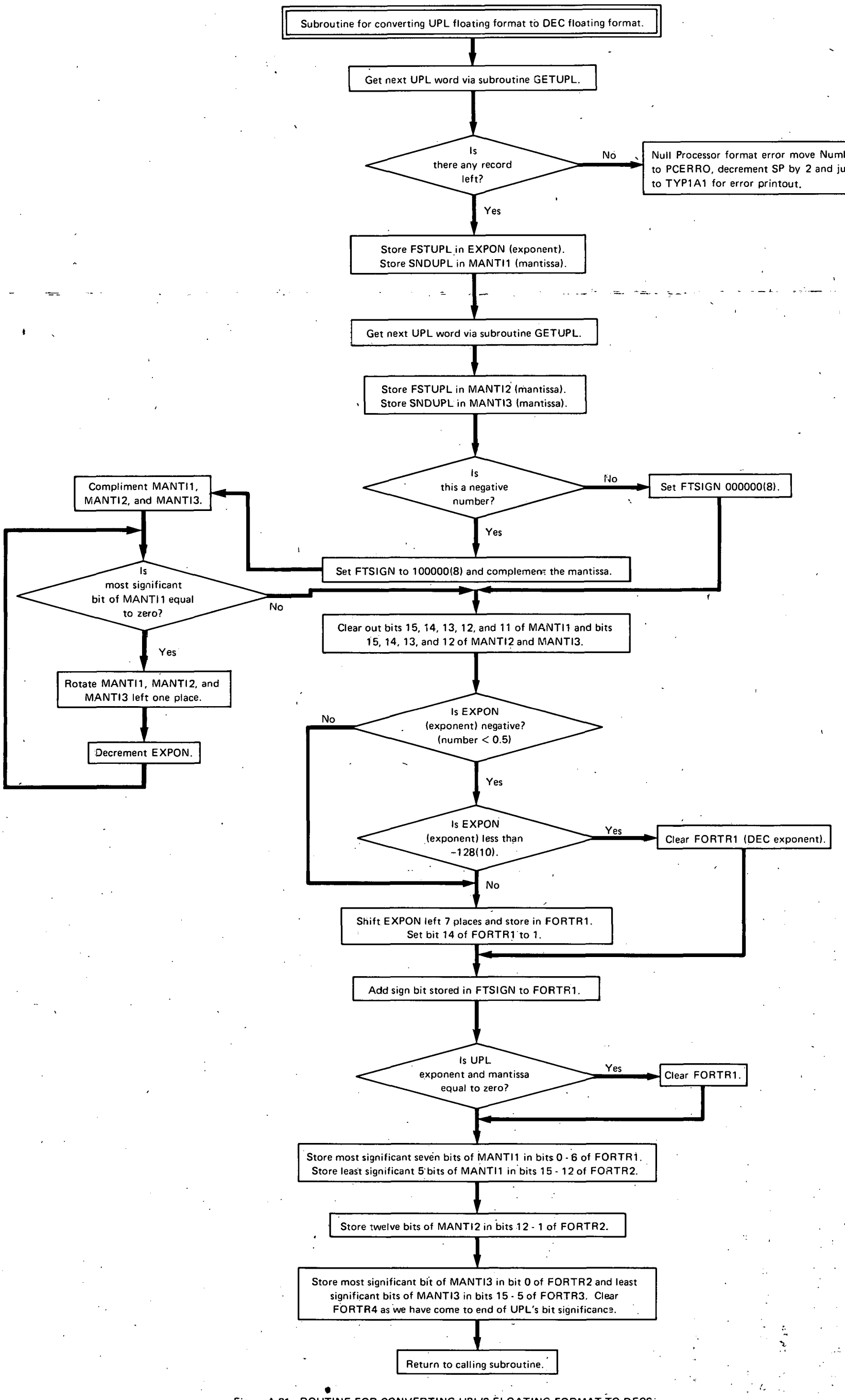


Figure A-21. ROUTINE FOR CONVERTING UPL'S FLOATING FORMAT TO DECS.

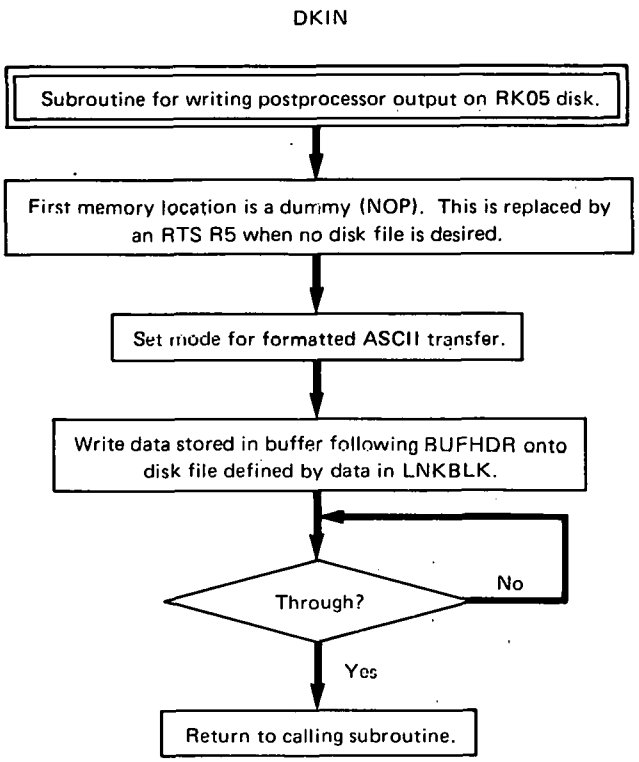


Figure A-22. ROUTINE FOR WRITING POSTPROCESSOR OUTPUT ON THE RK05 DISK.

Normally "SEQNOA" is called first which, in turn, calls "SEQNOB' . A flow diagram is seen in Figure A-26.

Subroutine PAKDEL - This routine (Figure A-27) is used to store departure (bypassing spaces and decimals) for output. R1 is used as a data retrieve pointer for buffer "BUFDLT", R2 as a -10 counter for detecting the end of data stream, R3 as a data storage counter, and R4 as a data storage pointer for buffer "BUFSTR".

Subroutine INSRAN - The spindle controller on the Ex-Cell-O turning machine must receive a correct range code (S20, S40, S60, or S80) before the M code that turns the spindle on. This subroutine (Figure A-28) performs this function. It must be called before any M03 or M04 is outputted for correct system operation.

Subroutine SPINDL - This subroutine (Figure A-29) calculates the spindle code values from the range (RANGE) and RPM (SPNRPM) entries for a previously calculated data block. It contains four least-squares polynomial third-order curves, giving spindle code as a function of RPM for each of the four allowed gear ranges. These curves were developed from a table of RPM-versus-spindle codes provided by the machine-controller manufacturer. The standard error for the least-squares curve is 0.28 in the calculated code value over a span of code values from 0 - 999. In practice, the calculated RPM may fall outside the range of validity. In such a case, the RPM is set to the limiting value. For example, if the range entry implies

CRTOUT

Subroutine for drawing postprocessed part on CRT screen. This routine is a software interpolator that drives two digital-to-analog converters.

Calculate constants for software interpolation using FORTRAN subroutine CALCRT. These are XVELOC, YVELOC, XDISTA, YDISTA, XSIGN, and YSIGN.

Is XDISTA = 0 (Is X movement = 0)?

XDONE — Yes

No

Is YDISTA = 0 (Is Y movement = 0)?

Yes — YDONE

No

Add 1 to XVELOC.

Is XVELOC = 0?

No

Yes

Add 1 to XDISTA.

Is X direction positive?

No — Add 1 to XSTREG.

Yes — Subtract 1 from XSTREG.

Is XDISTA = 0 (Is X movement complete?

Yes — XDONE — A

No

Add 1 to YVELOC.

Is YVELOC = 0?

UPDDAC — No

Yes

Add 1 to YDISTA.

Is YDISTA = 0 (is Y movement complete)?

Yes — YDONE — B

Is Y direction positive?

No — Add 1 to YSTREG.

Yes — Subtract 1 from YSTREG.

Move XSTREG into XDAC (CRT X digital-to-analog converter) and YSTREG into YDAC. Intensify the CRT to display data.

XVELOC = Number loaded in X Velocity counter to give 10 ips velocity over X movement.

YVELOC = Number loaded in Y Velocity counter to give 10 ips velocity over Y.

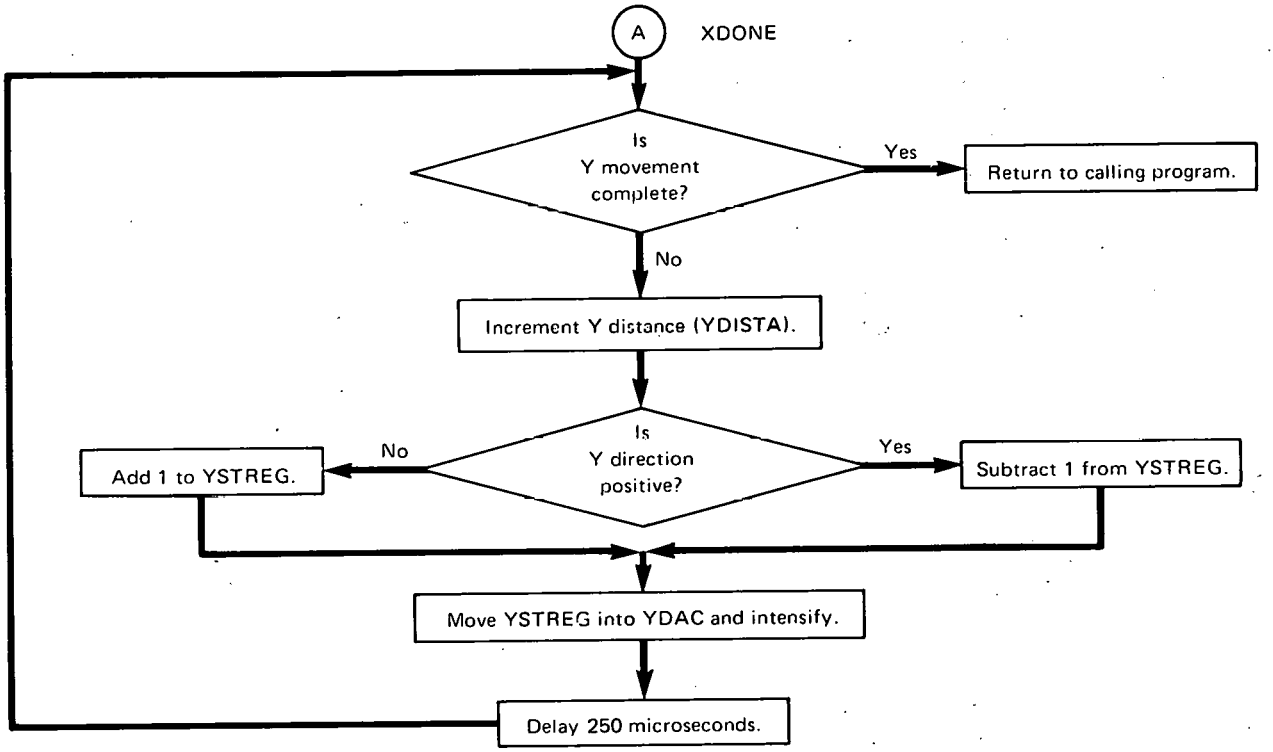
XDISTA = Number of movement pulses to X DAC.

YDISTA = Number of movement pulses to Y DAC.

YSIGN = Direction of Y.

XSIGN = Direction of X.

Figure A-23. SOFTWARE INTERPOLATOR FOR CATHODE-RAY TUBE CONTROL. (Section 1)



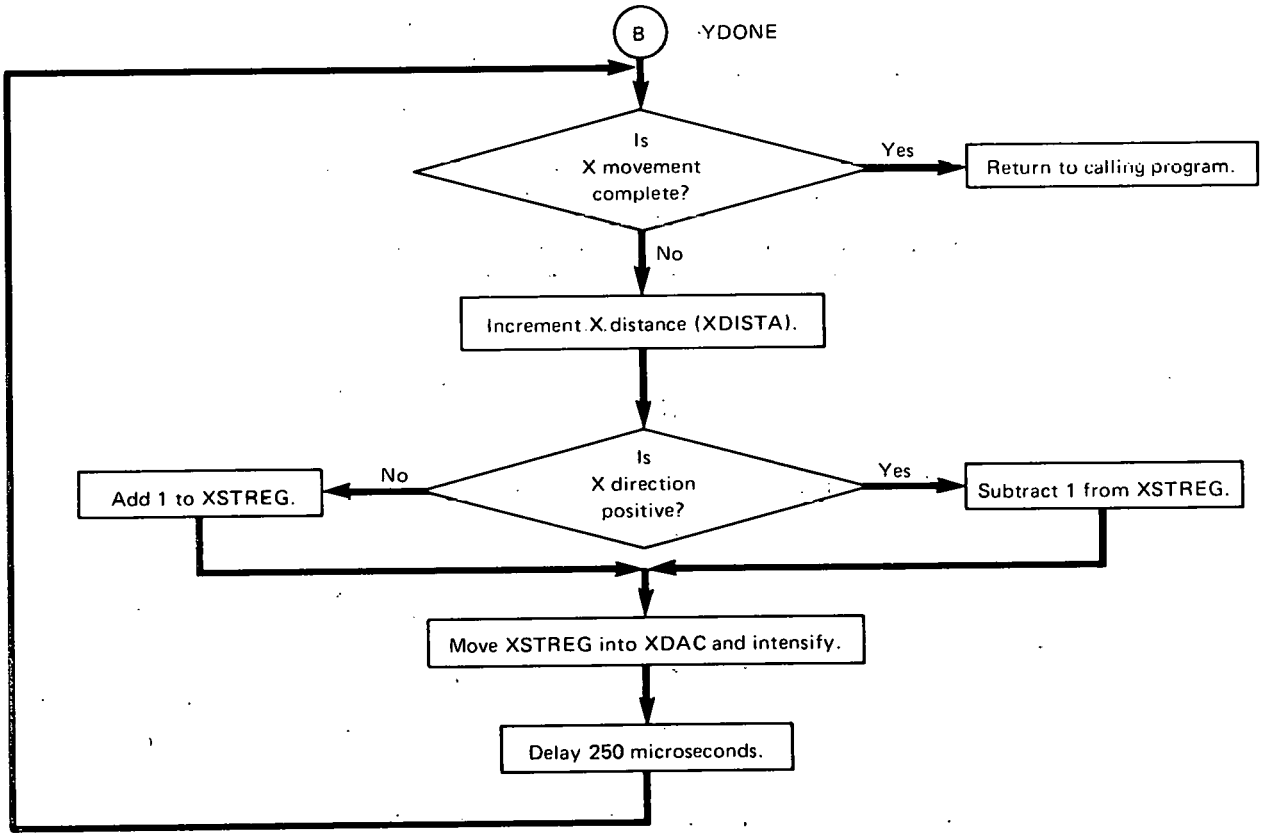


Figure A-23. (Section 2)

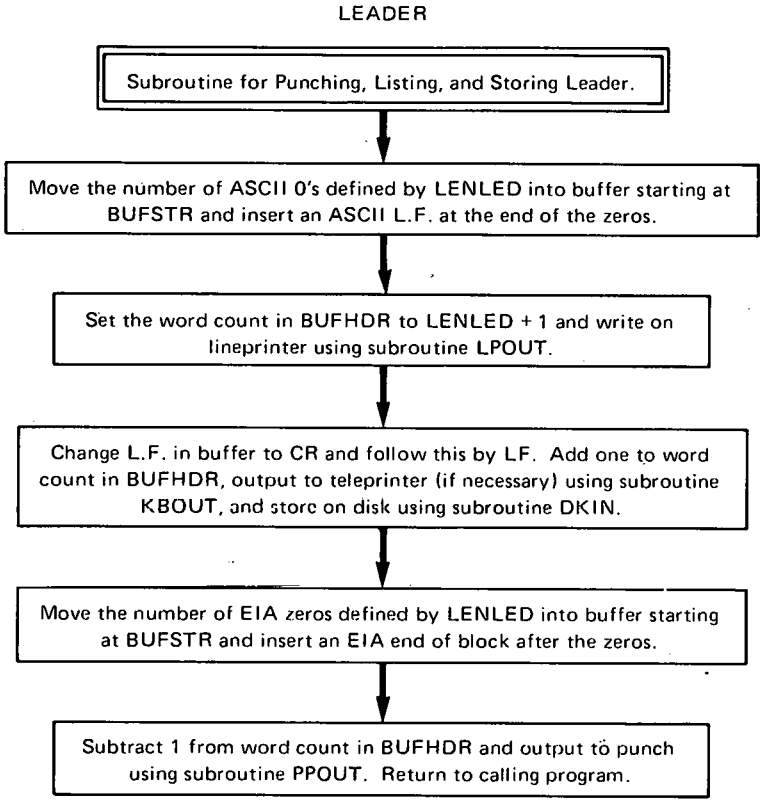


Figure A-24. ROUTINE FOR PUNCHING, LISTING, AND STORING THE LEADER CODE.

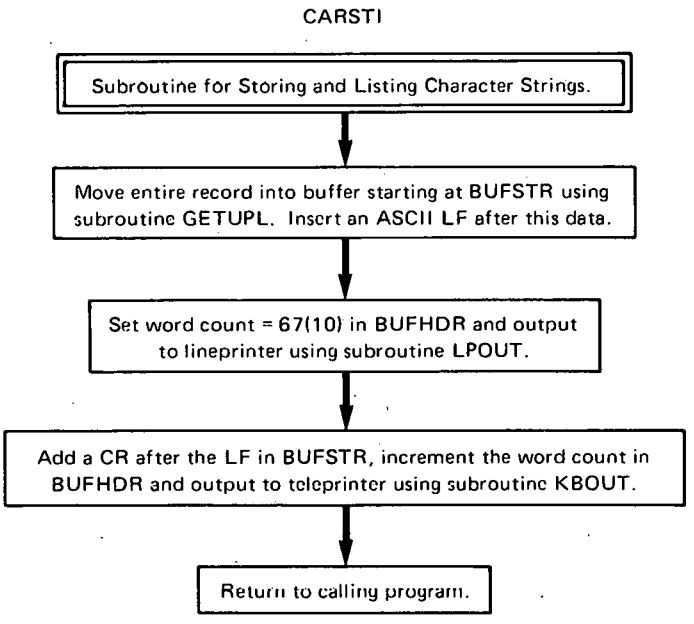


Figure A-25. SUBROUTINE FOR LISTING THE STRING FOLLOWING APT, PPRINT, OR INSERT.

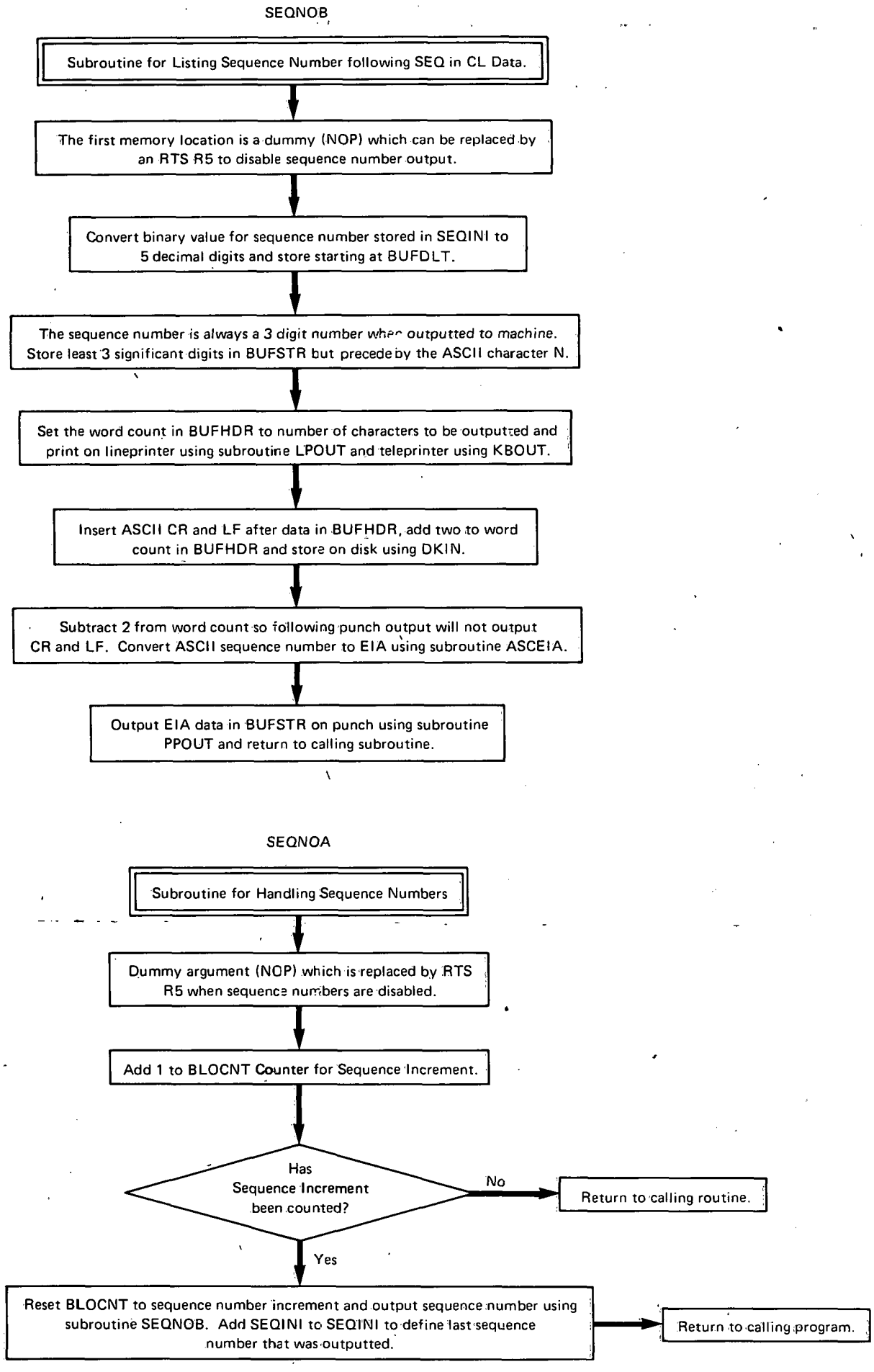


Figure A-26. ROUTINE FOR OUTPUT OF SEQUENCE NUMBERS

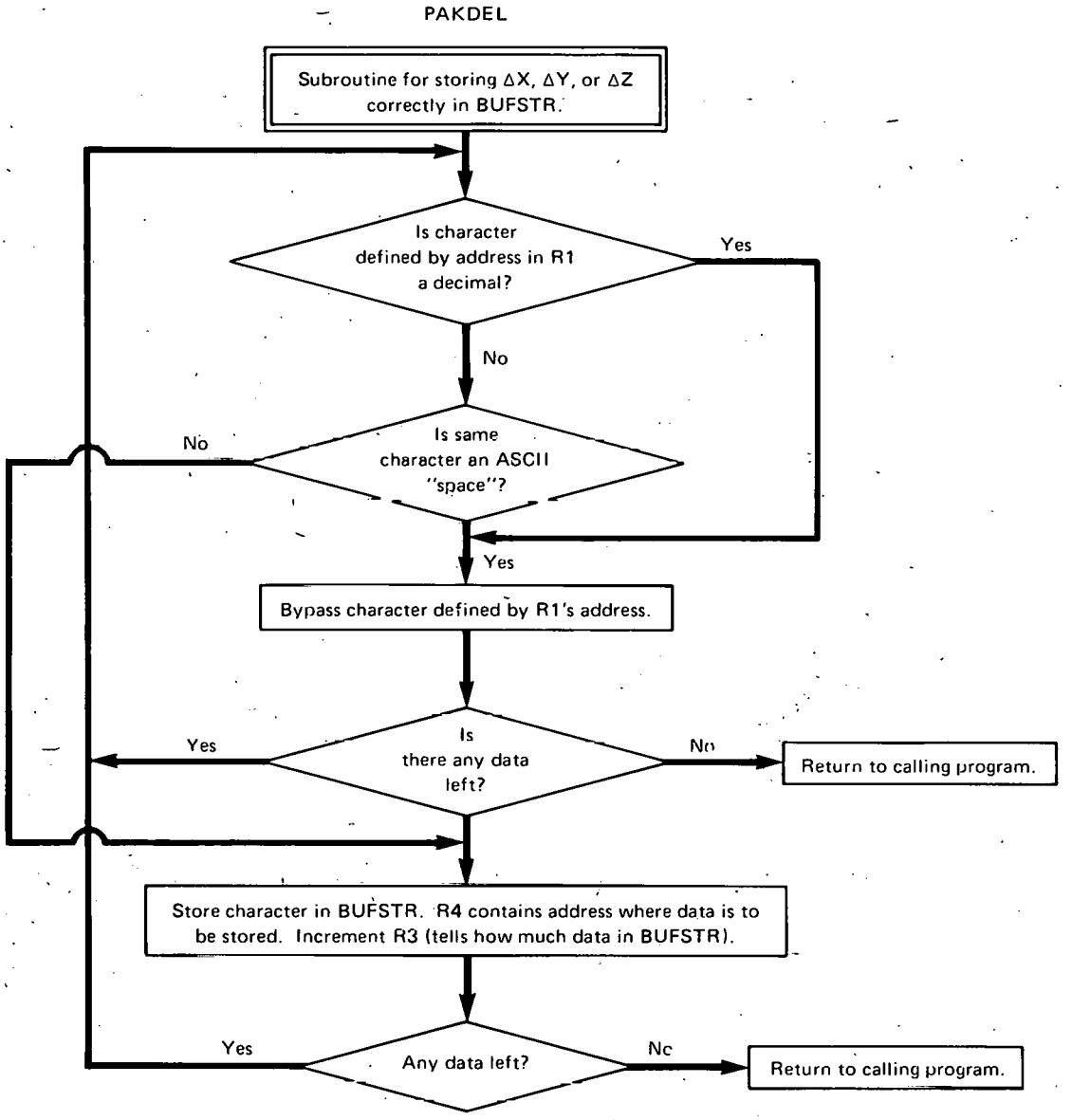


Figure A-27. SUBROUTINE FOR STORING DEPARTURES.

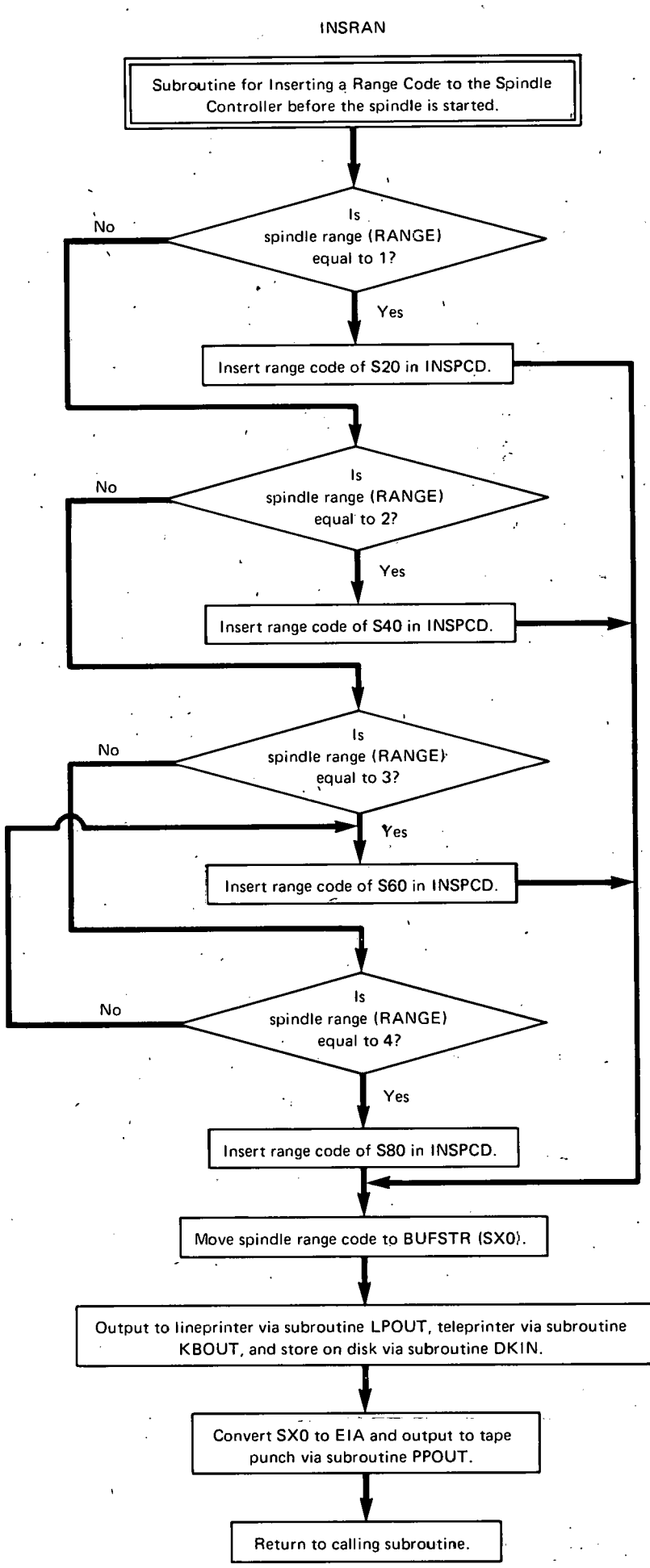


Figure A-28. SUBROUTINE FOR INSERTING THE S CODE BEFORE SPINDLE "ON".

SPINDL

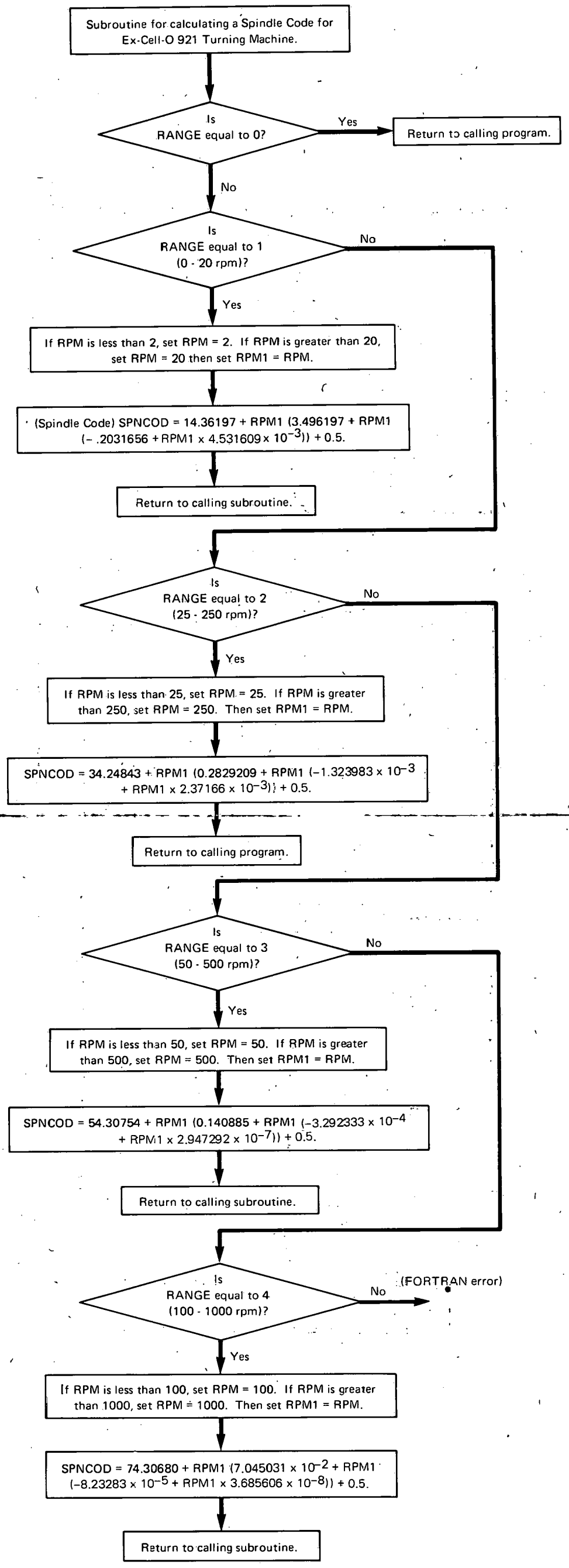


Figure A-29. FORTRAN SUBROUTINE FOR CALCULATING SPINDLE CODES.

RPM values from 100 - 1000, and if the RPM entry is 10, then the RPM entry will be reset to 100 before proceeding.

Subroutines ENCODE, CALDEL, and FLTFIX - Flowgraphs for these routines are identified in Figure A-30. "ENCODE" converts the real values of each X, Y, and Z departure to a ten ASCII character string and stores the data in buffer "BUFDLT". "CALDEL" calculates the feed numbers that will produce a delay equal to the number of spindle revolutions or seconds specified in the APT "DELAY" statement. "FLTFIX" converts a FORTRAN real number stored in "FORTR1" to an integer, storing the results in "FIXED".

Subroutine CALCRT - This FORTRAN subroutine (Figure A-31) is used to calculate data for the X and Y distances and velocity registers for the software interpolator used in the display software. This calculation is accomplished by first establishing the number of control pulses to travel the required distances in the following manner:

1. Make ΔX and ΔY (SCLXCO and SCLYCO) absolute, and truncate to the nearest 5.8608×10^{-3} inch.

2. Calculate the error for X and Y (XERR and YERR) caused by this operation.

3. Calculate the number of control pulses required to travel ΔX and ΔY (SCLXCO and SCLYCO).

4. Calculate the total error by summing the calculated error with that previously accumulated (total error is TXERR and TYERR).

5. If either of these errors is more than the movement of a control pulse, adjust SCLXCO and TXERR or SCLYCO and TYERR to keep the tool-path error (display error) within $\pm 5.8608 \times 10^{-3}$ inch.

After the distance register numbers are determined, velocity register data are calculated. These data are negative numbers such that when they are inserted into two counters and incremented at an equal rate, with counter overflows used to increment the distance counters (when an overflow occurs, the velocity register's original contents are restored), they will cause a path to be drawn on the CRT analogous to that of the machine's tool movement. Velocity numbers are calculated with an assumed constant feedrate that is the maximum value at which a legible line can be drawn on the CRT. These are determined as follows:

$$XCLOC = \left| \frac{4}{SIN\,\theta} \right|,$$

$$YCLOC = \left| \frac{4}{\cos\theta} \right|, \text{ and}$$

$$\theta = TAN^{-1} \frac{\Delta X}{\Delta Y}$$

Subroutine PRODA1 - This FORTRAN subroutine (Figure A-32) is used to calculate X departure, Y departure, Z departure, feednumber, and the spindle speed from CL data input. It is entered after each Type 2 CL data record that is not an APT "FROM" statement. The subroutine rounds off each departure to the nearest machine control pulse value, keeping account of the round-off error, and calculates the spindle speed and feedrate control numbers using the surface feet per minute (or revolution per second) and inches per revolution (or inches per second). These control parameters are set by previous APT "FEDRAT" and "SPINDL" statements. It also monitors feedrate and/or the spindle-speed change over a departure block, keeping these within a preset value by breaking movements into smaller parts when necessary and limits spindle speed and/or feedrate to previously set values. A general listing of the subroutine's operations are:

1. Add error due to last roundoff to X, Y, and Z.

2. Set DLTXCO, DLTYCO, and DLTZCO equal to the absolute value of movement to request control pulse. For example, DLTXCO = ABS (DSTXCO - FRMXCO) where DSTXCO is the records end point and FRMXCO is the previous end point.

3. If the roundoff error is greater than one half the control-pulse value, modify the delta movements to keep the error within ± one half the control-pulse movement value.

4. If the spindle-control parameter is in SFM, calculate the RPM. If the RPM change over the movement is greater than 15%, break the movement into a smaller section, setting the flag so outside software is aware of this.

5. Calculate the feednumber, adhering to previous input-control parameters.

6. Calculate the new roundoff error that will be used in the next Type 2 record processing.

7. Make this record's destination the following record's origin.

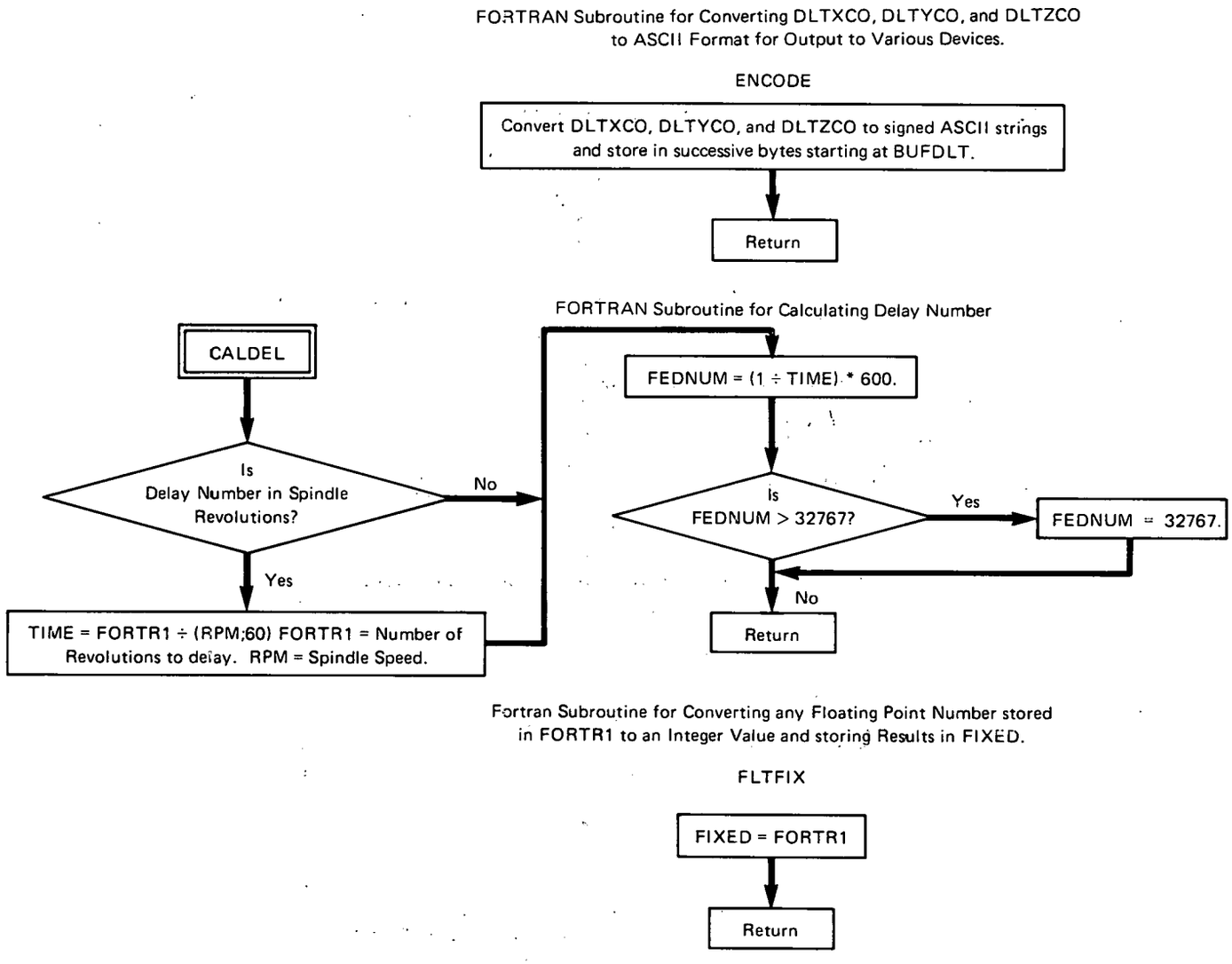


Figure A-30. FORTRAN SUBROUTINES FOR CALCULATING DELAYS, CONVERTING FROM REAL TO INTEGER, AND DECODING REAL DEPARTURES TO AN ASCII CHARACTER STRING.

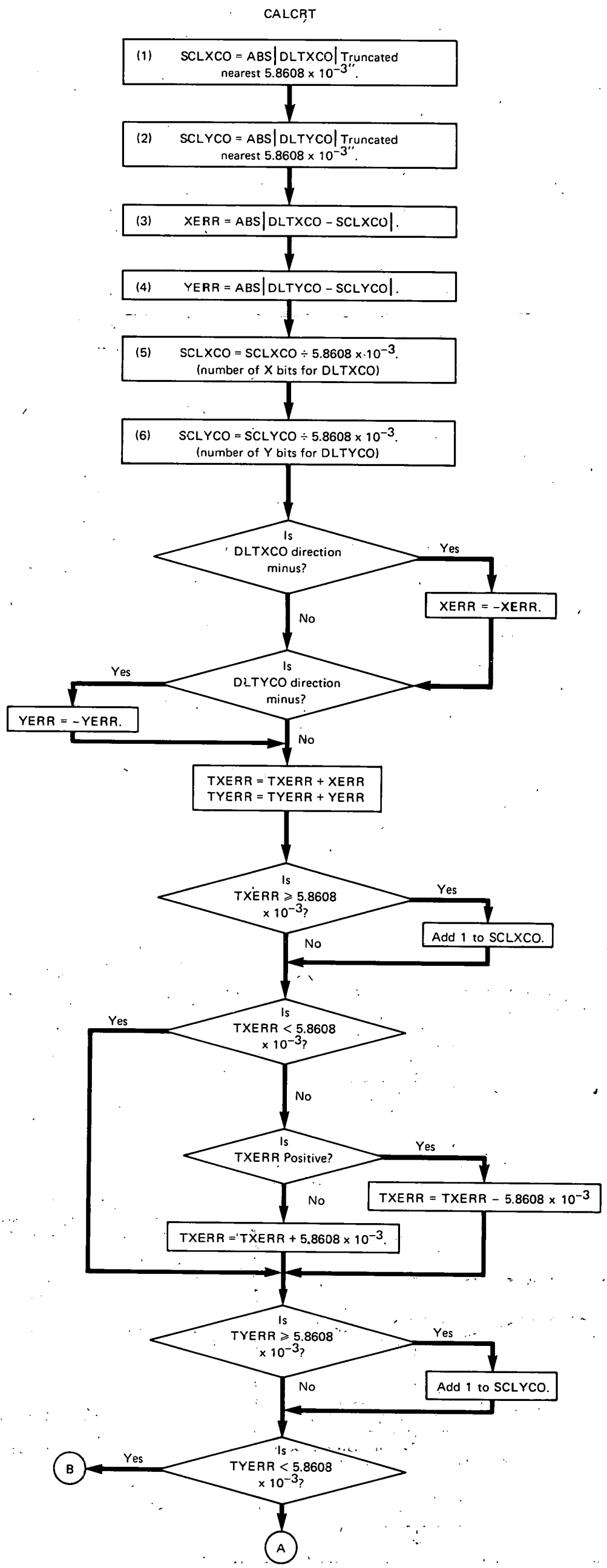


Figure A-31. SUBROUTINE FOR CALCULATING CATHODE-RAY TUBE DISPLAY CONSTANTS. (Section 1)

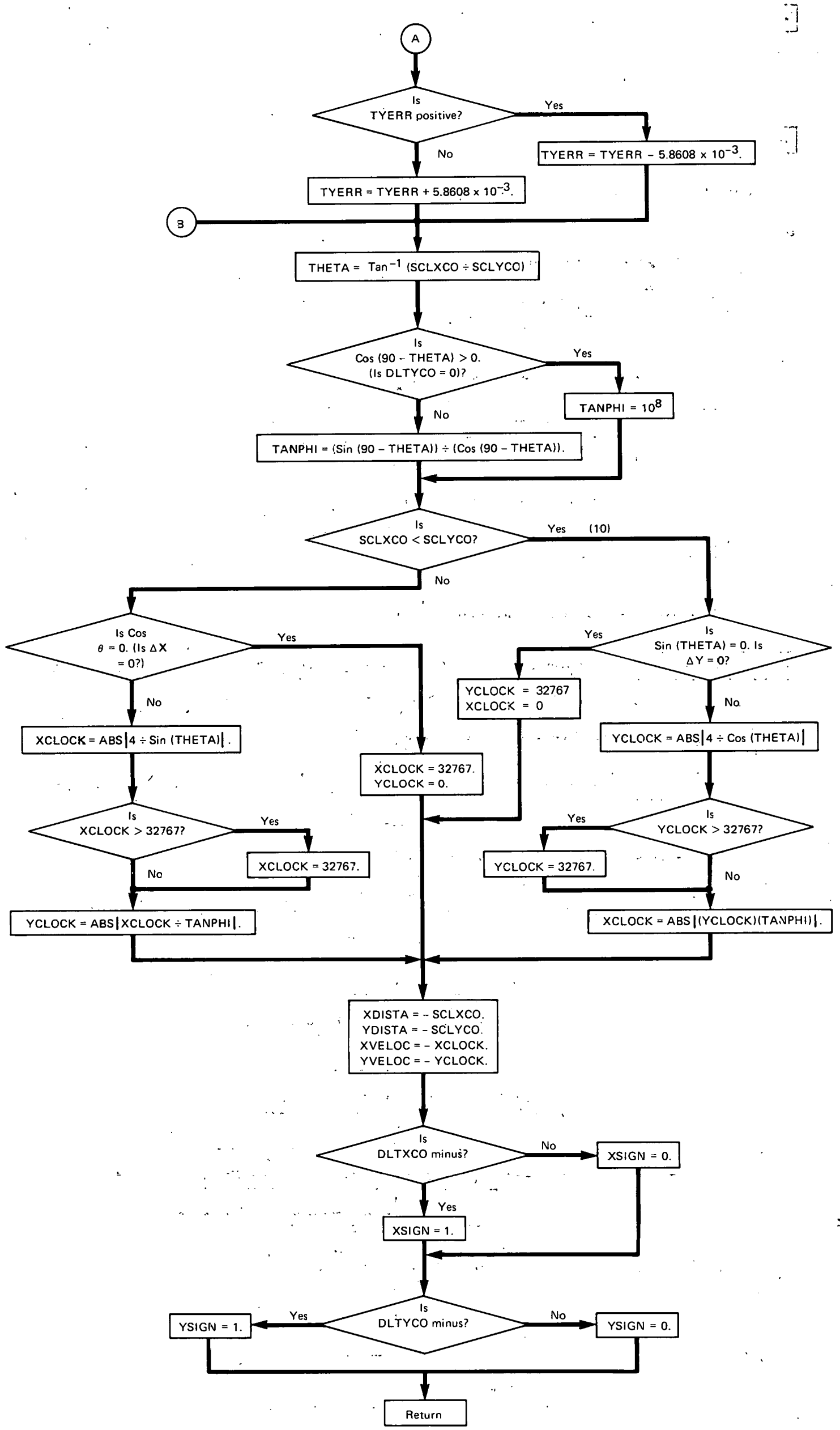


Figure A-31. (Section 2)

FORTRAN Subroutine for Calculating Data for the DDNC and Bendix 1800 Series Numerical Controllers.

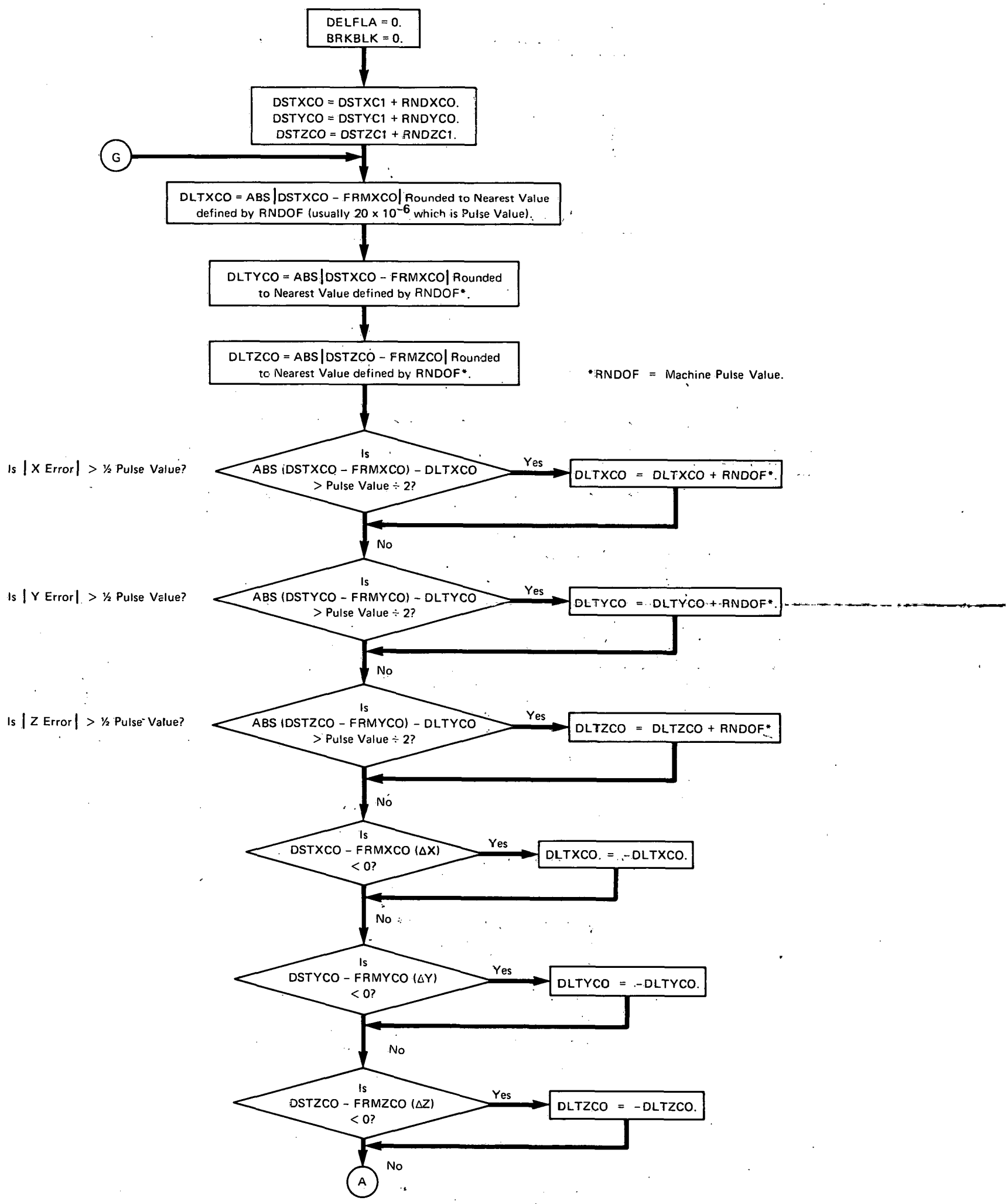


Figure A-32. SUBROUTINE FOR CALCULATING ΔX, ΔY, ΔZ, SPINDLE SPEED, AND FEEDNUMBERS. (Section 1)

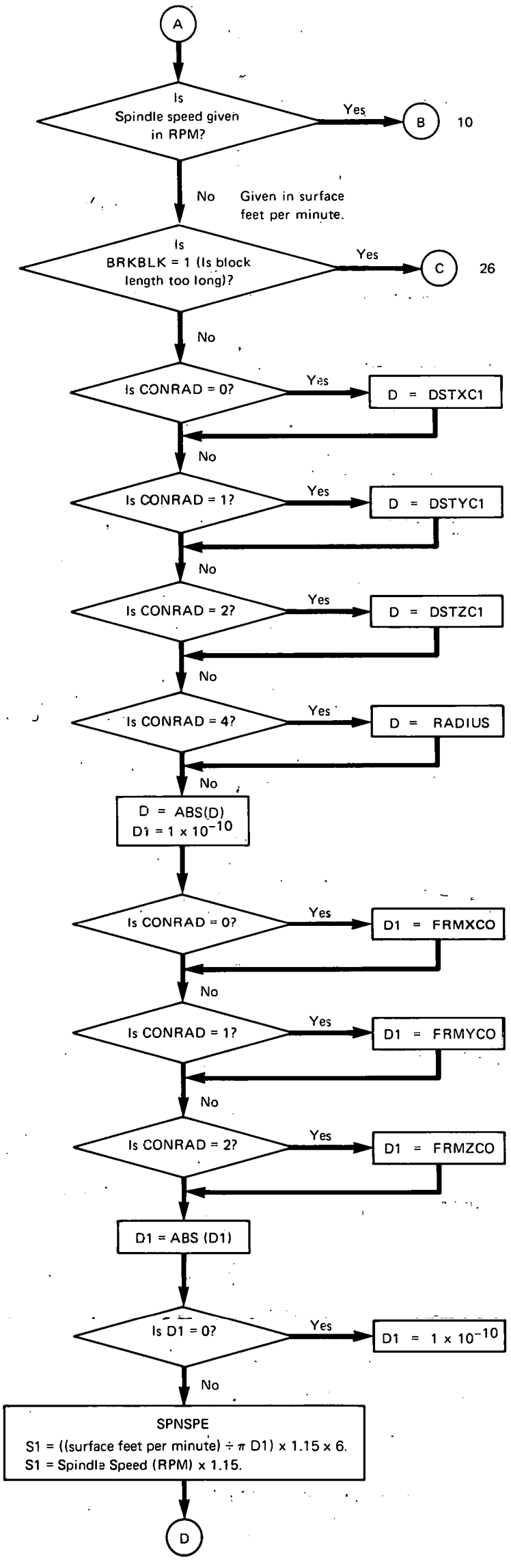


Figure A-32. (Section 2)

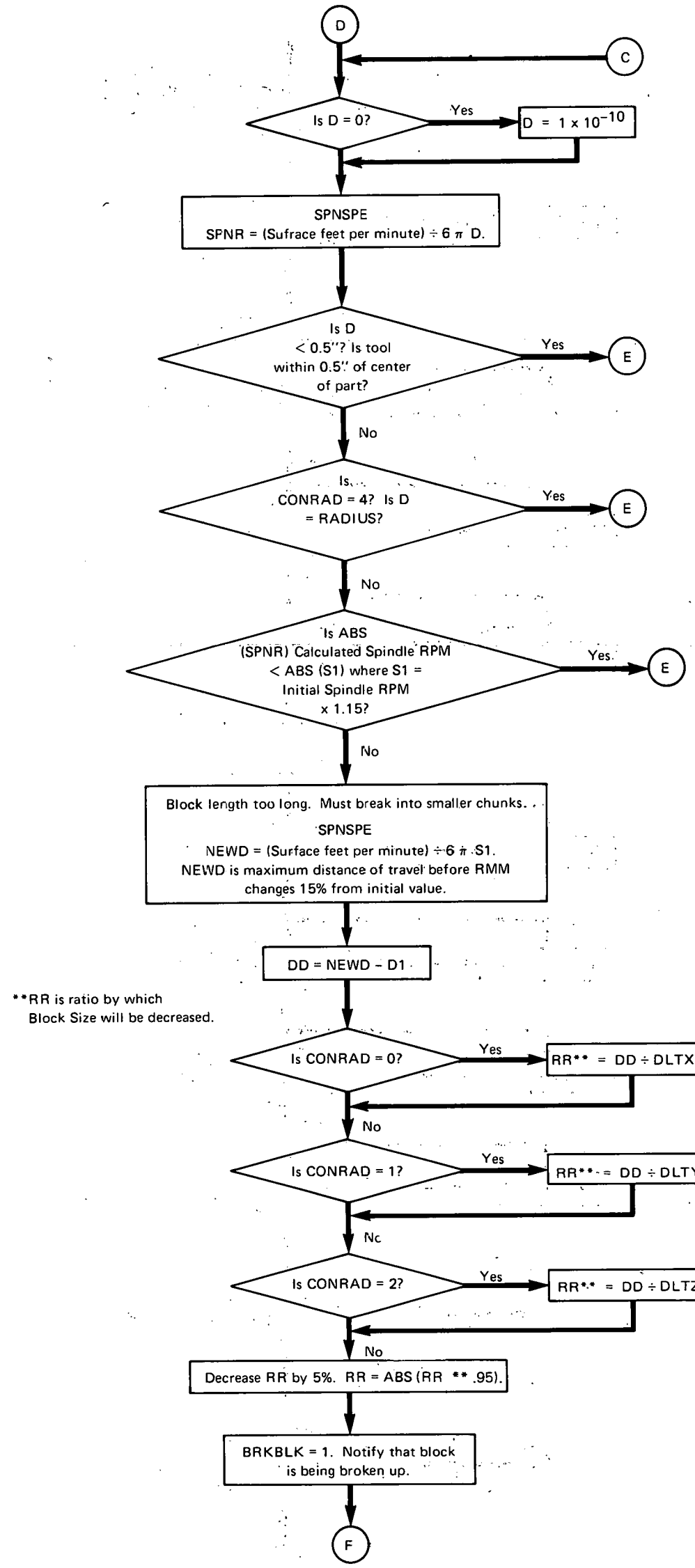


Figure A-32. (Section 3)

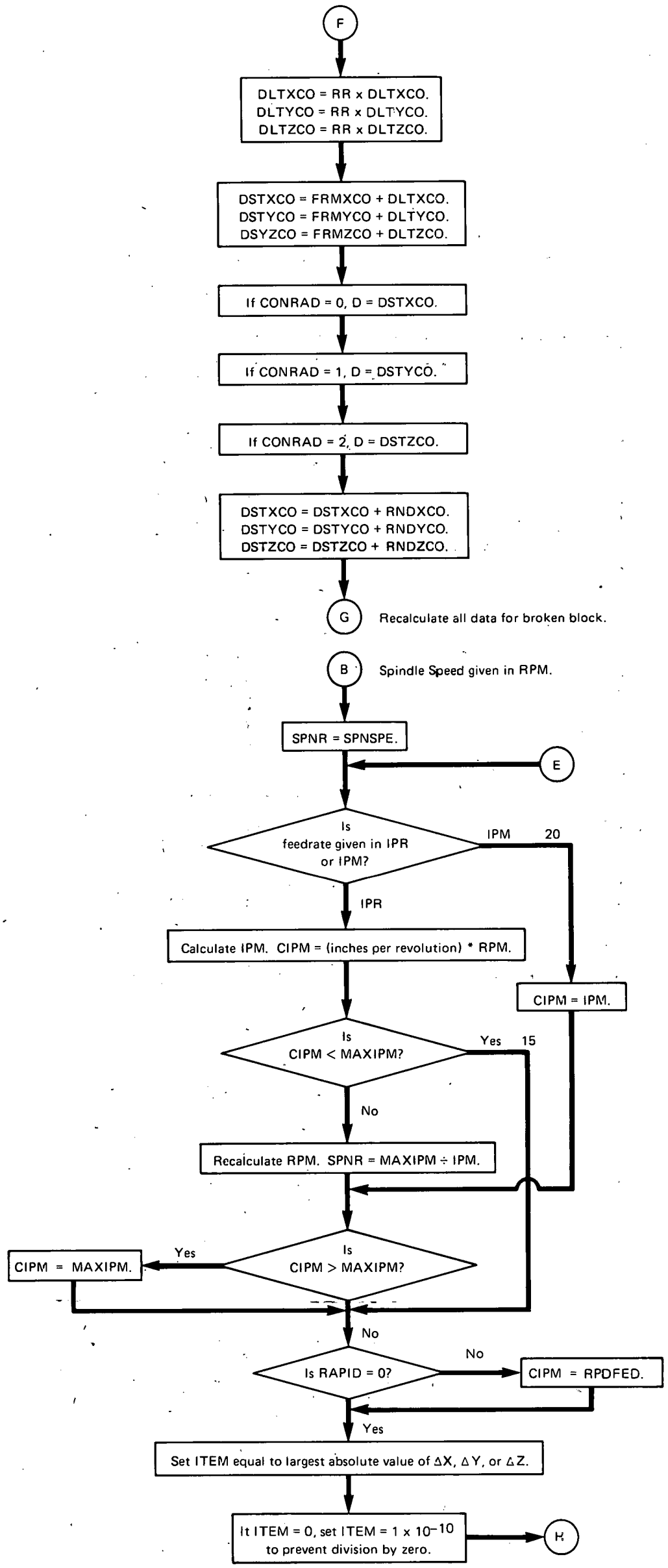


Figure A-32. (Section 4)

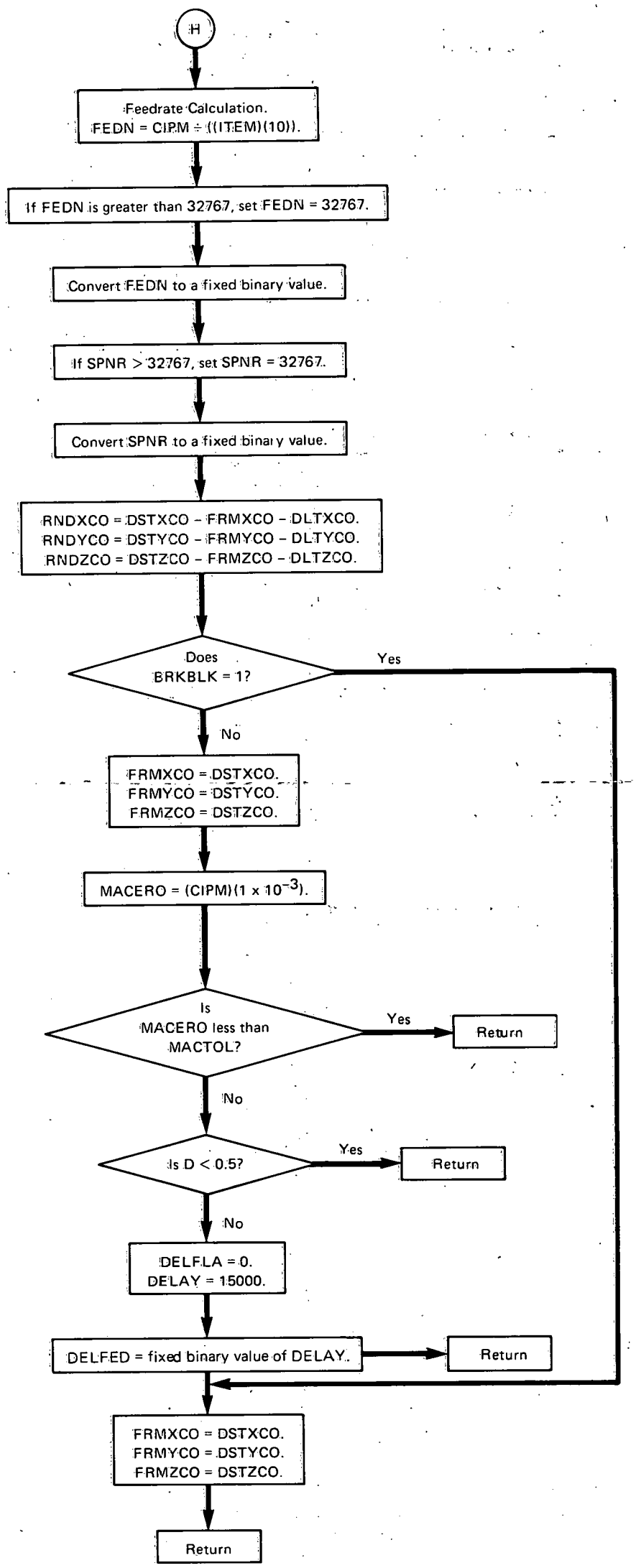


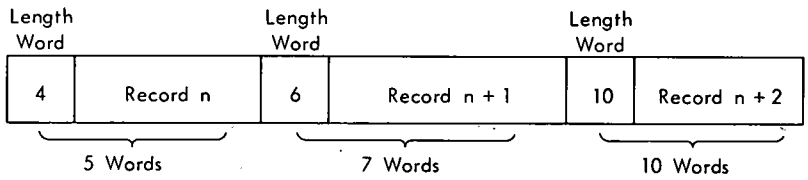
Figure A-32. (Section 5)

APPENDIX B

DISK CUTTER LOCATION DATA

Introduction

The disk CL data are stored in variable-length records as follows:

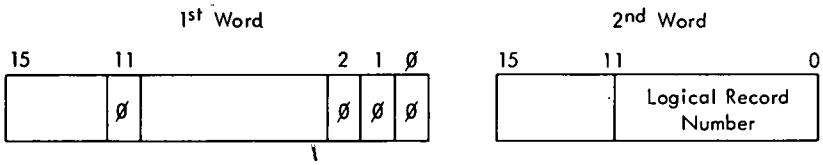


There are five CL record types, as listed in Table B-1. These types are of fixed length (with the exception of Type 2 which is variable) and contain the postprocessor commands listed in Table 1.

Table B-1
RECORD TYPES

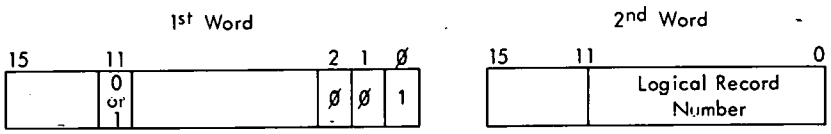
Type	Description	Length (number of bit words)
0	FINI Record	2 - 16
1	Postprocessor Commands	Variable
2	Cutter Position Record	14 - 16
3	Canonical Form of Circle	10 - 16
4	Multax Position Record	26 - 16

Type 0 - Type 0 record is the last record in the CL file. It is produced when the APT command "FINI" is encountered. The record will contain the following data:



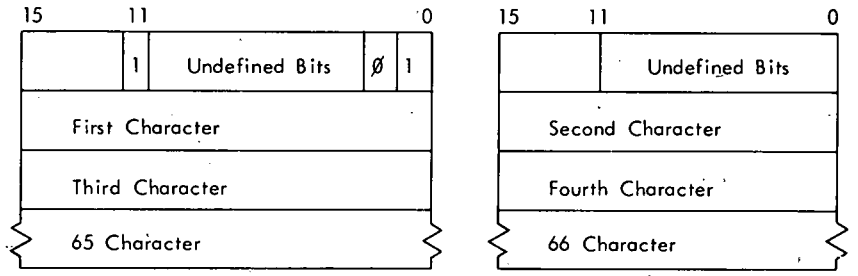
When this record is encountered, postprocessing is terminated.

Type 1 - Type 1 records are of variable length, but each record will begin with the following two computer words, followed by a subtype identifier:

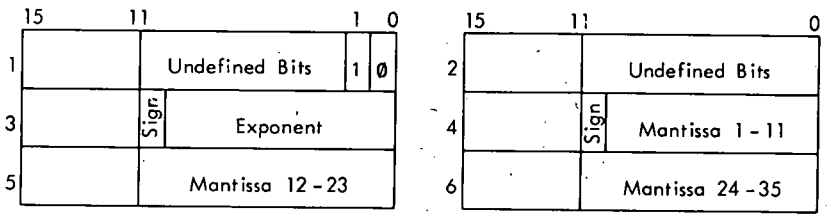


When Bit 11 of the first word is set, two records are being used to describe one CL record. There are three subtypes of which one will follow the previously defined data:

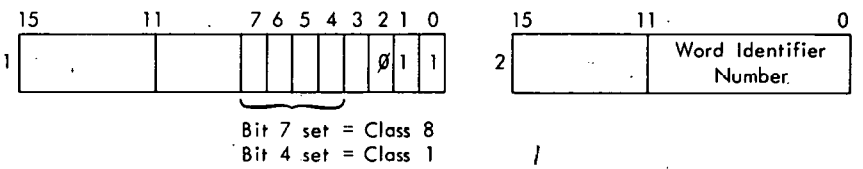
Subtype 1: This subtype notifies the postprocessor that an ASCII character string is to follow. The data format that follows is included everytime an APT PARTNO, INSERT, or PPRINT statement is encountered.



Subtype 2: Floating-point numbers occur on many APT postprocessor statements (example: SPINDL/720, RPM). The value for spindle RPM will be given as a UPL floating-point word and will be preceded by two identifier words as noted:

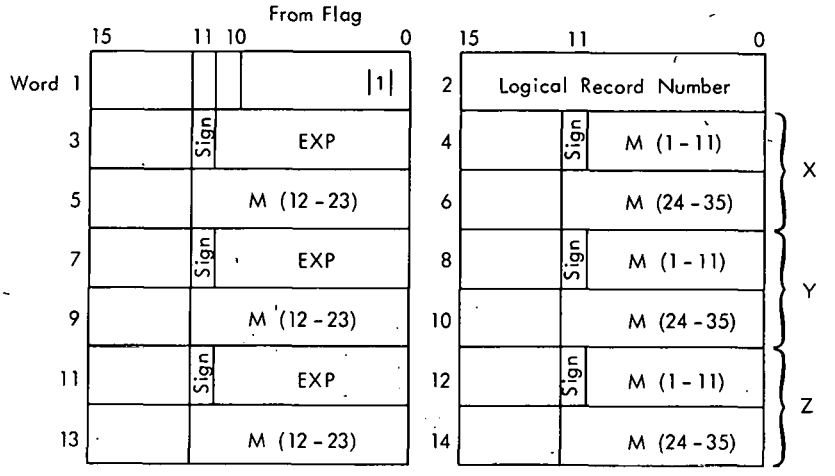


Subtype 3: Such vocabulary words as FEDRAT, SPINDL, and CLW are identified by the following data block:



There may be many subtypes within a Type 1 record. The number of subtypes is determined by the APT input statement. Bit 4 of Word 1 notifies the postprocessor that a Class 1 vocabulary word is to follow (example: FEDRAT, COOLNT, DELAY), and Bit 7 of Word 1 defines Class 2 words (example: ON, RPM, SFM, CLW). A table of vocabulary words and their numbers equivalent is included in Appendix A (Tables A-1 and A-2).

Type 2 - Cutter centerline data are given as three floating point numbers preceded by two identifier words as follows:



When the "FROM" flag is set in Word 1, it signifies that the following three numbers are the coordinates in an APT "FROM" statement.

Types 3 and 4 - These data types are used to transfer the canonical form of a circle and multiaxes centerline data. Neither of these are used by this postprocessor. Further information may be found in a manual by the United Computing Corporation.(d)

(d) *UNIAPT 3-D Contouring N/C Tape Preparation System for the PDP 11/10 Computer*; United Computing Corporation, Carson, California.

APPENDIX C

SYSTEM OPERATION

Introduction

The experienced operator will find that there are several methods that may be followed to acquire the desired results which is a part program that will produce a part on the selected machine tool. The following discussion outlines one procedure which will most likely change as the United Computing Corporation (UCC) updates its APT system.

Program Preparation and Input

Actually, there are only two ways to prepare a program for processing: (1) via the teleprinter keyboard, or (2) input from punched cards. Keyboard input requires the following procedure be followed: (Note that any underlined character is a response from the computer. For operation of the United Programming Language (UPL) editor, refer to United Computing's document: "3-D Contouring N/C Tape Preparation System for the PDP-11/10 Computer".)

```
$ RU PIP [CR]
PIP-11 V7A01
# XXXXXX.YYY< KB:[CR]
```

Note: XXXXXX is a six-character alphanumeric file name and YYY is a three-character alphanumeric file name extension. The naming system for part programs includes the type of operation and part number. An example would be FIC123.456. This number is derived from the finish inside contour on Part Number 1234-56.

Input part program via teleprinter:

```
PARTNONC1234567    W.R. XXXX    FIC9876-54
_____________
_____ _______
_____________
FINI
. [CR]
[CTRL] C
. END [CR]
[LF]
#
```

At this time, file XXXXXX.YYY is stored on the disk for APT processing. The program is now transferred to the UPL editor's library as follows:

```
# [CTRL] C
. KI [CR]
$ ASSIGN DK0: XXXXXX.YYY, CRI [CR]
$ RU UPL [CR]
$ ED/␣UPL EDITOR ---- (␣= Space Bar)
/ RC, XXXXXXYYY,
```

The disk file XXXXXX.YYY is now read into the UPL editor's library where the APT processor is called and run on the program as well as any required MACROs:

```
/ U, SYNTAB, A5W128, AL, STRTUP, SHUTDN, ATLSET, XXXXXXYYY,,
```

It is assumed that MACROs have already been installed in the editor's library. The processor call statement is terminated with the double commas. At this time, processing begins. United Computing's APT processor is a two-pass system where the first pass checks for syntax errors. Any error will terminate processing back to the editor after Pass 1, where editing is done to eliminate any errors and processing is reinitiated:

```
END PASS ONE XX ERRORS
/
```

If no errors are encountered during Pass 1, Pass 2 is entered for the generation of required output:

```
END PASS TWO
XX ERRORS
EOJ YYY RECORDS
$ EX/␣(␣= Space Bar)
$
```

The system is now running under the DEC disk operating system (DOS).

Punched-card input of part programs requires that the following procedure be followed:

1. Load the card deck in the card reader, making sure there is a UPL terminator card followed by a blank card.

2. Enable the card reader.

3. At the teleprinter:

```
$ ASSIGN CR:, CRI [CR]          (Later versions will not require this
                                 statement.)
$ RU UPL [CR]                   Call UPL operating system.
$ ED/␣EDITOR V005               Call UPL editor.
/ RC, XXXXXXYYY,                Read the card deck into the editor
                                 library with the name
                                 XXXXXXYYY.
/ U, SYNTAB, A5W128, ---, ---, XXXXXXYYY,,
```

Run the APT processor as before.

Running the APT Processor

There are three possible outputs from the processor: (1) first-pass listing of the APT program, (2) second-pass CL listing, and (3) the CL disk file. Any or all of these outputs may be disabled while processing a program. There are also two ways to process an APT program: (1) from the UPL operating system, and (2) from the UPL editor. Operation from the editor has been previously outlined, but no mention was made of any processing options. These options are enabled by inserting an "S" after the "U" when calling APT under editor control followed by:

US, ij, SYNTAB, A5W128, D38, ---, XXXXXXYYY,,

by an i j option code (options are defined in Figure C-1). When running from the editor, j must either be 3 or 7 for disk input, but i can assume any of the numbers from 1 through 7. Default options for editor control of APT processing are i = 1 and j = 3 for error list on Teletype, source listing on the lineprinter, disk input, and CL output to the disk.

Output	Response	Meaning		
OPT = i	0	List errors on Teletype.	List source.	Teletype
	1			Lineprinter
	2		No source listing.	Teletype
	3			Lineprinter
	4	Suppress Pass 1 Teletype error messages.	List source.	Teletype
	5			Lineprinter
	6		No source listing.	Teletype
	7			Lineprinter
		Use default options-bypass IO = 1, 3.		
IO = j	0	Card Reader Input	CL Tape to Disk	
	1	Teletype Input		
	2	HS Paper Tape Input		
	3	Disk Input		
	4	Card Reader Input	Suppress CL Tape (NO TAPE)	
	5	Teletype Input		
	6	HS Paper Tape Input		
	7	Disk Input		
	,	Use Default IO selection.		
	,	Start Pass 1.		

Figure C-1. UNIAPT PROCESSING OPTIONS.

Part programs may also be processed under control of UDS (United Disk System) by following this procedure:

$ RU PIP [CR]	(Call DOS File Utility Package)
#XXXXXX.YYY <CR: [CR]	(Input APT source program and store on disk using the naming convention previously defined. Input may also come from the keyboard.)
#CTRL [C]	
. KI [CR]	
$ ASSIGNDKO:XXXXXX.YYY, CRI [CR]	
$ RU UPL [CR]	

Run the APT processor on the file defined by the ASSIGN statement:

$ UNIAPT/, ⎵ or UNIAPT/ij ⎵	(i and j are optional procedure flags just as was with the editor (refer to Figure C-1). When not included in the command, as is the first case, i and j default to 1 and 3).

It is generally necessary to edit the APT source program. This work should be done with the DEC editor when running APT from UDS, and the UCC editor when running under control of the UCC editor. For guidance in their use, refer to DEC's "Edit-11 Text Editor Programmer's Manual" and UCC's "UNIAPT Operator's Manual". Editing should occur before the "ASSIGN" statement when running the processor under control of UDS.

Running the Postprocessor

The UCC APT processor stores a CL file on disk when run under the proper i and j options, and the UCC APT postprocessor statement MACHIN/40, i, j, k is included in the source program. The options defined by i, j, and k are shown in Figure C-2. Normally, all postprocessors are written by UCC and are automatically called by this statement where the first number (40 in the example) defines the particular postprocessor. This approach is not the case with the Y-12 system. A Null postprocessor was purchased (defined by MACHIN/40) that takes the CL data and stores it on disk in a previously defined area with the format defined by UCC's "Postprocessor Manual CL Tape-to-Disk". This equipment allows access by any program (postprocessor) written for the PDP-11 computer system.

As stated before, the APT source program must include a MACHIN statement for postprocessing. One other statement that is required is a PARTNO immediately followed by nine alphanumeric characters (normally the NC tool path number without its dashes PARTNONC1234567 instead of TAPENO NC-12345-67). These nine characters are used by the postprocessor to setup a DEC DOS file with a file name NC1234 and extension of 567 on the disk peripheral (NC1234.567). If these two criteria have been fulfilled, a DOS RU POS922 statement will initiate successful operation of the postprocessor.

i (2nd) Field[1]		j (3rd) Field		k (4th) Field		
Code	CLTAPE (input to postprocessor)	Code	Object Tape (output)	Code	Listing (output)	Legend[2]
1	Teletype		TT Paper Tape	1	TT Printer	X
		1	TT Paper Tape	2	Printer	√
			TT Paper Tape	0	None	√
			HS Paper Tape	1	TT Printer	√
		2	HS Paper Tape	2	Printer	√
			HS Paper Tape	0	None	√
			None	2	TT Printer	√
		0	None	1	Printer	√
			None	0	None	√
2	High-Speed Paper Tape		TT Paper Tape	1	TT Printer	X
		1	TT Paper Tape	2	Printer	√
			TT Paper Tape	0	None	√
			HS Paper Tape	1	TT Printer	√
		2	HS Paper Tape	2	Printer	√
			HS Paper Tape	0	None	√
			None	1	TT Printer	√
		0	None	2	Printer	√
			None	0	None	√
0	Disk		TT Paper Tape	1	TT Printer	X
		1	TT Paper Tape	2	Printer	√
			TT Paper Tape	0	None	√
			HS Paper Tape	2	TT Printer	√
		2	HS Paper Tape	1	Printer	√
			HS Paper Tape	0	None	√
			None	2	TT Printer	√
		0	None	1	Printer	√
			None	0	None	√

(1) MACHIN/40, i, j, k.
Default options for i, j, and k or 0, 0, 1 for input from the disk, no object tape (EIA), and high-speed printer output of listing.

(2) A check mark (√) indicates that the option is valid; an "X" indicates that it is invalid.

Figure C-2. MACHIN/STATEMENT OPTIONS.

There are four types of outputs from the postprocessor: (1) disk postprocessed part program file, (2) object tape (EIA), (3) CRT display of toolpath, and (4) listing of the part program with interspersed comments and error messages. The CRT display is enabled by the APT source program command "DISPLY/ON" and disabled by "DISPLY/OFF". (A list of error messages and their meaning is given in Table A-4.) The disk file created by the postprocessor is an ASCII image of the EIA part description used for controlling the machine tools in question. These data are accessed by other software for storage in the DDNC's core memory which acts as the tape reader.

Source and Object File Record Retention

Files that reside on the disk are considered to be volatile. Permanent records of APT source and object files must be kept on DECtapes. Source files are generally transferred from the UDS editor library, but also may be transferred from the DOS files on the disk. Object programs (those generated by the postprocessor) are renamed to PARDES.XXX (where XXX is the program number used by the DDNCs for program requests) and stored on magnetic tape using the DOS PIP software.

Transfers from the UDS Editor Library - Assume that APT processing is complete, the postprocessor has been run, and no additional corrections are to be made to the source program. Also, assume that the computer is running under DEC's DOS under the proper user identification code (uic) which was previously entered as $ LOGGIN 200,200 [CR]. Observe the following procedure for transfer to DECtape:

1. $ ASSIGN DTX: FOC123.456, PPO [CR]
2. $ RUN UPL [CR] — System is operating under UDS.
3. $ ED/_ — Call UCC editor.
4. Make sure the "WRITE" switch is set to "WRITE ENABLE".
5. / P, SOURC1,, — Write program on DECtape selected by DTX.
6. / $ [CR] — Close file on DECtape by exiting editor.
7. $ EX/_ — Operating under UDS Exit UDS.
8. $ — Now operating under DEC DOS.

Sometimes the source program is broken into two parts (when the turning MACROs are being used). The first part being preliminary information, surface definitions, and a MACRO that defines the surface to be processed by the turning MACRO. The second part contains all tool path statements. When making permanent records it is desirable to store these as one program. This storing is accomplished by using the title of each part in Step 5:

/ P, DEFINI, TLPATH,,

This step causes the two library files (DEFINI and TLPATH) to be written on DECtape with a file name and extension FOC123.456.

Transfers from DOS Disk Files - As mentioned earlier, the user is able to process APT source programs directly from DOS disk files. It is also possible to process from DECtape, punched tape, card reader, or even keyboard entry, but most efficient operation is achieved by entering data as a DOS disk file or into the editor library. After all processing is completed, these files are transferred to DECtape using PIP. Assume the computer is running DOS under proper uic:

```
$ RUN PIP [CR]
PIP-11 V7A01
#DTX: FOC123.456 < DK0: FOC123.456 [CR]
#              Transfer finished.
```

Note: It is not necessary for file names to be the same.

As before, the DECtape must be write "enabled" (control switches set to "WRITE ENABLE' and "REMOTE" before data can be written on tape).

Object programs (those created by the postprocessor) are also stored on magnetic tape using PIP, but one additional operation is required. The only programs that may be loaded in the DDNCs are those with a file name of "PARDES", followed by a variable numerical three-digit extension. The postprocessor stores the object file with the name and extension defined by the first nine characters following the APT statement "PARTNO", which is generally NC followed by seven digits and never PARDESXXX. This action is necessary to prevent a machine operator from accessing a tool path before it is fully debugged. Assume operation under PIP:

PARDES.XXX < NC1234.567/RE [CR]

This operation renames the file NC1234.567 located on disk to PARDES.XXX

#DTX: PARDES.XXX < DK0: PARDES.XXX [CR]

Write renamed file on DTX:. As before, the DECtape control switches must be properly set to allow entry of data.

#

The object file is now stored on tape and on the disk. It should remain on the disk until all parts that it describes are manufactured. This step is not necessary, but it allows the DDNC to obtain the tool path within seconds instead of minutes (which occurs when tape searches are made). The program may be removed from the disk under PIP using the following command:

DK0: PARDES.XXX/DE [CR].

Table of Contents for the Disk and DECtape

Each tape and disk may contain many tool paths with an identification file name of PARDES and a variable extension of 1 to 999. Each tape has a table of contents stored with a file name and extension of LIBRAR.Y. The table of contents contains the part number, type of operation, NC number, work order number, and file name and extension for each tool path stored on the tape, thus:

PART NO	TYPE OPERATION	NC NUMBER	WO NUMBER	FILE NAME & EXT
6659	FOC	NC12345-67	W.R. 1234	PARDES.XXX

The table of contents may be appended by observing the following procedure:

1. Assuming running under DOS with correct uic.

2. $ RUN PIP [CR]
   PIP-11 V7A01

3. #TEMP/DE Eliminate any disk file called TEMP.

4. # TEMP<DTX: LIBRAR.Y

5. # Hold "CTRL" key and strike "C".

6. . KILL [CR]

7. $ RUN EDIT [CR]
   EDIT-11 V006A
   # TEMP<TEMP

8. * R500AI [CR]

9. Operator types in the following data:

   a. Part Number
   b. Hold "CTRL" Key and Strike I (Tab)
   c. Operation (FIC, FOC, RIC, POT, FIX, etc)
   d. Hold "CTRL" Key and Strike I (Tab)
   e. NC Number
   f. Hold "CTRL" Key and Strike I (Tab)
   g. Work Request Number
   h. Six Dashes
   i. Carriage Return [CR]
   j. Line Feed

10. * EX [CR] Restore edited file in TEMP.

11. Hold "CTRL" key and strike "C".

12. . KILL [CR]

13. $ RUN PIP [CR]
    PIP-11 V7A01

    Set "WRITE" switch to "WRITE/ENABLE" on DECtape
    where LIBRAR.Y originated.

    #DTX: LIBRAR.Y/DE [CR] Delete old file.
    #DTX: LIBRAR.Y<DK0: TEMP [CR] Replace with edited library.

14. # Hold "CTRL" key and strike "C"
    . KILL [CR]

15. $

A table of contents also exists on the disk which contains a list of all part programs that are presently stored on the disk and four DECtapes. It is also stored with file name and extension LIBRAR.Y. To append this table, observe the following procedure:

1. $ RUN EDIT [CR]
   EDIT-11 V006A

2. #DK0: LIBRAR.Y <DK0: LIBRAR.Y [CR]

3. * R500AI [CR]

4. Input data defined under previous Step 9.
   a - j

5. * EX [CR]

6. * Hold "CTRL" key and strike "C".

7. . KILL [CR]

8. $ RUN PIP [CR]
   PIP-11 V7A01
   #*.BAK/DE. (Delete all back up files.)

9. Hold "CTRL" key and strike "C".

10. . KILL [CR]
    $

APPENDIX D

DIRECT NUMERICAL CONTROL COMPUTER SYSTEM OPERATION

Introduction

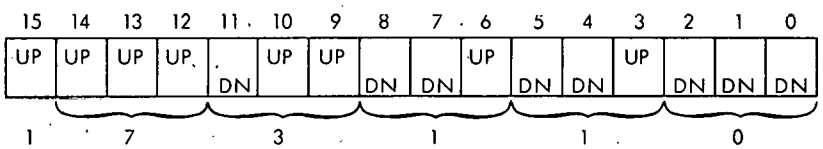
The computer system consists of a 24,000-word computer (16-bit words) with an electronic extended arithmetic element (EAE) and a small read only memory (ROM) for permanent storage of startup programs for the DECtapes and cartridge disk. Peripherals include a card reader, lineprinter, four DECtape drives, one 1.25×10^6 word movable head disk, cathode ray tube display, paper tape reader punch, and a teleprinter.

Starting Up the Computer

Before system startup, the following conditions must be met:

Computer Power Switch	- "ON"
RK05 Disk Cartridge	- Loaded with "LOAD-RUN" Switch to "RUN"
DECtapes	- Remote-Off-Local Switch to "REMOTE"
Lineprinter (if being used)	- Power Switch - "ON" Select Switch - "ON"
Teleprinter	- Set Control Switch to "ON LINE"
Card Reader (if being used)	- Power Switch "ON" Start Switch "ON"
CRT (if being used)	- Power Switch "ON"

The computer is started by loading the octal number 173110 in the computer's front panel switch. This action is accomplished by setting the switches according to the following diagram: (The switches marked "UP" are to be lifted up and those marked "DN" are to be pushed down.)



PDP-11/05 Switch Register

The "HALT-ENABLE" switch is moved to "HALT" then to "ENABL", the "LOAD ADDR" switch is depressed, and the "START" switch is depressed. If the disk is up to speed, the teleprinter will type out the following message:

```
BATCH V08.02
$
```

The operator responds by typing in the date, time, and log-in number.

```
$ DA 13-NOV-75 [CR]
$ TI 14:55:30 [CR]
$ LO 200,200 [CR]
$
```

The operator may now call and run programs via the DOS command "RUN filename. extension [CR]". Programs that will generally be used by the operator are PIP, EDIT, UPL, DNCEXC, POS922, EIAASC. Other programs that are available are a FORTRAN IV compiler, a MACRO assembler for processing assembly language programs, LINK for building system programs, and CILUS for system modification.

PIP (Peripheral Interchange Programs) - This is a file utility package that is used for housekeeping. It is called via the DOS command $ RUN PIP [CR] and responds with the following message:

```
PIP-11 V7A01
#
```

The # sign notifies the operator that the software is ready for input. Detailed information on PIP may be found in the DEC manual DEC-11-UPUPA-B-D.

Edit - This is the PDP-11 editor and is a text-editing program. Operated by user commands from the keyboard, the editor will read ASCII files from any logical device, make directed changes, and write the file on any logical device. In addition to basic editing functions, the editor provides for command MACROs and multiple input and output files. As was PIP, the software package is called via the DOS RUN command. (See DEC manual DEC-11-EEDA-D for further details.)

```
$ RUN EDIT [CR]
```

UPL - Called by the DOS command $ RUN UPL [CR], UPL (United Programming Language) consists of a United disk system, an APT processor, a null processor, and a file-oriented editor for maintenance of ASCII source files and EIA object files. When initially installed, UPL commandeers 1024 blocks on the disk under priorities such that no other device other than its own operating system can modify any part of this area. This area in fact acts as if it were a separate disk with its own operating system. This package allows the user to process ASCII source programs stored on any logical peripheral, entry of source files from any logical peripheral into the editor's library, editing of files stored within the editor's library, APT processing of any file or conbination of files within the library, and output of the source files to any logical device. Further information about UPL is contained in the other Appendixes and UCC's "UNIAPT Operator's Manual".

POS922 - This is a postprocessor for APT CL data generated by the APT processor and stored in UPL's extend disk area. The program is called via the DOS $ RUN POS922 command.

DNCEXC - This software is again called via the DOS $ RUN DNCEXC [CR] command. The software monitors the DNC communications interface for part program requests from the DDNC units. When a request is generated, the software searches all library peripherals for the requested part program. When found, the program converts the ASCII data to 36-bit words of data that are formatted for the DDNC. These data are then serially transmitted to the requesting DDNC unit where the part data are stored in a magnetic core memory.

EIAASC - Sometimes it is necessary to machine parts on the DDNC-controlled machines from programs that were generated for standard units with punched tape readers. This program, called by the DOS $ RUN EIAASC [CR] command, reads the EIA punched tape using the DNC's high-speed tape reader, converts the EIA data to ASCII, and stores the tape according to user keyboard entry commands.

Shutting Down the Computer

Before the computer power can be turned off, the following procedure should be followed:

1. Hold down the "CTRL" key and strike "C".
2. Computer responds with a "_".
3. Type "KILL [CR]".
4. Computer responds with a "$".
5. Type "FINISH [CR]".
6. Computer responds with "BATCH VXX".
7. Turn computer system power key to "OFF".

APPENDIX E

BUILDING THE CENTRAL CONTROL COMPUTER'S DISK

Introduction

The disk must contain DEC's disk-operating system, United Computing Corporation's NC tape-preparation system, and a postprocessor for the Ex-Cell-O 921/922 using a DDNC or Bendix 1800 control, DDNC service software package, and the EIA-to-ASCII tape converter for proper system operation. These systems are built onto the disk according to the procedure that follows.

DEC's Disk Operating System

This system is constructed from four DECtapes. The procedure is outlined in DEC's "System Manager's Guide DEC-11-OMGRA-A-D". FORTRAN capability is added by loading the FORTRAN backup DECtape on Unit 0 and following the outlined procedure.

1. If the user is not logged in under 1,1, do so; otherwise ignore this step.

```
$ LOGIN 1,1↓
DATE: 06-MAY-75
TIME: 13:41:26
$
```

2. Use PIP to transfer backup FORTRAN files to disk.

```
$ RUN PIP
PIP-11 V7A01
# DK0:<DT0: FORTRN.LDA↓
# DK0:<DT0: FORTRN.OVR/CO↓
# DK0:<DT0: FTNLIB.OBJ↓
# DK0:<DT0: FORCOM. DGN/CO↓
# DK0:<DT0: FORRUN.DGN/CO↓
# ↑C
. KILL↓
$
```

UCC's NC Tape Preparation System

In order to eliminate the necessity for writing a software package for every possible computer, UCC writes all of its system software in its own language (UPL) for a fictitious 24-bit computer. This method allows them to write one NC tape preparation system and an interpreter for each computer on which the package is to be installed. The interpreter is far less complex than the tape preparation system and is used to convert the UPL computer commands into that of the host computer before they are transferred to permanent storage.

The software package includes the following paper tape programs.

1. Interpreter - Two unmarked tapes of which the very short one contains a DEC assembly language list of variable control parameters which is edited to fit the user's own unique system. The longer tape is an object program of the interpreter which is linked to the previously edited and assembled short tape to form the interpreter program.

2. B2UDS - One tape which contains an initial UDS which is defined as PL00. When the interpreter is initially called, it first looks on the disk for a 1024-block continuous file called DISK.UPL; if it is not found, one is created and the B2UDS is read in as the operating system for installing the remaining software.

3. PL00 — The latest version of the operating system.

4. PL01 - PL03
   PL10 - PL16
   PL20 - PL37 — Tape Preparation System (APT)

5. PL40 — Null Postprocessor.

6. PL77 — UDS Editor

The following procedure must be observed to install the software:

1. Read in the short-assembly-language interpreter tape using PIP. (Assume that the user is logged in under uic 200,200.)

```
$ RU PIP↓
PIP-11 V7A01
# DK0:  UPLICB.MAC< PR:/FA↓
```

2. Read in the long-object interpreter tape using PIP.

   After this input, kill PIP.

```
# DK0:  UPL.OBJ < PR:/FB↓
# ↑C
. KILL↓
```

3. Edit the short tape (UPLICB.MAC) using the DOS editor, then kill the editor.

```
$ RUN EDIT↓
EDIT-11 V006A
# UPLICB.MAC < UPLICB.MAC↓
* R↓
```

Change ZKBI, ZPRI, ZPPO, ZCRI, and ICN to the following:

```
ZKBI = Q + O + C + E + X + U
ZPRI = Q + U + B
ZPPO = X + D + U + B
ZCRI = C + D + F + A
ICN = 30.

* EX↓
# ↑C
. KILL↓
$
```

4. Assemble the previously edited program.

```
$ RUN MACRO↓
# UPLICB, LP: < UPLICB

END OF PASS 1
ERRORS DETECTED: 0
FREE CORE XXXXX. WORDS
# ↑C
. KILL↓
$
```

5. Link UPL.OBJ with the previously assembled program to obtain a load module of the interpreter.

```
$ RUN LINK↓
LINK V11A01
# UPL,LP: < UPLICB, UPL/E↓
LINK U11A01
# ↑C
. KILL↓
$
```

6. Load B2UDS into the tape reader and run the previously linked interpreter.

```
$ ASSIGN KB:, KBI↓
$ ASSIGN PR:, PRI↓
$ RUN UPL↓
```

At this time, UPL looks on the disk for file DISK.UPL. If not found, a 1024-block contigeous file is created for storage of the entire tape preparation system. If a contigeous area of this size is not available, a DOS error message will be typed on the teleprinter and installation aborted. After the file is created, B2UDS (initial operating system) is automatically installed via the high-speed tape reader, and the following message is typed:

```
0020 3535
PL00 0 0170600 0020 3535
 15-JAN-74  UDS  V1RS
$
```

7. Load paper tapes PL0-PL3, PL10-PL16, PL20-PL40, and PL77, using UDS operating commands.

```
$ LOAD/0 ⌴   Load tape (latest version).
0020 3535
PL00 0 017060 0020 3535 5-SEP-74 UDS U2R0
$ LOAD/1⌴   Load tape 1.
0000 3022
PL01 0 0165400 0000 3022 21-DEC-74
$ Continue locating until all tapes are entered.
```

8. Reset control parameters.

```
$ P, SET/25-137602, 26-134203, 27-134407, 30-36004,
  31-134004, 46-377777⌴
  25-107402 137602
  26 4003 134203
  27 0407 134407
  30 2004 036004
  31 134404 134004
  46 00577777 00377777
```

9. Install UDS editor.

```
$ ED/↓
EDITOR
LIB BA,N,/ 400000,2,
LIBNAME, SA, EA,/ LBMACR, 402000, 450000,
LIBNAME, SA, EA,/ LBPART, 450000, 577700,
SCR SA,/240000,
PTR SA,/310000,
ASM SCR,/  400000,
SEQ BASE,/ 8,
SEQ SI,/  100,100
UAPT IJ,/ 53
RETURN,/77,
EDITOR PL NBR,/ 77,
AUTO LIB DEL? Y OR N,/ Y
SAVE EDITOR 77
```

```
E02 1220
$ SA/77 _
  PL77 0 010 1600 0000 6635 25-FEB-75 EDITOR 7
Exit operating system
$ EX/␣
$
```

10. The system is now installed on the disk. Invariably, due to user or hardware, the contents of the disk will be lost and the system must be rebuilt. To make this easier, a backup system should be created on DECtape. This provision can be made by observing the following procedure:

    a. Mount a formatted tape on Unit 0 and set controls to "WRITE" and "REMOTE". Zero the DECtape and enter a command control file for each APT MACRO that is to be installed in the editor's library using PIP logged in under uic 200,200.

```
$ RUN PIP↓
PIP-11 V7A01
# DT0:/ZE↓  (Zero DECtape on Unit 0.)
# DT0: CMD0 < KB:↓  (Enter 1st command file.)
$ ED/↓
RC, MACRO0↓  (MACRO0 is name of first MACRO.)
$ ↓
$ EX/↓
↑C    (↑C is a Control C key.)
. END ↓↑    (↓↑ is a return followed by a line feed.)
# DT0:CMD1 < KB:↓  (Enter 2nd command file.)
$ ED/↓
RC, MACRO1↓  (MACRO1 is name of 2nd MACRO.)
$↓
EX/↓
↑C
. END↓↑
#
```

    Continue entering command files for each MACRO. When this is finished, proceed to Step b.

    b. Input batch control file on DECtape 0 (UPL.DOS). PIP is still being used.

```
# DT0:UPL.DOS < KB:↓
$ JOB UNIAPT [200,200] ↓
$ RUN MACRO↓
# UPLICB,LP: < DT0:UPLICB↓
$ RUN LINK↓
# UPL,LP: < UPLICB,DT0: UPL/E↓
$ RUN UPL↓
```

Note: Write the three following statements for each MACRO that will reside in the UPL editor library:

```
$ AS DT0:CMD0,KBI↓
$AS DT0:MACRO0,CRI↓     (MACRO0)
$ RU UPL↓
$ AS DT0:CMD1,KBI↓
$ AS DT0:MACRO1,CRI↓    (MACRO1)
$ RU UPL↓
```

Continue this for each APT MACRO to be inputted to the library.

```
$ FINISH↓
↑C
. END↓↑
#
```

c. Input each MACRO that is to reside in the editor library using PIP.

```
# DT0:MACRO0 < CR:↓
# DT0:MACRO1 < CR:↓
#
```

Continue this operation until all MACROs reside on DECtape. This operation assumes that the MACRO source is a card deck. If the MACROs happen to be on another tape, use "DTX:MACROX" instead of "CR": in the previously listed commands. In order to speed up the system building, store MACROs in ascending order using program length as basis of judgement.

d. Create a UPL command file using PIP under uic 200,200.

```
# DT0:FILNAM.KBI < KB:↓
$ NAME/'UNIAPT'↓
$ LOAD/0-3, 10-16, 20-40, 77↓
$ P, SET/25-137602, 26-134203, 27-134407, 30-36004,
  31-134404, 46-377777↓
$ ED/↓
I, 400000, 2↓
LBMACR, 402000, 450000↓
LBPART, 450000, 577700↓
240000↓
310000↓
400000↓
8↓
100, 100↓
53↓
77↓
77↓
```

```
Y↓
$ SA/77↓
$ EXIT/↓
_↓
↑C
_ END↓
#
```

e. Store B2UDS paper tape or initial UDS operating system on tape using PIP.

```
# DT0:FILNAM.PRI < PR:↓
# ↑C
_ KILL↓
$
```

f. Store the entire UDS system on DECtape using the UDS operating system.

```
$ ASSIGN DT0:UNIAPT.PRI, PPO↓
$ RUN UPL↓
$ PUNCH/0-3,10-16, 20-40, 77
  PL00 0 0170600 0020 3535 5-SEP-74
  PL01 0 0165400 0000 3022 31-DEC-74
```

These data are listed for each tape as they are being written on the DECtape.

```
PL77 0 0101600 0000 6635 25-FEB-75
$ EX/_
$
```

The software is now backed up on DECtape. To reinstall the system, the user only has to type the following commands (assuming DISK.UPL does not reside on disk) with

```
$ BA DT0:UPL.DOS↓
```

this backup tape on Unit 0. The system will be built, the editor installed, and all MACROs will automatically be stored on disk without any further operator intervention.

In actual practice, the user does not have to go through this entire operation when creating a new backup from updated modules that are periodically received from UCC. If no new, interpreter or B2UDS tape is received, skip Operations a and d and carry on the following procedure to acquire an up-to-date system and backup tape:

a.
```
$ LOGIN 200,200↓
$ RUN PIP↓
PIP-11  7401
#
```

Delete UNIAPT.PRI, UPL.OBJ, UPLICB, or FILNAM.PRI if a new tape is issued for any of these by:

{ UPL.OBJ - long interpreter tape
UPLICB.MAC - short interpreter tape
FILNAM.PRI - B2UDS tape }

using the following commands with backup tape mounted on Unit 0 with controls set to "WRITE' and "REMOTE".

```
# DT0: UNIAPT.PRI/DE↓
# DT0: UPL.OBJ/DE↓
# DT0: UPLICB.MAC/DE↓
# DT0: FILNAM.PRI/DE↓
```

Mount new short interpreter tape in reader.

```
# DT0: UPLICB.MAC < PR:↓
```

Mount new long interpreter tape in reader.

```
# DT0: UPL.OBJ < PR:↓
```

Mount new B2UDS tape in reader.

```
# DT0: FILNAM.PRI < PR:↓
# ↑C
_ KILL↓
#
```

b. Load new PLXX tapes into system using UDS commands.

```
$ ASSIGN PR:,PRI↓
$ RUN UPL↓
```

Load tape PLXX into reader,

```
$ LOAD/XX ⎵
$
```

Perform the operation just listed for each new tape, where XX is the tape PL number.

```
$ EX/⎵
$
```

c. Write an updated version of the system on tape files UNIAPT.PRI.

```
$ ASSIGN DT0:UNIAPT.PRI, PPO
$ RUN UPL
$ PUNCH/0-3, 10-16, 20-40, 70 ⌴
$ EX/⌴
$
```

The user now has a new backup tape as well as a new system unless any of Step a was performed. If Step a was performed, do Step d.

d.
```
$ RUN PIP↓
PIP-11 V7A01
# DISK.UPL/PR:000↓  (Set priority to 0.)
# DISK.UPL/DE↓  (Delete UCC system.)
# ↑C
. KILL↓
$ BA DT0:  UPL.DOS↓  (Build new system.)
```

Postprocessor, DDNC Service Package, and EIA-to-ASCII Converter Installation

The three programs are entered from DECtape using PIP under uic 200,200. Mount the tape that contains programs on Unit 0 and perform the following operations:

```
$ RUN PIP↓
PIP-11 V7A01
# DK0:  POS922.LDA < DT0:  POS922.LDA↓
# DK0:  DDNC1.LDA < DT0:  DDNC1.LDA↓
# DK0:  EIAASC.LDA < DT0:  EIAASC.LDA↓
# ↑C
. KILL↓
$
```

Distribution

Energy Research and Development Administration - Oak Ridge

Hickman, H. D.
Leed, R. E.
Zachry, D. S., Jr

Oak Ridge Gaseous Diffusion Plant

Stief, S. S.
Wilcox, W. J., Jr

Oak Ridge Y-12 Plant

Anderson, P. J.
Arehart, T. A., Jr
Bookhart, T. W.
Bowers, G. L.
Briscoe, O. W.
Burditt, R. B.
Burkhart, L. E.
Burleson, R. R.
Duggan, H. G.
Ebert, T. H.
Foulk, D. L.
Fraser, R. J.
Jones, F. W.
Kahl, K. G.
Keith, A.
Kite, H. T. (25)
Lay, C. M. (5)
Mason, D. L.
McLendon, J. D.
Mills, J. M., Jr
Murphy, S. M., Jr
Noey, J. L.
Phillips, L. R.
Riker, R. B. (3)
Schreyer, J. M.
Smith, R. D.
Stephens, A. E.
Tewes, W. E.
Thompson, C. H.
Tilson, F. V.
Weathersby, W. E.
Webber, T. R.
Whitten, L. G., Jr
Williams, T. L. (5)
Wright, C. C.
Yaggi, W. J./Googin, J. M.
Y-12 Central Files (5)
Y-12 Central Files (master copy)
Y-12 Central Files (route copy)
Y-12 Central Files (Y-12RC)
Zerby, C. D.

Paducah Gaseous Diffusion Plant

Levin, R. W.

In addition, this report is distributed in accordance with the category UC-32, **Mathematics and Computers**, as given in the *USERDA Standard Distribution Lists for Unclassified Scientific and Technical Reports*, TID-4500.