E 80 - 10.2 9 0.

JSC-13030
Volume 3

NASA CR-
160684

# "AS-BUILT" DESIGN SPECIFICATION FOR THE CAMS IMAGE-100 HYBRID SYSTEM

Job Order 71-195

(TIRF 76-0106)

VOLUME 3

UTILITIES AND SHARED SUBROUTINES

(E80-10290) AS-BUILT DESIGN SPECIFICATION FOR THE CAMS IMAGE-100 HYBRID SYSTEM. VOLUME 3: UTILITIES AND SHARED SUBROUTINES (Lockheed Electronics Co.) 200 p HC A09/MF A01 CSCL 05B G3/43

N80-30858

Unclas
00290

Prepared By

Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION
SCIENCE AND APPLICATIONS DIRECTORATE

***National Aeronautics and Space Administration***
**LYNDON B. JOHNSON SPACE CENTER**
***Houston, Texas***

August 1977

LEC-10822
Volume 3

JSC-13030
Volume 3

# "AS-BUILT" DESIGN SPECIFICATION FOR THE CAMS IMAGE-100 HYBRID SYSTEM

Job Order 71-195

VOLUME 3

UTILITIES AND SHARED SUBROUTINES

Assembled By

L. E. Giddings

From contributions of the following persons:

R. T. Minter
K. L. Pattison
P. S. Lin
G. J. Champagne
E. J. Hightower
W. A. Holley
J. S. Huang
T. R. Kell
D. L. Loe
L. F. Robinson
R. M. Rodriguez
J. K. Rowland
C. D. Shih
H. G. Thadani
S. G. Thadani
B. R. Thompson
E. L. Wilson

APPROVED BY

[illegible]

P. L. Krumm, Supervisor
Applications Software Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

Science and Applications Directorate

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

August 1977

LEC-10822
Volume 3

## SPECIAL NOTE

The information in all three volumes of this document has been carefully checked. It is current at the time of publication, the end of August, 1977. This document will not be revised to show corrections and further changes. Rather, a new document will be issued toward the end of 1977 incorporating all changes, and making necessary corrections. The new volumes will be issued under the title: "As-Built Design Specifications for the CAMS Image-100 Hybrid System, as modified. The new document will be issued as LEC-11216 and JSC-13118.

Please bring errors and corrections to the attention of L. Giddings, 333,6311, mail code C42.

CONTENTS

**Most sections include brief descriptions and a program listing. Sections marked with two asterisks contain optional flowcharts.

+These sections refer to other documents or other volumes of this document.

Section Page

## 1. BITSET.MAC

Sets or clears a bit in an array.

- Call sequence:

```
CALL BITSET (#, A, I)
```

| Segment | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| # | I | 1 | I | Bit # to set/clear (O-N) |
| A | I | ? | I | Name of array |
| I | 1 | I | I | New value of bit |

```
;         BIT SET ROUTINE
;         CALL BITSET(N,J,V)
;         N IS THE NUMBER OF THE BIT TO SET/CLEAR, J IS THE FIRST BIT
;         IN THE ARRAY J
;         V IS A 1 OR A 0, THE VALUE TO SET BIT N OF ARRAY J.
;
;
;         CONVERTED TO BITSET FROM GETBIT BY T. KELL/LEC/ 2/22/77
;
          .TITLE  BITSET
          .GLOBL  BITSET
;
          R0=%0
          R1=%1
          R2=%2
          R5=%5
          PC=%7
;
BITSET:   MOV      @2(R5),R0
          MOV      R0,R1
          BIC      #177760,R0
          ASL      R0
          ASR      R1
          ASR      R1
          ASR      R1
          BIC      #1,R1
          ADD      4(R5),R1
          BIC      DOTS(R0),(R1)
          MOV      #17,R2
          ASR      R0
          SUB      R0,R2
          MOV      @6(R5),R0
          BIC      #177776,R0
          ASH      R2,R0
          BIS      R0,(R1)
          RTS      PC          ;MODIFIED BY SIS  - 7-22-75
;
DOTS:     +100000
          +40000,20000,10000,4000,2000,1000
          +400,200,100,40,20,10,4,2,1
          .END
```

## 2. [300,6] BLKTHM.FTN

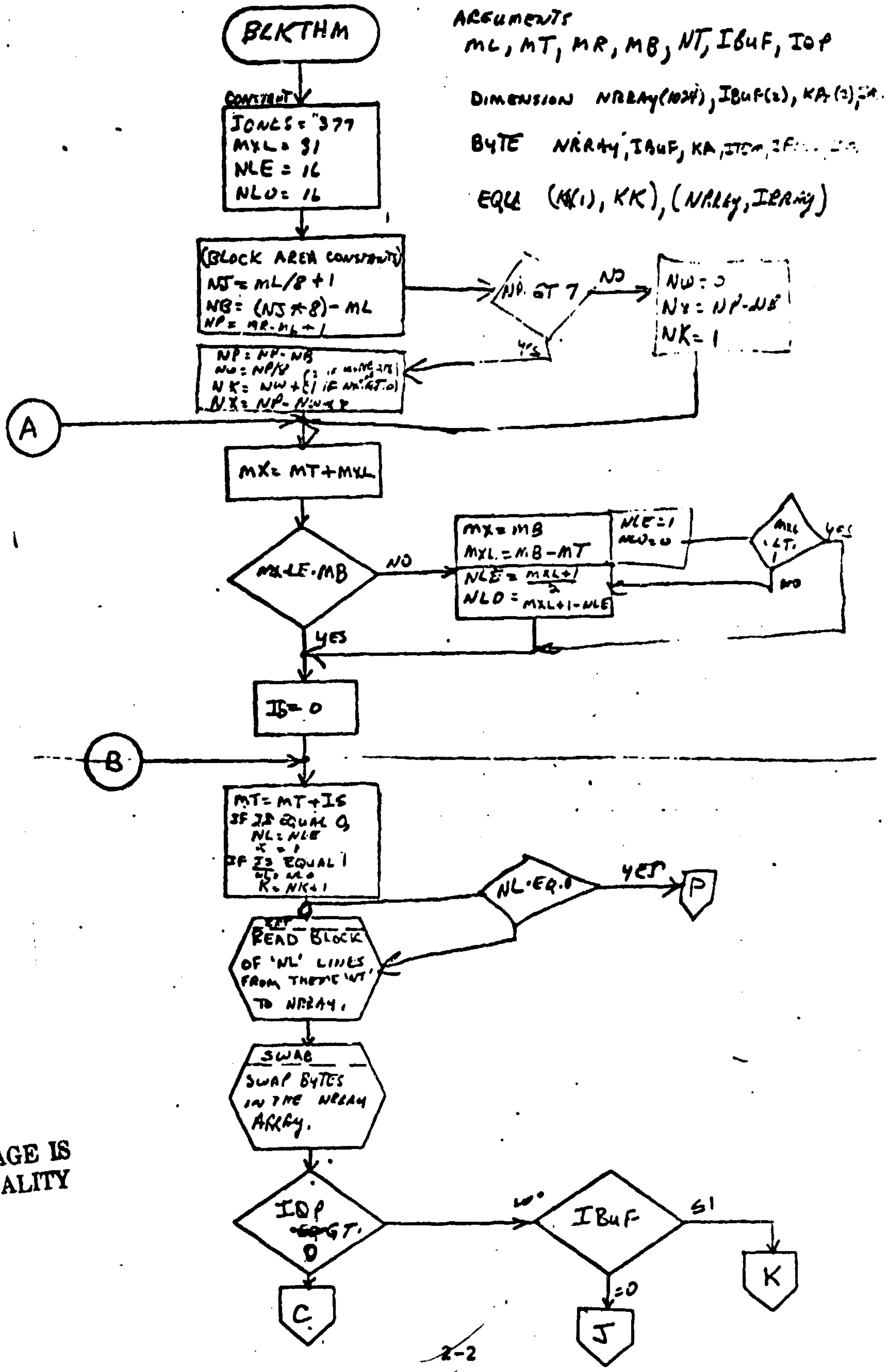
The subroutine writes a theme on a block.

- **Calling sequence:**

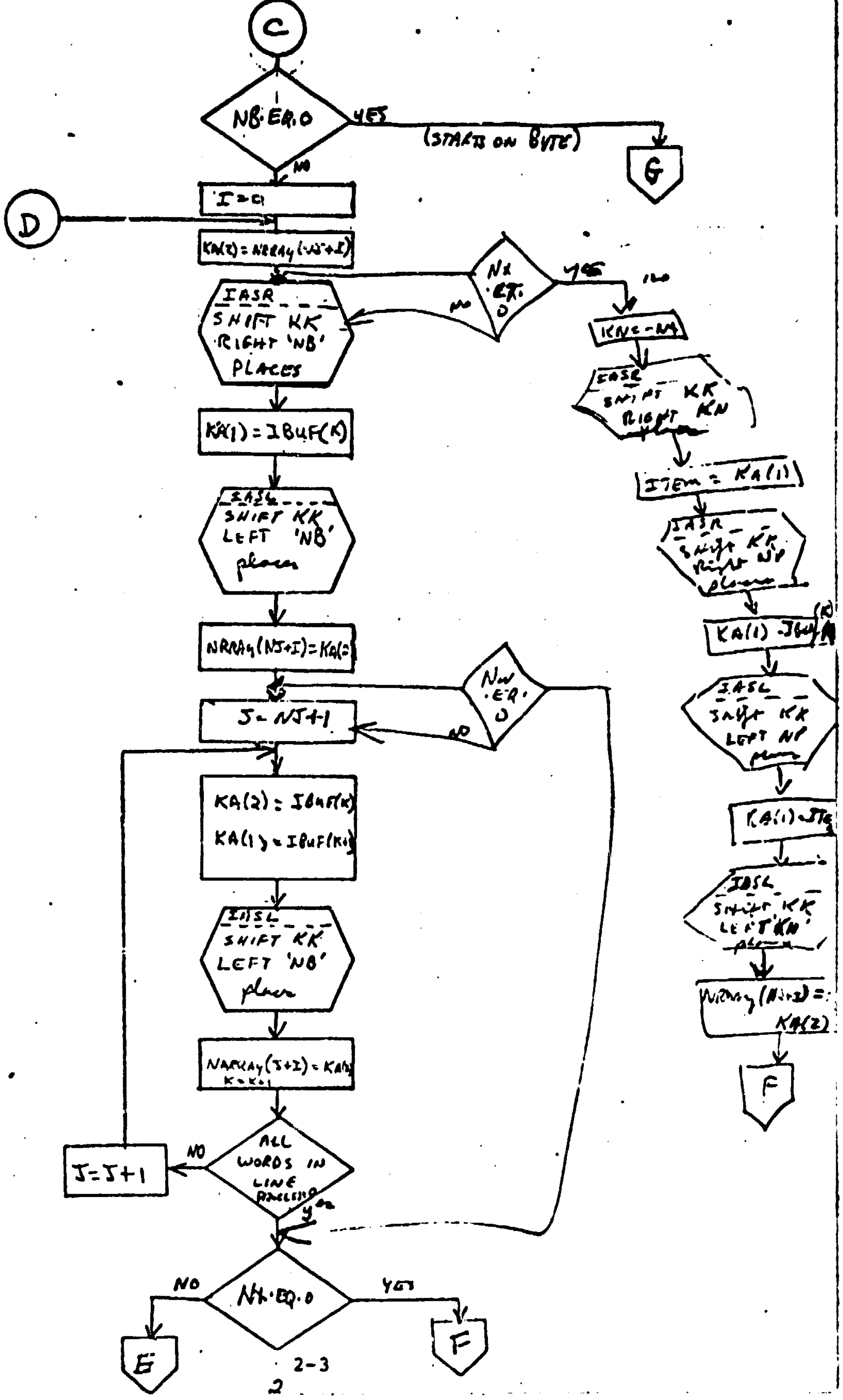
```
CALL BLKTHM (ML,MT,MR,MB,NT,lBUF,lOP)
```

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| ML | Integer | 1 | In | Left coordinate of block |
| MU | Integer | 1 | In | Upper coordinate of block |
| MR | Integer | 1 | In | Right coordinate of block |
| MB | Integer | 1 | In | Bottom coordinate of block |
| NT | Integer | 1 | In | Theme number |
| IBUF | Integer | 1 | In | Data to be transferred |
| IOP | Integer | 1 | In | 0=fill with data 1=fill with 0's if lBUF(1)=0 or fill with 1's if lBUF(1)=1 |

For details refer to section 3.5.2.5.1.8.1 of volume 1 of this document.

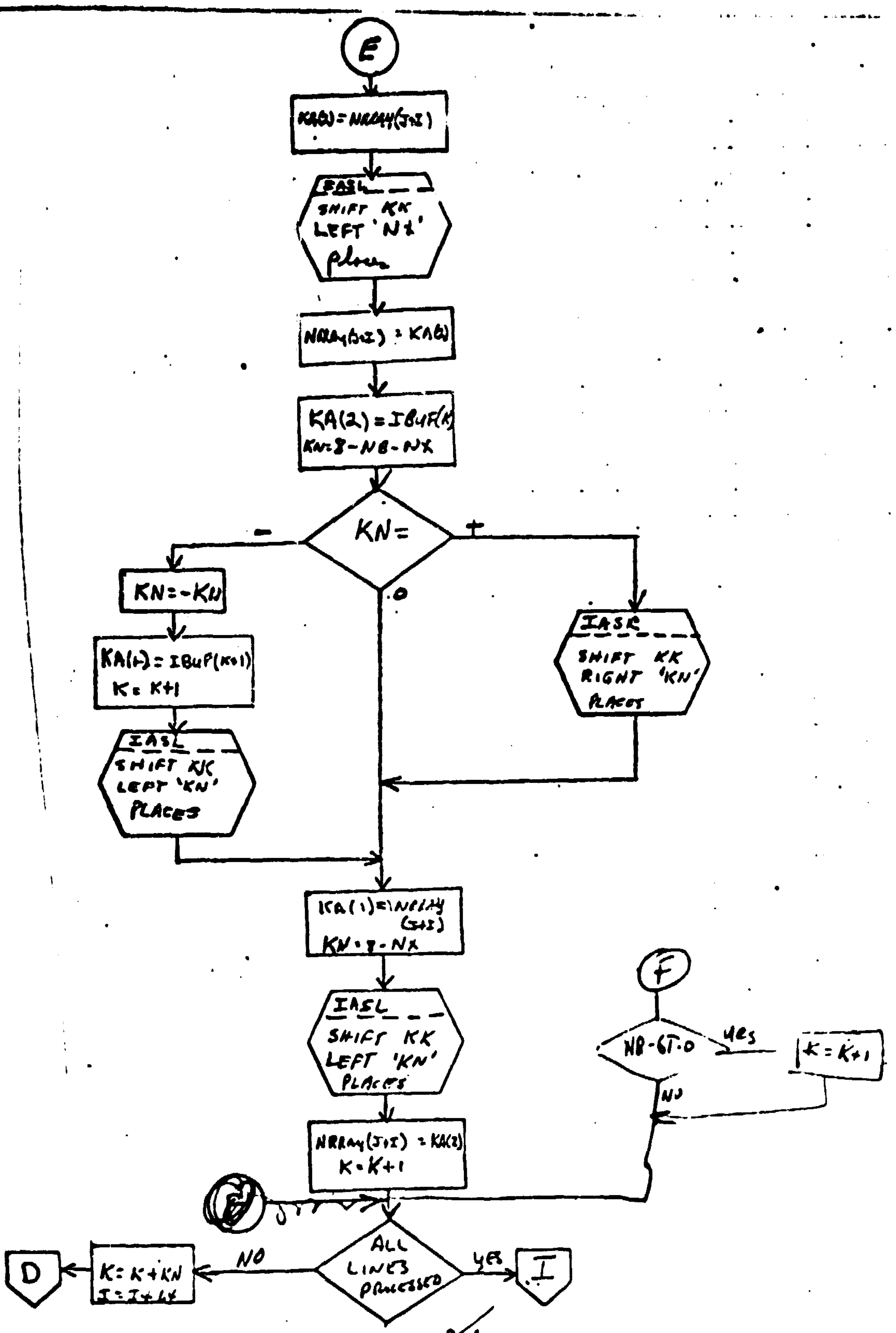


ORIGINAL PAGE IS
OF POOR QUALITY


C
NB·EQ·0
YES
(STARTS ON BYTE)
G
NO
I=0
D
KA(2)=NRRAY(NJ+I)
NX·EQ·0
YES
NO
IASR
SHIFT KK RIGHT 'NB' PLACES
KN=-NX
IASR
SHIFT KK RIGHT KN places
KA(1)=IBUF(K)
ITEM=KA(1)
IASL
SHIFT KK LEFT 'NB' places
IASR
Shift KK Right NP places
KA(1)=IBUF(K)
NRRAY(NJ+I)=KA(2)
IASL
Shift KK LEFT NP places
NW·EQ·0
NO
J=NJ+1
KA(1)=ITEM
KA(2)=IBUF(K)
KA(1)=IBUF(K+1)
IASL
SHIFT KK LEFT 'NB' place
IASL
Shift KK LEFT 'KN' places
NARRAY(J+I)=KA(2)
K=K+1
NRRAY(NJ+2)=KA(2)
F
ALL WORDS IN LINE PROCESSED
NO
J=J+1
yes
NX·EQ·0
NO
YES
E
F

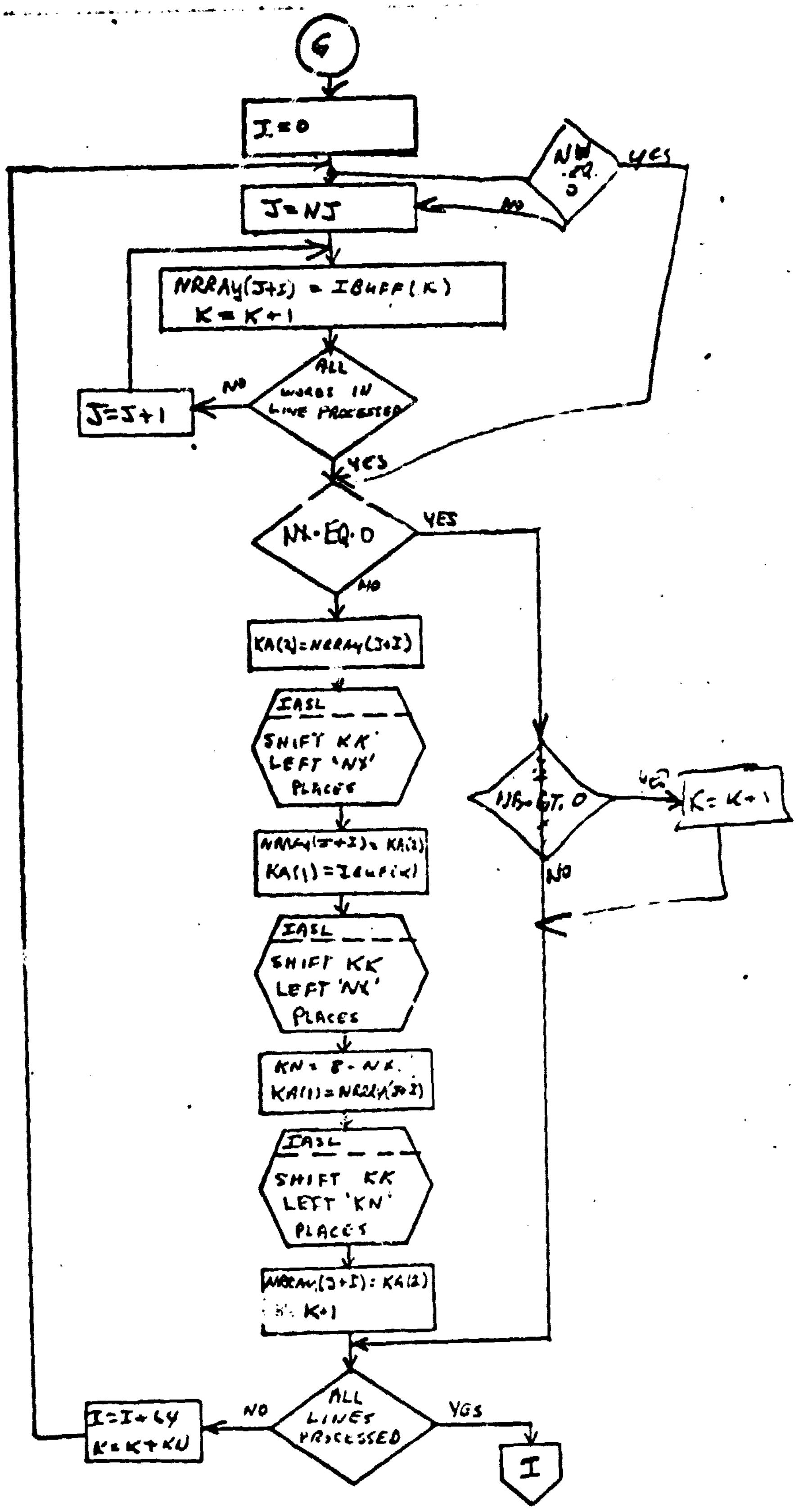
2

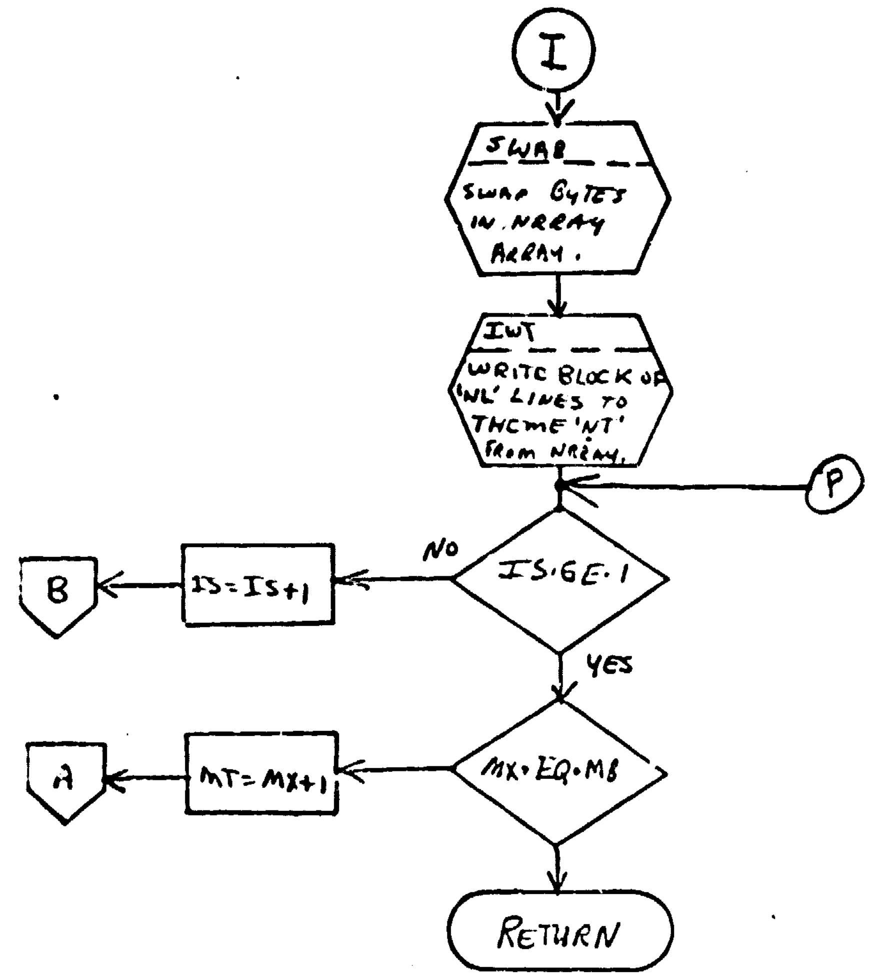
5

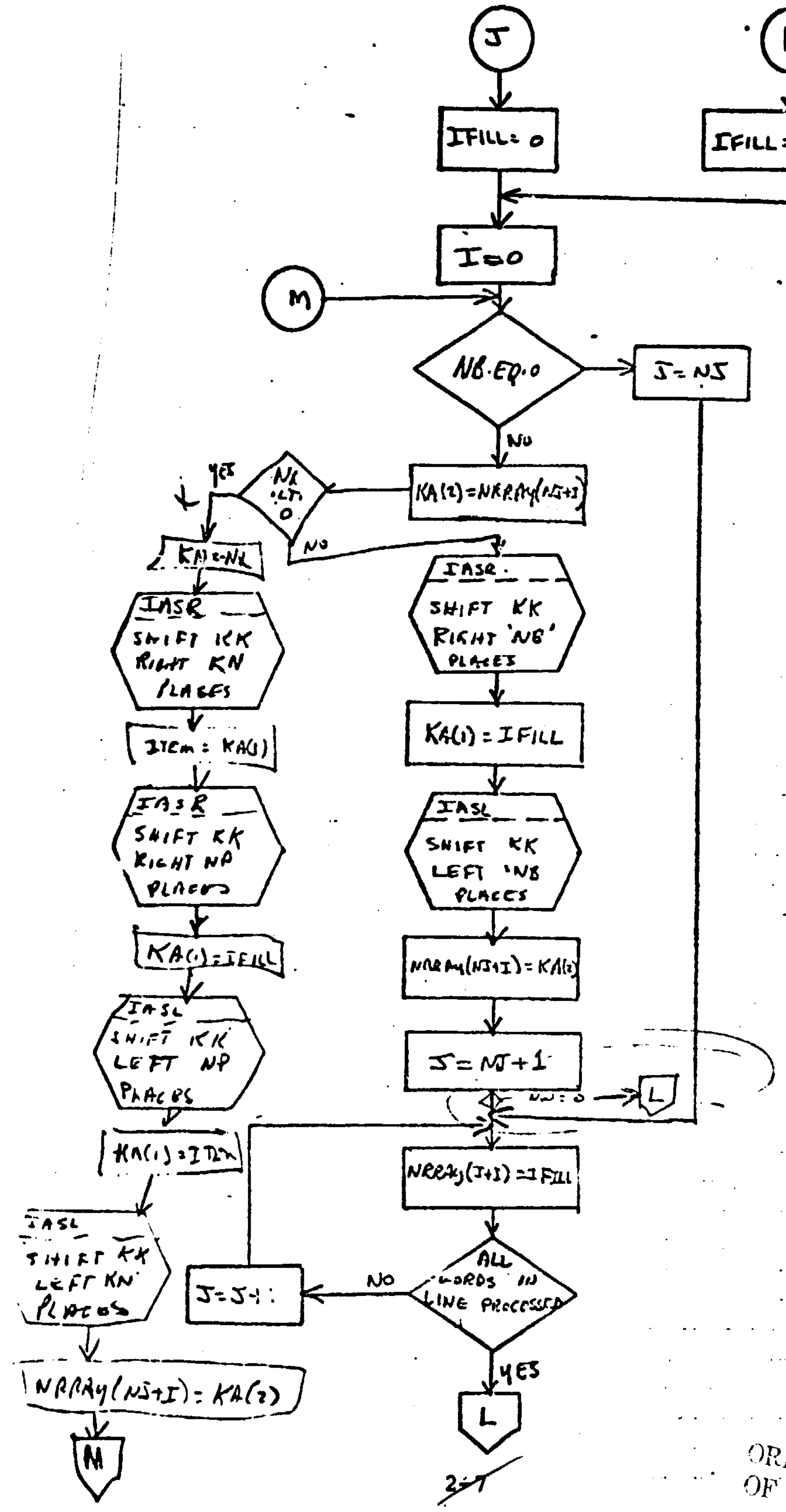

E
KA(1) = NRRAY(J+I)
IASL
SHIFT KK LEFT 'NX' places
NRRAY(J+I) = KA(1)
KA(2) = IBUF(K)
KN = 8 - NB - NX
KN =
−
+
0
KN = -KN
KA(2) = IBUF(K+1)
K = K+1
IASL
SHIFT KK LEFT 'KN' PLACES
IASR
SHIFT KK RIGHT 'KN' PLACES
KA(1) = NRRAY(J+I)
KN = 8 - NX
IASL
SHIFT KK LEFT 'KN' PLACES
NRRAY(J+I) = KA(2)
K = K+1
F
NB .GT. 0
yes
K = K+1
NO
ALL LINES PROCESSED
NO
yes
K = K+KN
J = J+LX
D
I

~~2-4~~

6

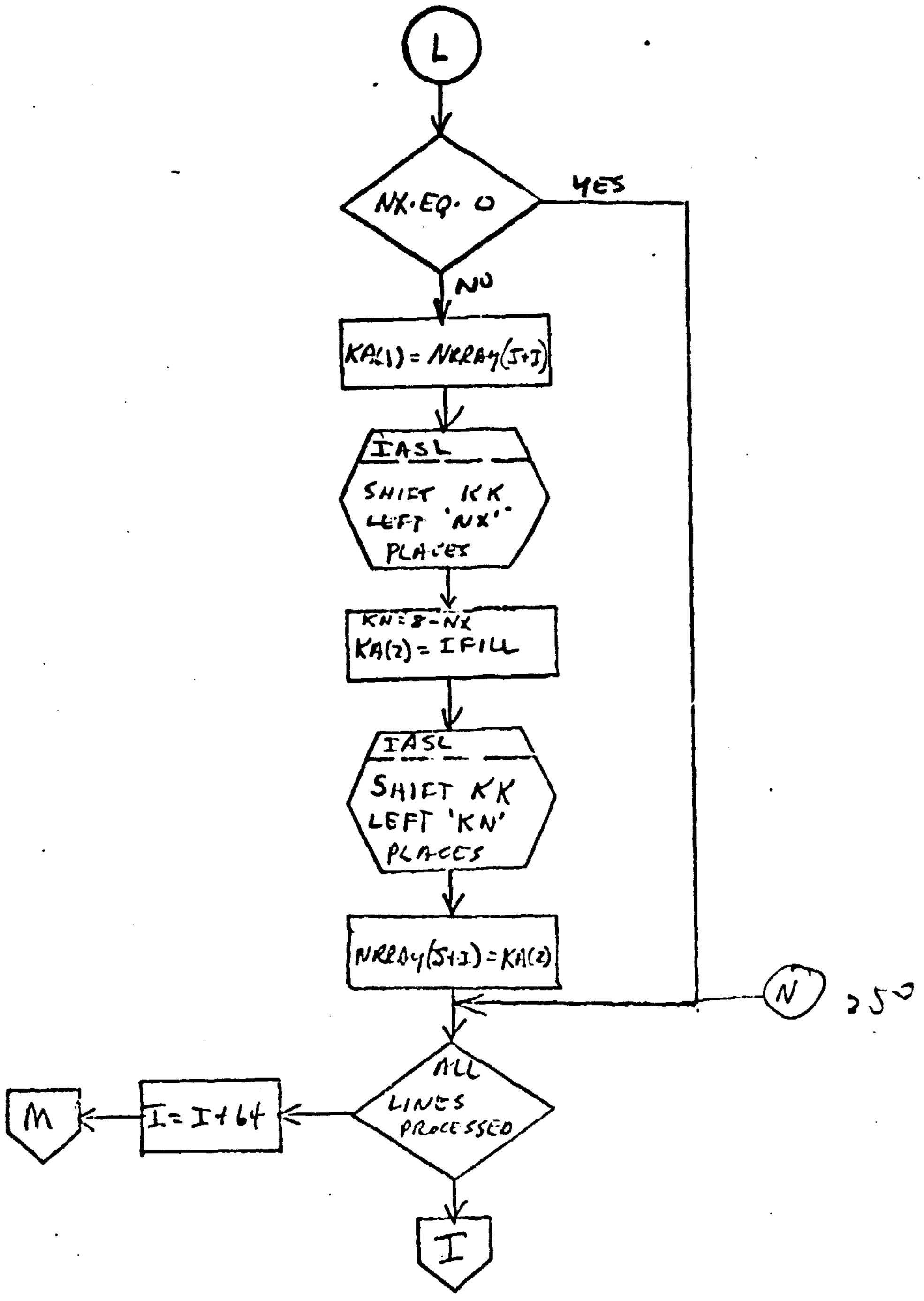

G
I=0
NW .EQ. 0
YES
NO
J=NJ
NRRAY(J+I) = IBUFF(K)
K=K+1
ALL WORDS IN LINE PROCESSED
NO
J=J+1
YES
NX.EQ.0
YES
NO
KA(2)=NRRAY(J+I)
IASL
SHIFT KK LEFT 'NX' PLACES
NRRAY(J+I)=KA(2)
KA(1)=IBUFF(K)
IASL
SHIFT KK LEFT 'NX' PLACES
KN = 8 - NX
KA(1)=NRRAY(J+I)
IASL
SHIFT KK LEFT 'KN' PLACES
NRRAY(J+I)=KA(2)
K=K+1
NB.GT.0
YES
K=K+1
NO
ALL LINES PROCESSED
NO
I=I+64
K=K+KU
YES
I


I
SWAB
SWAP BYTES IN NRRAY ARRAY.
IWT
WRITE BLOCK OF 'NL' LINES TO THEME 'NT' FROM NRRAY.
P
NO
IS.GE.1
IS=IS+1
B
YES
MX.EQ.MB
MT=MX+1
A
RETURN



~~2-7~~

ORIGINAL PAGE IS OF POOR QUALITY


L
NX·EQ·0
YES
NO
KA(1) = NRRAY(J+I)
IASL
SHIFT KK LEFT 'NX' PLACES
KN = 8 - NX
KA(2) = IFILL
IASL
SHIFT KK LEFT 'KN' PLACES
NRRAY(J+I) = KA(2)
N
250
ALL LINES PROCESSED
I = I + 64
M
I

## BLKTHM

Arguments:

ML - left coordinate of block

MT - top coordinate of block

MR - right coordinate of block

MB - bottom coordinate of block

NT - theme number

IBUF - data array (buffer)

```
IOP - option = 0, fill block with data in IBUF(1)
             = 1, fill block with 0's if IBUF(1) = 0
                  or fill block with 1's if IBUF(1) = 1
```

ARRAYS/CONSTRAINTS

| | |
|---|---|
| NRRAY | work area 1024 bytes long |
| IRRAY | equivalenced to NRRAY to perform swap. |

```
FØRTRAN IV-PLUS V02-04                14:14:30    31-AUG-77          PAGE 1
BLKTHM.FTN       /TR:BLØCKS/WR
      SUBRØUTINE BLKTHM(ML,MU,MR,MB,MT,IBUF,IØP)
C
C  RØUTINE TØ PUT LABELS ØN THEMES
C
      DIMENSIØN NRRAY(1024),IBUF(1),KA(2),IRRAY(512)
      BYTE NRRAY,IBUF,KA,ITEM
      BYTE IFILL,IØNES
      EQUIVALENCE (NRRAY,IRRAY),(KK,KA)
C
        IØNES="377
      MXL=31
      NLE=16
      NLØ=16
        K1=1
        MT=MU
C
C   BLØCK CØNSTANTS
      NJ=ML/8+1
      NB=NJ*8-ML
      NP=MR-ML+1
      IF(NP.GT.7)GØ TØ 5
      NW=0
      NX=NP-NB
      NK=1
      GØ TØ 10
    5 NP=NP-NB
      NW=NP/8
      NX=NP-NW*8
      NK=NW
      MX=NX+NB
      IF(MX.GT.0)NK=NK+1
      IF(MX.GT.8)NK=NK+1
C
C
C  PRØCESS AN AREA ØN THEME
C
   10 MX=MT+MXL
      IF(MX.LE.MB)GØ TØ 20
      MX=MB
      MXL=MB-MT
       NLE=1
       NLØ=0
       IF(MXL.LT.1)GØ TØ 20
      NLØ=(MXL+1)/2
      NLE=MXL+1-NLØ
   20 CØNTINUE
C
C   PRØCESS EVEN LINES ,THEN  ØDD LINES
C
      DØ 100 IS=0,1
      MT=MT+IS
      NL=NLE
      K=K1
      IF(IS.EQ.1)NL=NLØ
      IF(IS.EQ.1)K=NK+K1
      IF(NL.EQ.0)GØ TØ 100
      C
      C     READ LINES FRØM THEME 'NT'
      C
            CALL IRT(NT,MT,NL,NRRAY)
            CALL WAIT
      C
      C     SWAP BYTES IN INPUT ARRAY
      C
            LX=NL*32
            DØ 30 L=1,LX
             CALL SWAB(IRRAY(L))
   30       CØNTINUE
              IF(ISP.GT.0)GØ TØ 200
      C
      C     ADD LABEL TØ LINES
      C
            IF(NB.EQ.0)GØ TØ 60
            IL=(NL*64)-1
            DØ 50 I=0,IL,64
      C
            KA(2)=NRRAY(NJ+I)
            IF(NX.LT.0)GØ TØ 120
            KK = IASR(KK,NB)
      C
      C
            KA(1)=IBUF(K)
            KK = IASL(KK,NB)
      C
            NRRAY(NJ+I)=KA(2)
            IF(NW.EQ.0)GØ TØ 41
            NJJ=NJ+1
             DØ 40 J=NJJ,NJJ+NW-1
            KA(2)=IBUF(K)
            KA(1)=IBUF(K+1)
      C
            KK = IASL(KK,NB)
            NRRAY(J+I)=KA(2)
            K=K+1
   40       CØNTINUE
      C
   41       IF(NX.EQ.0)GØ TØ 49
            JT=J+I
            KA(2)=NRRAY(JT)
            KK = IASL(KK,NX)
            NRRAY(JT)=KA(2)
      C
             KA(2)=IBUF(K)
            KN=8-NB-NX
            IF(KN)42,46,44
   42       KN=-KN
            KA(1)=IBUF(K+1)
            K=K+1
            KK = IASL(KK,KN)
            GØ TØ 46
      C
   44       KK = IASR(KK,KN)
```

```
C
   46 CØNTINUE
      KA(1)=NRRAY(JT)
C
      KN=8-NX
      KK = IASL(KK,KN)
C
      NRRAY(JT)=KA(2)
      K=K+1
        GØ TØ 50
   49   IF(NB.GT.0)K=K+1
C
   50 K=K+NK
      GØ TØ 90
C
C
C
C     STARTS ØN A BYTE
C
   60 IL=(NL*64)-1
C
      DØ 80 I=0,IL,64
      IF(NW.EQ.0)GØ TØ 71
C
        DØ 70 J=NJ,NJ+NW-1
        NRRAY(J+I)=IBUF(K)
   70   K=K+1
C
   71   IF(NX.EQ.0)GØ TØ 79
        JT=J+I
        KA(2)=NRRAY(JT)
        KK = IASL(KK,NX)
        NRRAY(JT)=KA(2)
        KA(1)=IBUF(K)
      KK = IASL(KK,NX)
C
      KN=8-NX
        KA(1)=NRRAY(JT)
        KK = IASL(KK,KN)
        NRRAY(JT)=KA(2)
        K=K+1
        GØ TØ 80
C
  79    IF(NB.GT.0)K=K+1
   80 K=K+NK
   90 IF(IS.EQ.0)K2=K
C
C    SWAP BYTES IN ARRAY BEFORE OUTPUTTING
C
      DØ 95 L=1,LX
      CALL SWAB(IRRAY(L))
   95 CØNTINUE
C
C     WRITE LINES TØ THEME 'NT'
C
      CALL INT(NT,MT,NL,NRRAY)
```

```
            CALL WAIT
      100   CØNTINUE
              K1=K2
C
            IF(MX.EQ.MB)GØ TØ 140
              MT=MX+1
              GØ TØ 10
C
C             LESS THAN 7 PIXELS
C
C
      120   KN=NX
            KK=IASR(KK,KN)
            ITEM=KA(1)
            KK=IASR(KK,NP)
            KA(1)=IBUF(K)
            KK=IASL(KK,NP)
             KA(1)=ITEM
            KK=IASL(KK,KN)
            NRRAY(NJ+I)=KA(2)
            GØ TØ 50
C
      140   CØNTINUE
            RETURN
C
C                         ** ERRØRS **
C           WRITE(6,151)
C     151   FØRMAT(' **ERRØRS ENCØUNTERED, PRØGRAM RETURNS IFLG=-1 **')
C           RETURN
      200   CØNTINUE
             IFILL=0
            IF(IBUF(1).EQ.1)IFILL=IØNES
            IL=(NL*64)-1
            DØ 250 I=0,IL,64
            IF(NB.NE.0)GØ TØ 210
            NJJ=NJ
            GØ TØ 230
      210   KA(2)=NRRAY(NJ+I)
             IF(NX.LT.0)GØ TØ 245
            KK=IASR(KK,NB)
            KA(1)=IFILL
            KK=IASL(KK,NB)
             NRRAY(NJ+I)=KA(2)
             NJJ=NJ+1
      230    IF(NW.EQ.0)GØ TØ 241
              DØ 240 J=NJJ,NJJ+NW-1
             NRRAY(J+I)=IFILL
      240   CØNTINUE
      241   IF(NX.EQ.0)GØ TØ 250
             KA(1)=NRRAY(J+I)
             KK=IASL(KK,NX)
             KN=8-NX
             KA(2)=IFILL
            KK=IASL(KK,KN)
             NRRAY(J+I)=KA(2)
            GØ TØ 250
```

```
FORTRAN IV-PLUS V02-04              14:14:30    31-AUG-77           PAGE 5
BLKTHM.FTN         /TR:BLOCKS/WR
        245  KN=-NX
             KK=IASR(KK,KN)
             ITEM=KA(1)
             KK=IASR(KK,NP)
             KA(1)=IFILL
             KK=IASL(KK,NP)
             KA(1)=ITEM
             KK=IASL(KK,KN)
             NRRAY(NJ+I)=KA(2)
        250 CONTINUE
             GO TO 90
            END
```

~~2-14~~

16

# 3. CSGDPH.FTN

## 3.1 ENTRY POINT - CSGDPH

Clears Screen, Gets Date, and Prints Header.

- Calling sequence

  CALL CSGDPH (IO,P,N,S)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| IO | I | 1 | I | TI: Lun |
| P | I | 1 | IO | Page number, incremented on exit |
| N | B | S | I | Program title |
| S | I | 1 | I | Size of N |

```
FORTRAN IV-PLUS V02-04                      14:15:25    31-AUG-77           PAGE 1
CSGDPH.FTN        /TR:BLOCKS/WR
                  SUBROUTINE CSGDPH(IO,P,N,S)
                  INCLUDE 'TTP.INC'
         *  CICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCIC
         *  C                                                                 C
         *  C                                                                 C
         *  C                                                                 C
         *  C                                                                 C
            C                    CLEAR SCREEN, GET DATE/PAGE HEADER           C
            C                    T. KELL/LEC/4/77                             C
            C                    CSGDPH.FTN                                   C
                  INCLUDE 'BCT.INC'
         *  C                                                                 C
         *  C                                                                 C
         *  C                                                                 C
         *  C                                                                 C
         *  CICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCICCIC
                  INTEGER P,S
                  BYTE N(S),TIM(8)
                  CALL IDATE(I,J,K)
                  CALL TIME(TIM)
                  CALL OUTPUT(27,12,7)
                  IX=(58-S)/2
                  IY=(59-S)/2
                  WRITE(IO,1000)I,J,K,TIM,(N(M),M=1,S),P
        1000      FORMAT(58X,'DATE: ',I2,'/',I2,'/',I2/58X,'TIME: ',8A1/
                  1 <IX>X,<S>A1,<IY>X,'PAGE: ',I2/)
                  P = P + 1
                  RETURN
                  END
```

# 4. [300,6] DTCLIO.FTN

## 4.1 ENTRY POINT - DOTIN

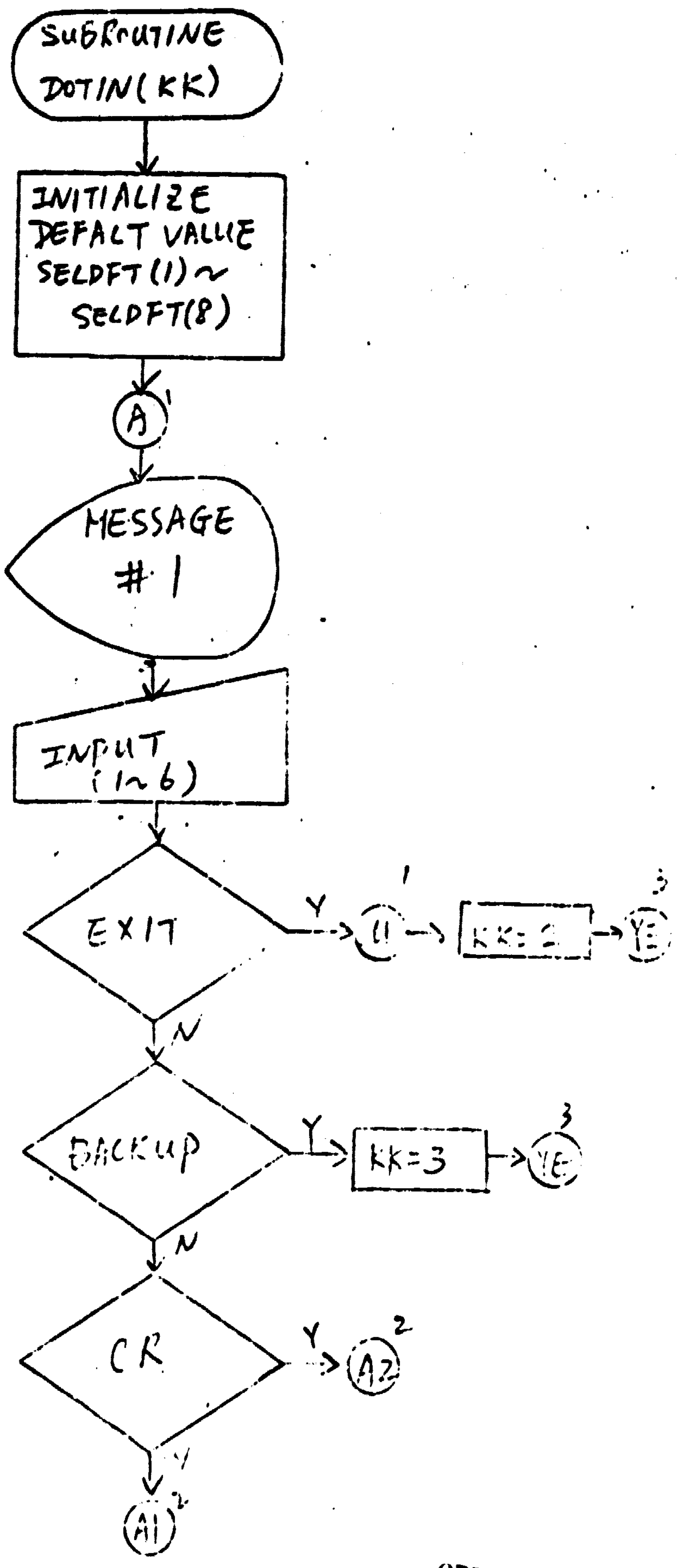
The subroutine DTCLIO uses the subroutine DOTIN to select the dot group of interest according to one of the following dot selection rules:

1. All 209 dots.
2. Unlabeled dots from the random sequence.
3. Dots by type, analyst label and clsssifier label.
4. Bias correction dots by the classification proportion.
5. Starting dots
6. DO/DU dots.

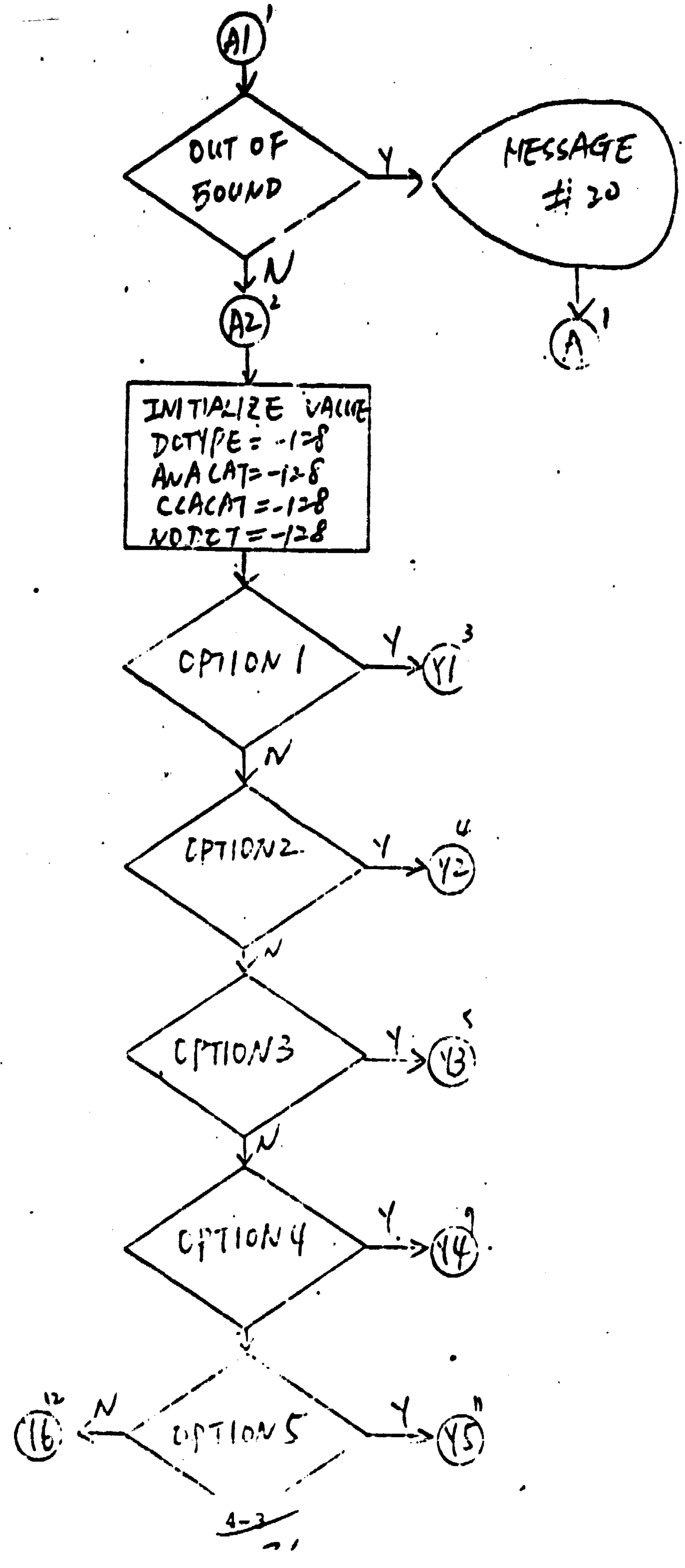
- Calling sequence

CALL DOTIN (IO,KK)

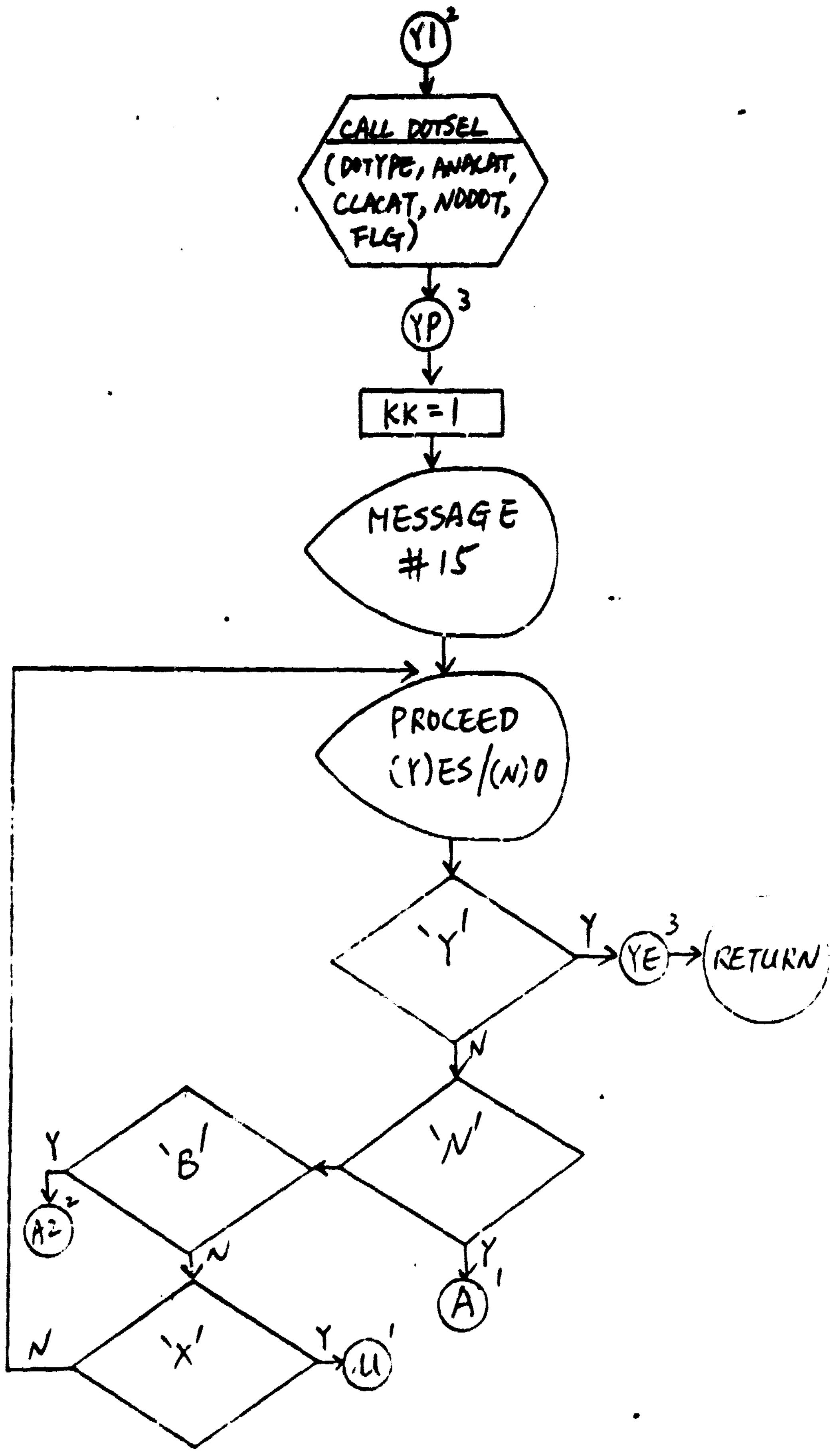
| Argument | Type | Dimensions | In/Out | Description |
|---|---|---|---|---|
| IO | Integer | - | In | Input-output unit |
| KK | Integer | - | Out | 1=normal return<br>2=exit<br>3=backup |

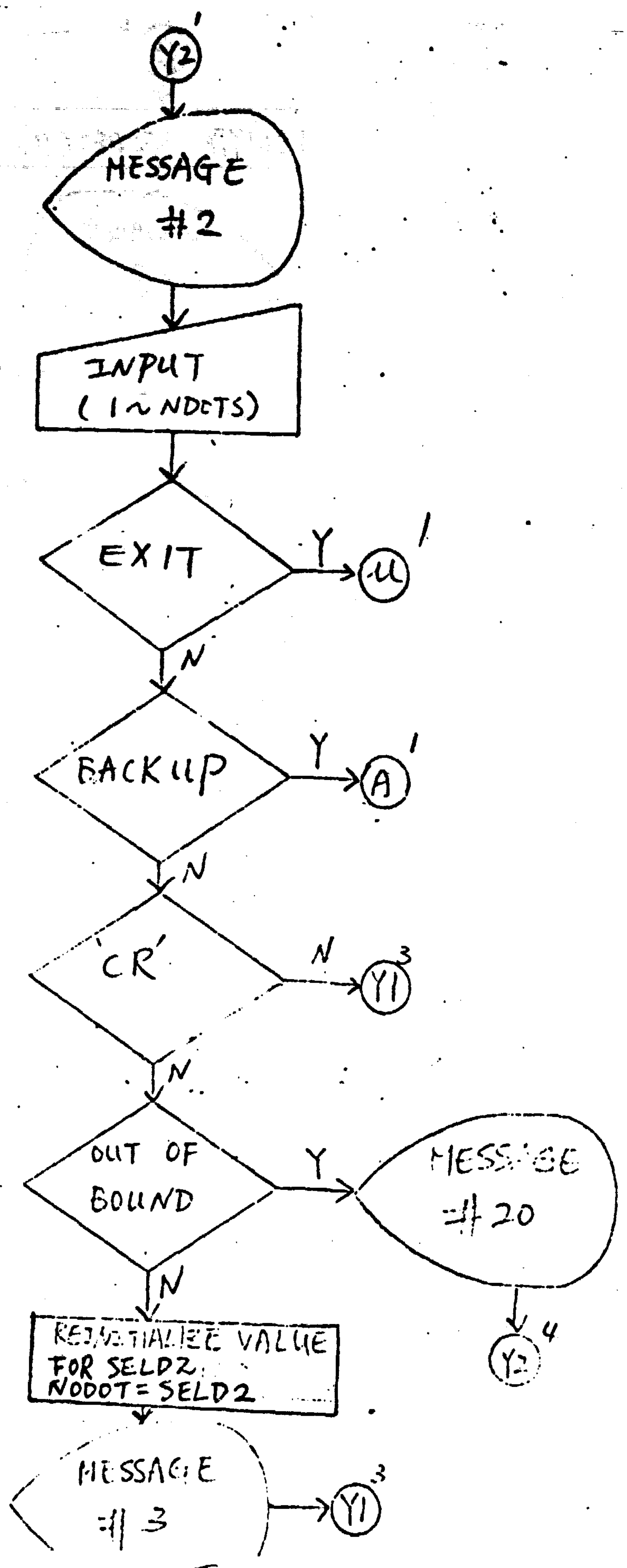


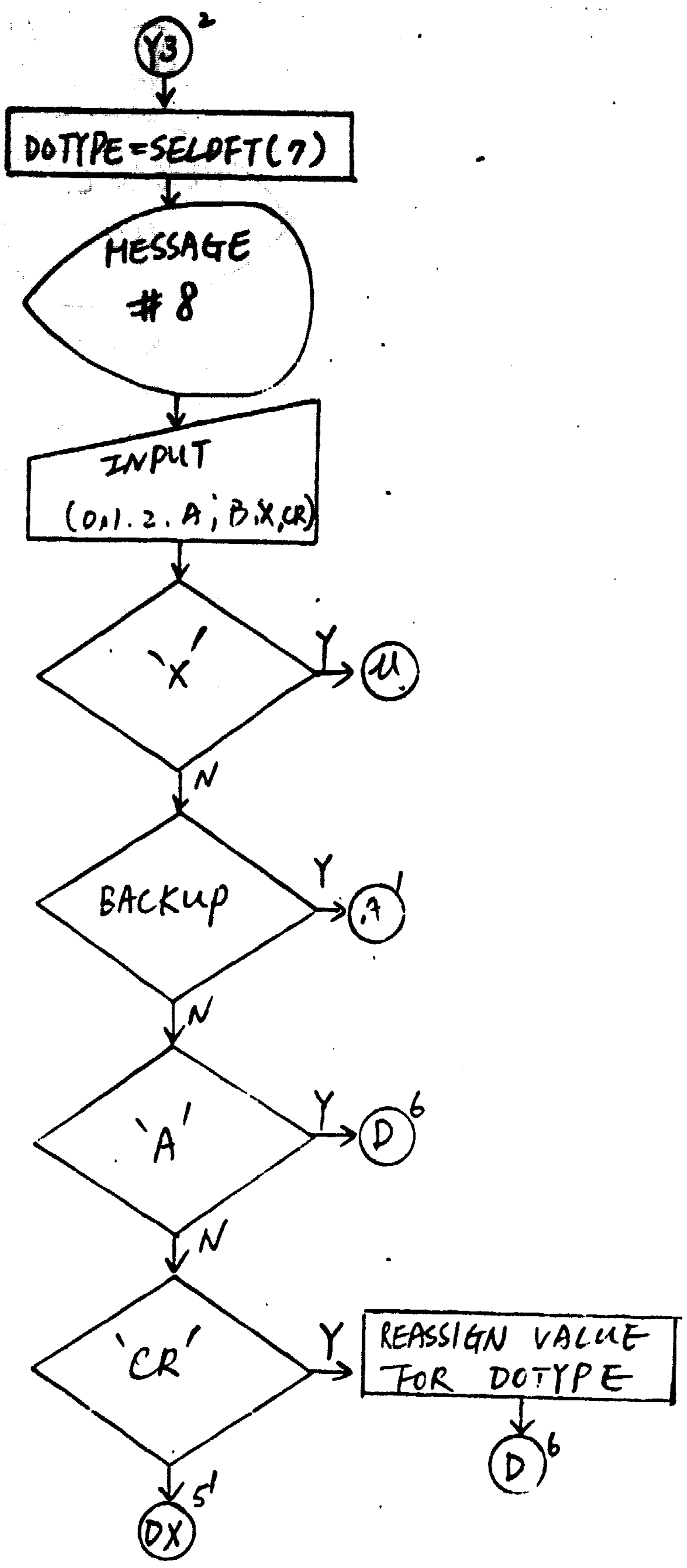
ORIGINAL PAGE IS OF POOR QUALITY



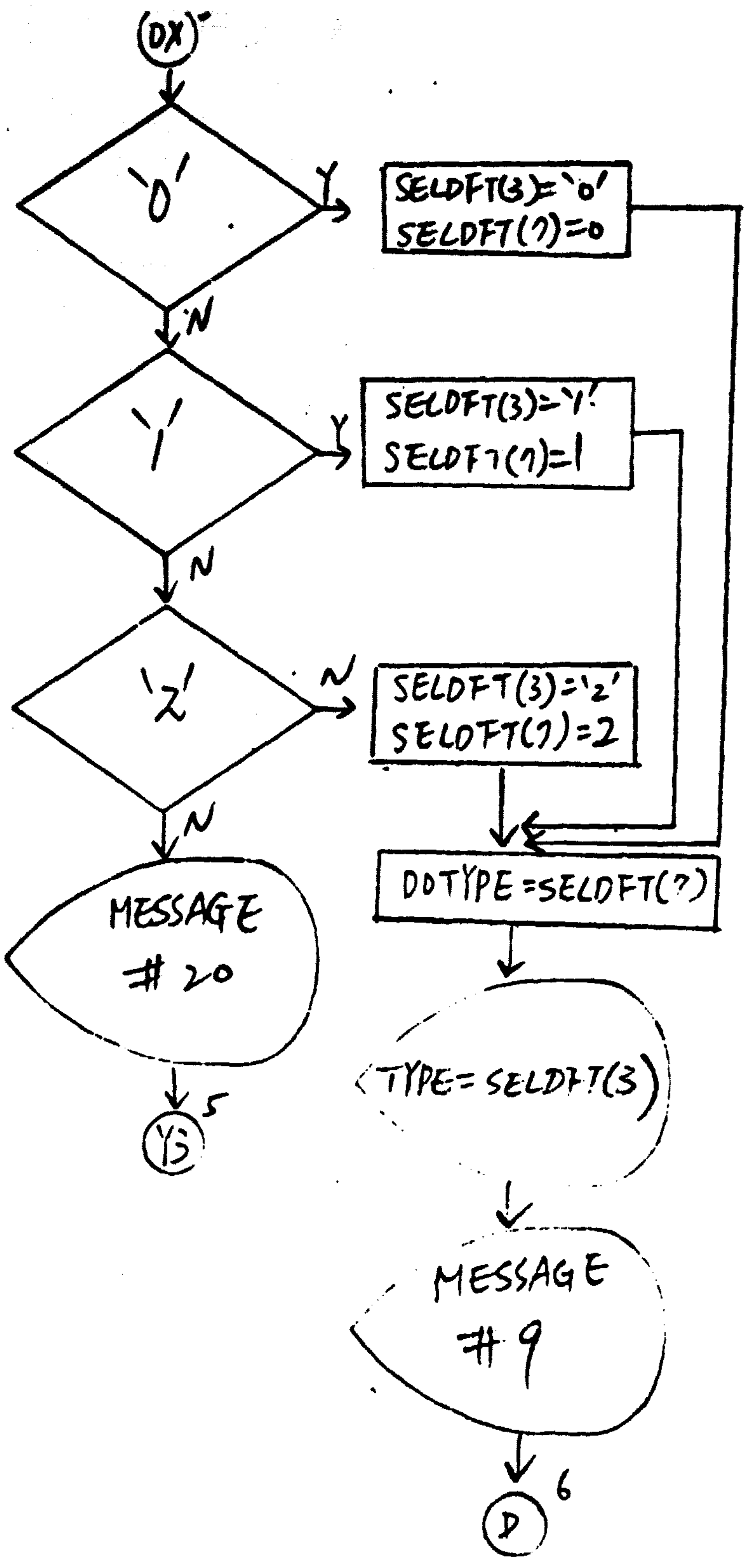
A1 1
OUT OF BOUND
Y
MESSAGE #20
A 1
N
A2 2
INITIALIZE VALUE
DCTYPE = -128
ANACAT = -128
CLACAT = -128
NODCT = -128
OPTION 1
Y
Y1 3
N
OPTION2
Y
Y2 4
N
OPTION3
Y
Y3 5
N
OPTION4
Y
Y4 9
OPTION 5
N
16 12
Y
Y5 11

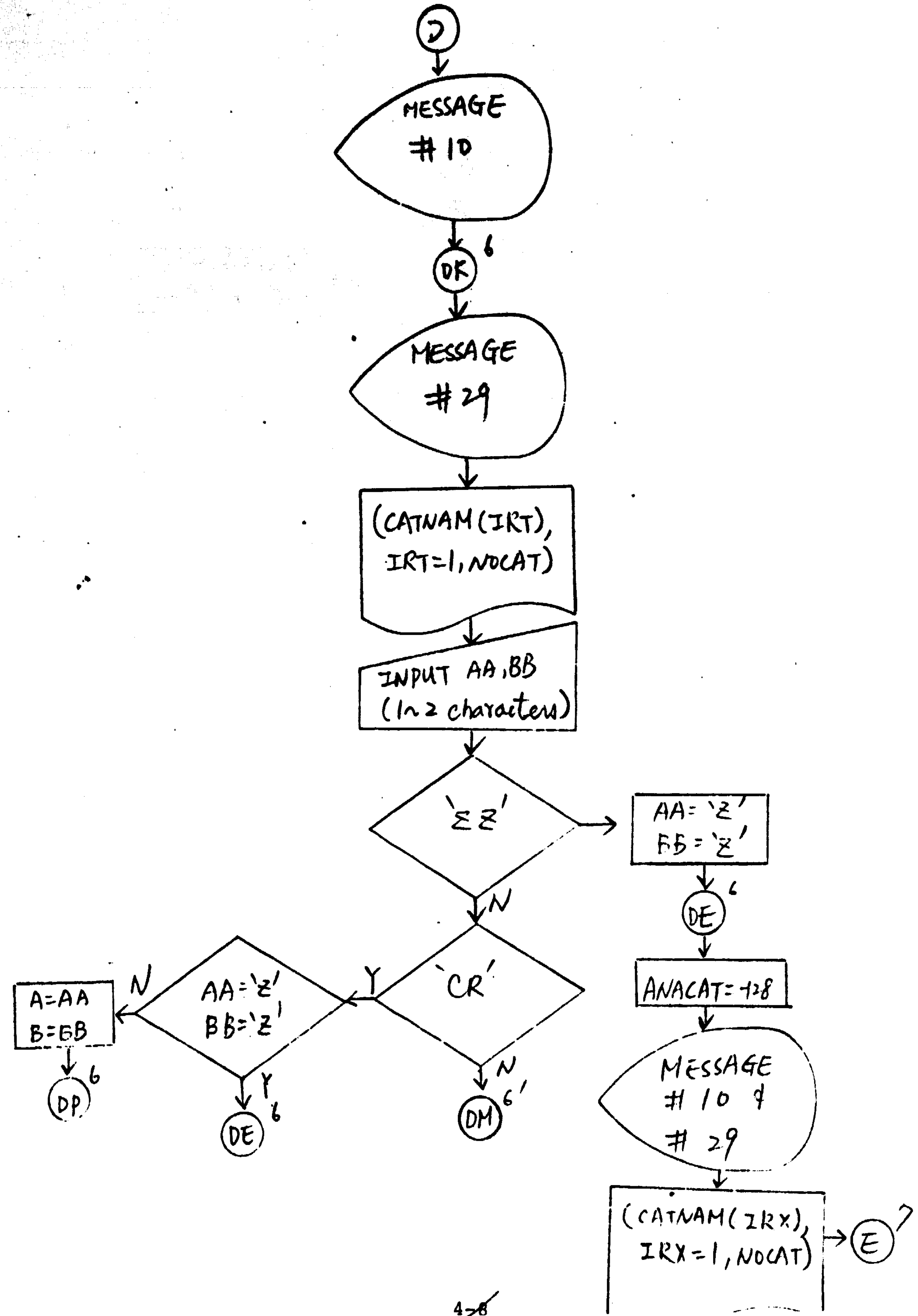

Y1 2
CALL DOTSEL
(DOTYPE, ANACAT, CLACAT, NODOT, FLG)
YP 3
KK=1
MESSAGE #15
PROCEED (Y)ES/(N)O
'Y'
Y
YE 3
RETURN
N
'N'
Y
A 1
'B'
Y
A2 2
N
'X'
Y
U 1
N

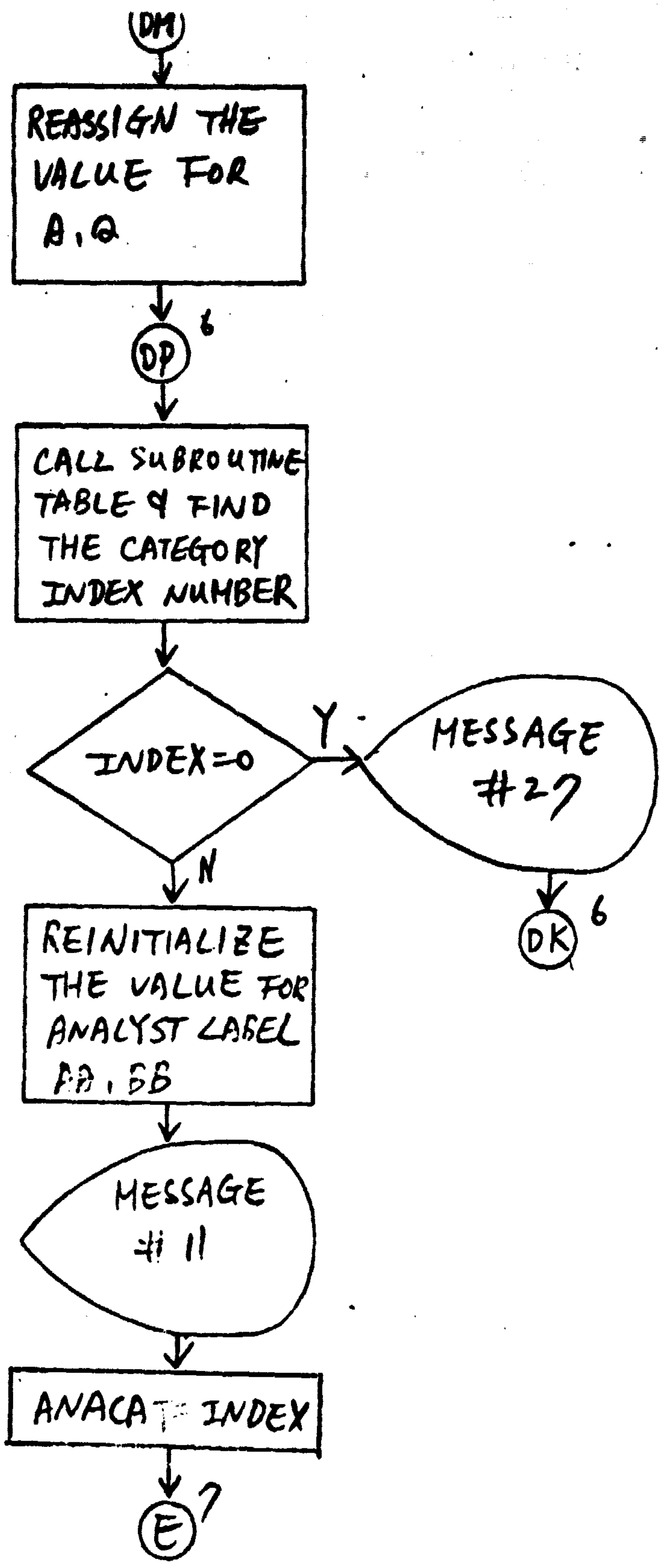

Y2 1
MESSAGE #2
INPUT (1~NDOTS)
EXIT
Y
U 1
N
BACKUP
Y
A 1
N
'CR'
N
Y1 3
N
OUT OF BOUND
Y
MESSAGE #20
Y2 4
N
REINITIALIZE VALUE FOR SELD2: NODOT=SELD2
MESSAGE #3
Y1 3

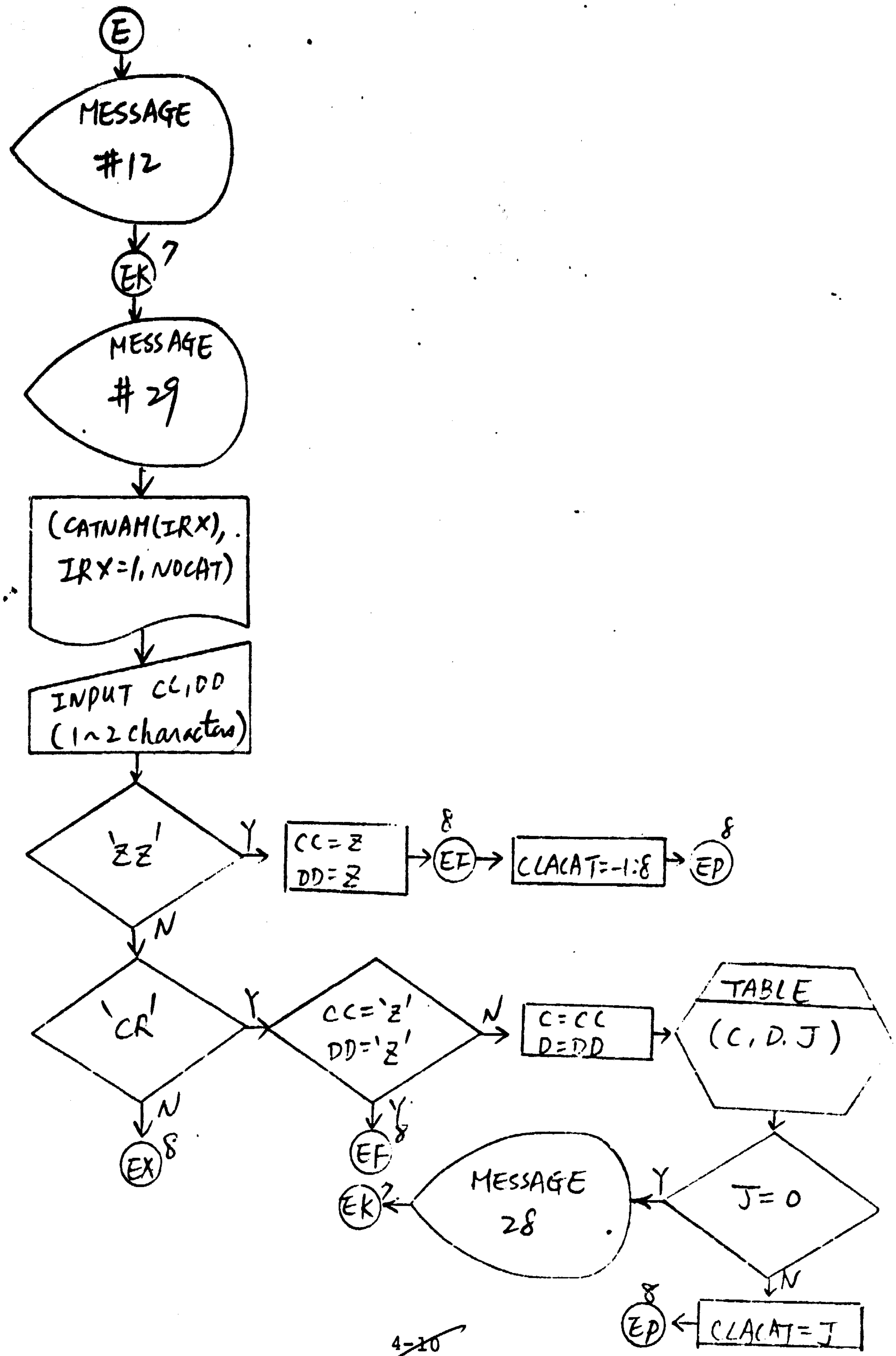


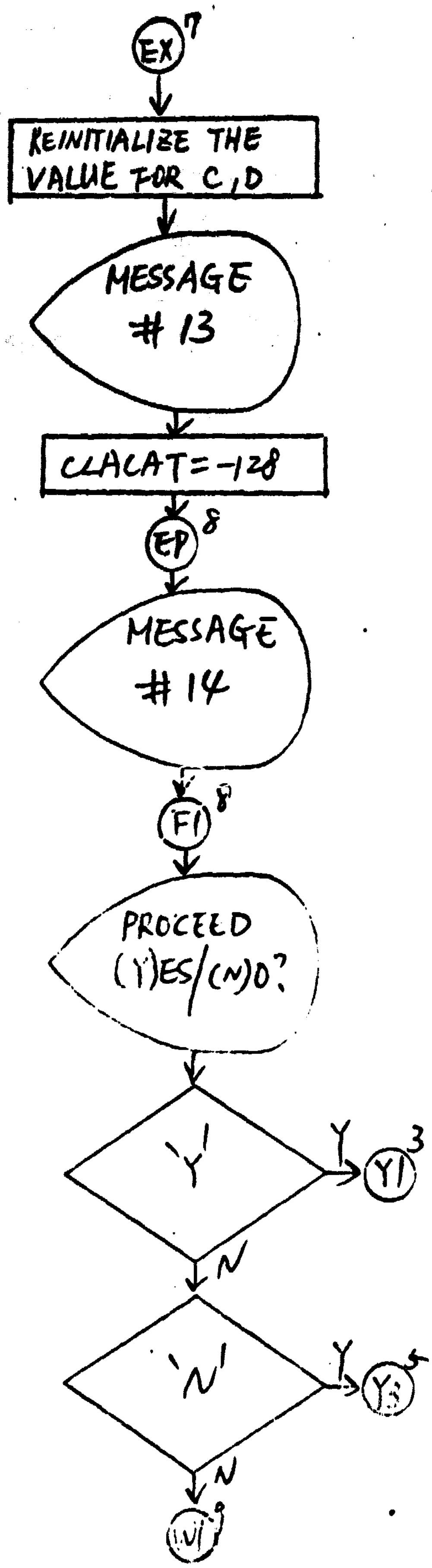
ORIGINAL PAGE IS
OF POOR QUALITY

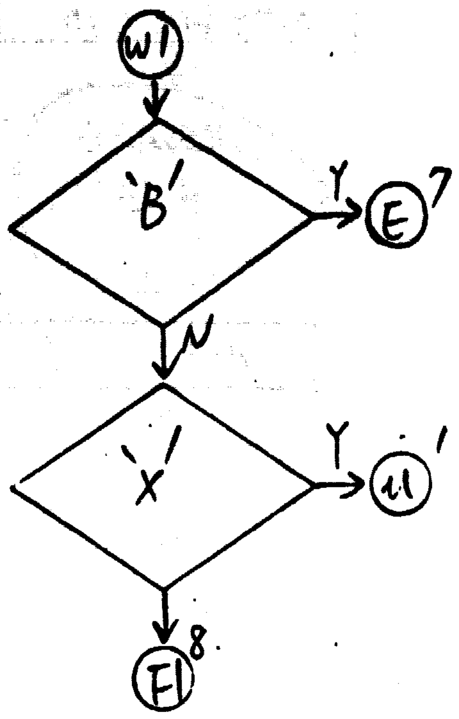

DX
'0'
Y
SELDFT(3)='0'
SELDFT(7)=0
N
'1'
Y
SELDFT(3)='1'
SELDFT(7)=1
N
'2'
N
SELDFT(3)='2'
SELDFT(7)=2
N
MESSAGE
# 20
DOTYPE=SELDFT(7)
TYPE=SELDFT(3)
Y3
5
MESSAGE
# 9
D
6

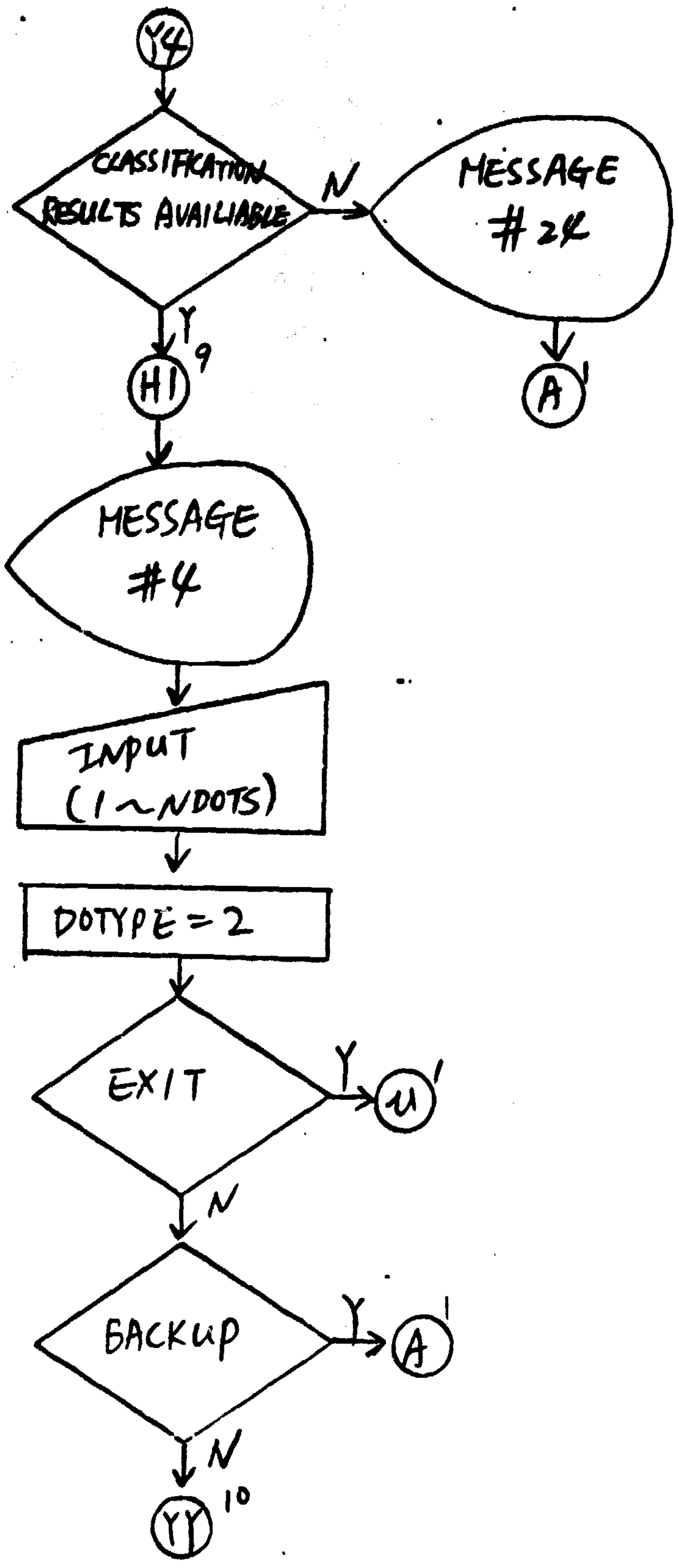

2
MESSAGE #10
DK 6
MESSAGE #29
(CATNAM(IRT), IRT=1,NOCAT)
INPUT AA,BB (1~2 characters)
'ZZ'
AA='Z' BB='Z'
DE 6
ANACAT=128
MESSAGE #10 & #29
(CATNAM(IRX), IRX=1,NOCAT)
E 7
N
'CR'
Y
AA='Z' BB='Z'
N
A=AA B=BB
DP 6
Y
DE 6
N
DM 6'

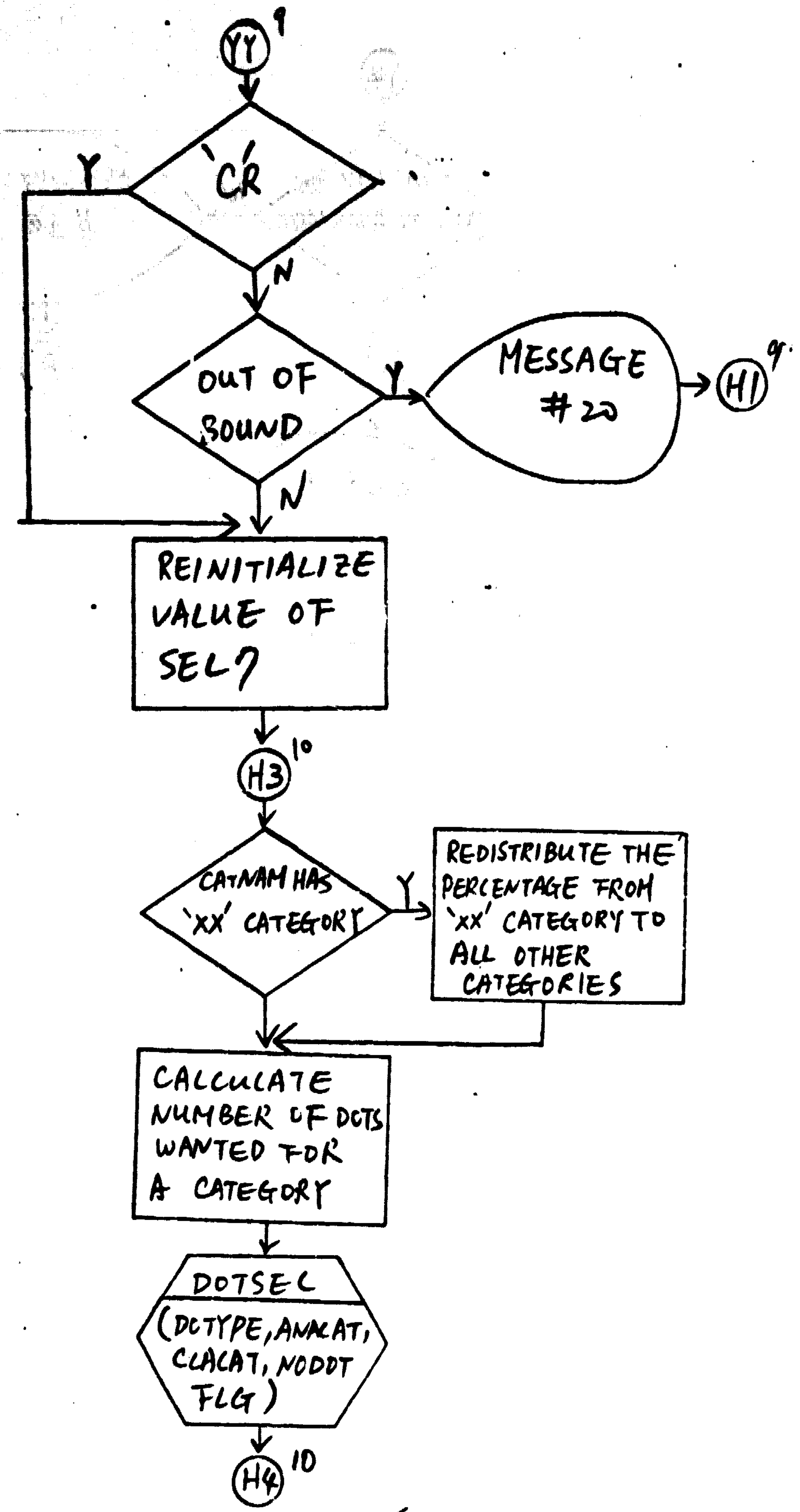

DM
REASSIGN THE VALUE FOR A, Q
DP 6
CALL SUBROUTINE TABLE 9 FIND THE CATEGORY INDEX NUMBER
INDEX=0
Y
MESSAGE #27
DK 6
N
REINITIALIZE THE VALUE FOR ANALYST LABEL AA, BB
MESSAGE #11
ANACAT= INDEX
E 7

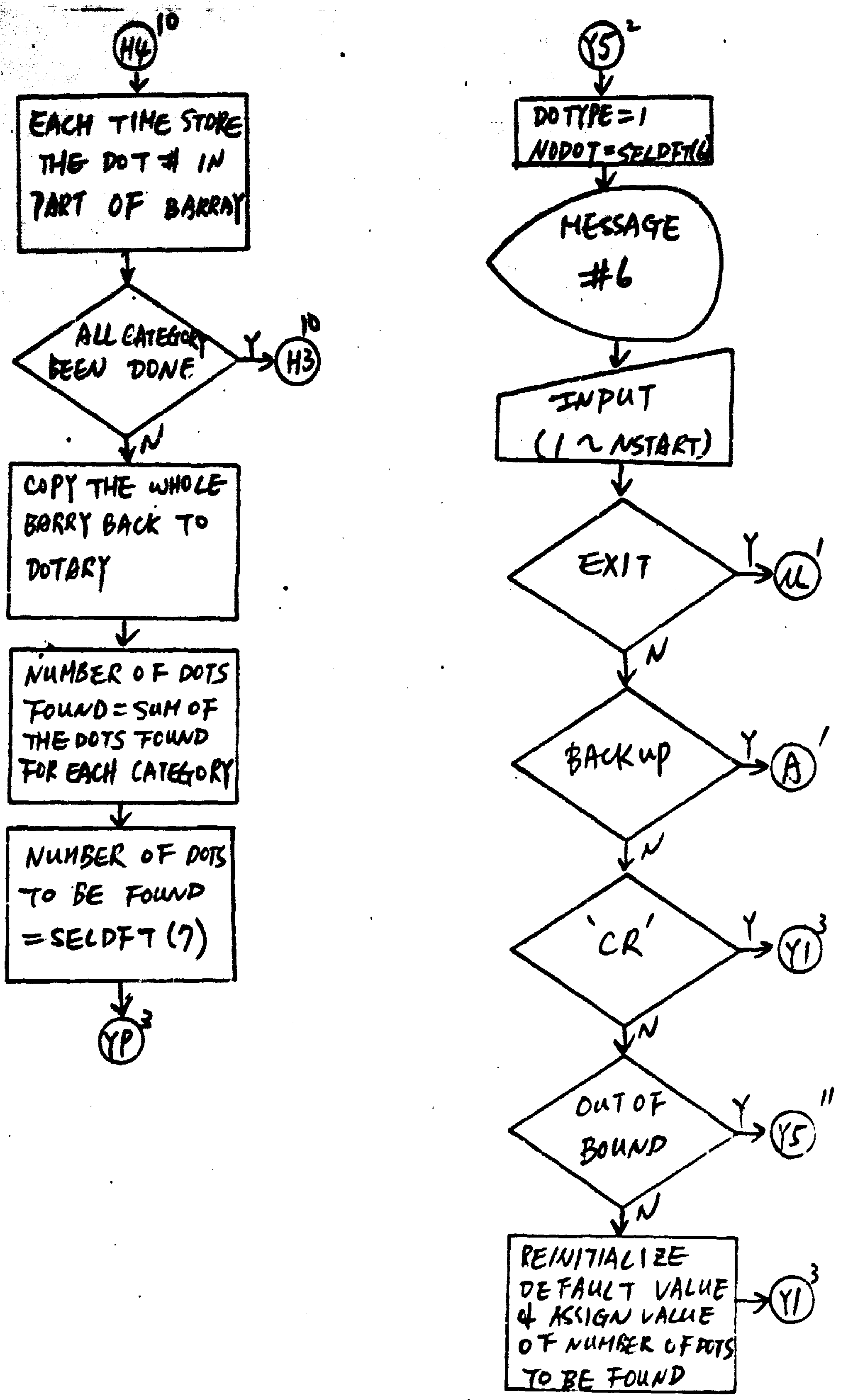

E
MESSAGE #12
EK 7
MESSAGE # 29
(CATNAM(IRX), IRX=1, NOCAT)
INPUT CC, DD (1~2 characters)
'ZZ'
Y
CC = Z DD = Z
8 EF
CLACAT=-1:8
8 EP
N
'CR'
Y
CC='Z' DD='Z'
N
C=CC D=DD
TABLE (C, D. J)
N
EX 8
Y
EF 8
J = 0
Y
MESSAGE 28
EK 7
N
CLACAT=J
8 EP


EX
7
REINITIALIZE THE VALUE FOR C,D
MESSAGE #13
CLACAT=-128
EP
8
MESSAGE #14
FI
8
PROCEED (Y)ES/(N)O?
'Y'
Y
YI
3
N
'N'
Y
Y5
5
N
WI
9

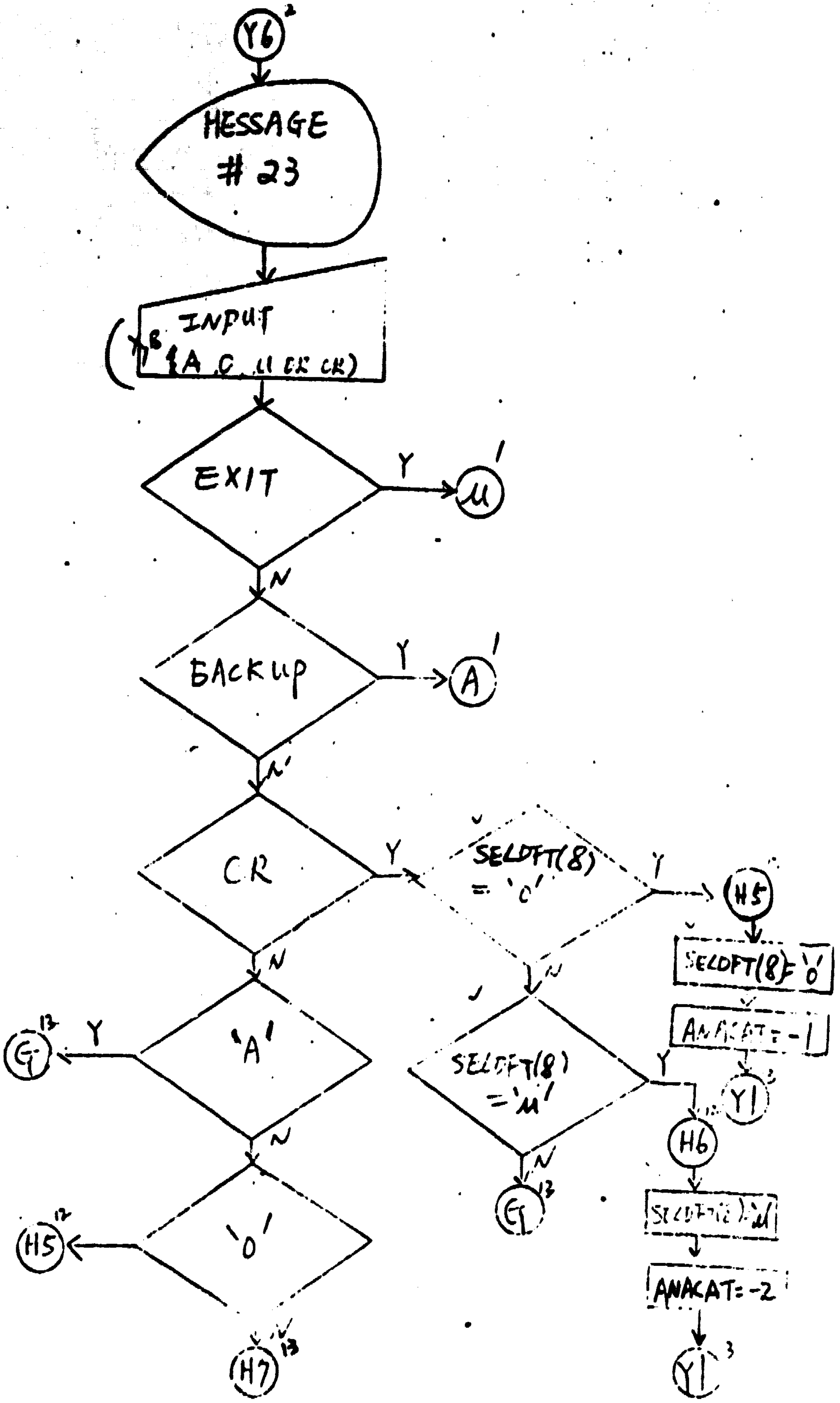

W1
'B'
Y
E
7
N
'X'
Y
11
'
F1
8


Y4
CLASSIFICATION RESULTS AVAILIABLE
N
MESSAGE #24
A 1
Y 9
H1
MESSAGE #4
INPUT (1~NDOTS)
DOTYPE = 2
EXIT
Y
U 1
N
BACKUP
Y
A 1
N
YY 10

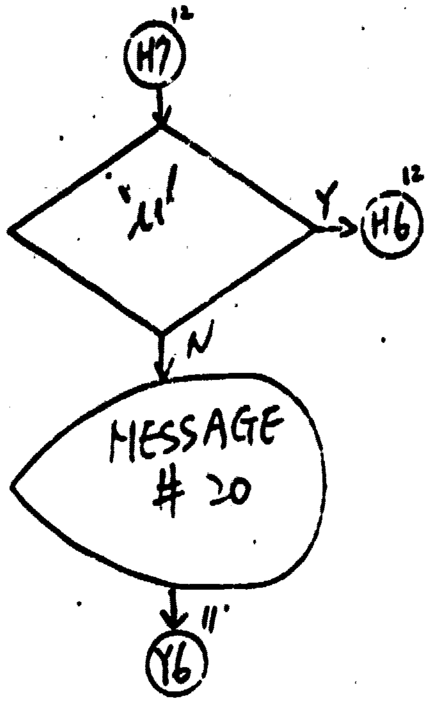

YY
9
'CR'
Y
N
OUT OF BOUND
Y
MESSAGE #20
H1
9
N
REINITIALIZE VALUE OF SEL7
H3
10
CATNAM HAS 'XX' CATEGORY
Y
REDISTRIBUTE THE PERCENTAGE FROM 'XX' CATEGORY TO ALL OTHER CATEGORIES
CALCULATE NUMBER OF DOTS WANTED FOR A CATEGORY
DOTSEL
(DCTYPE, ANACAT, CLACAT, NODOT FLG)
H4
10

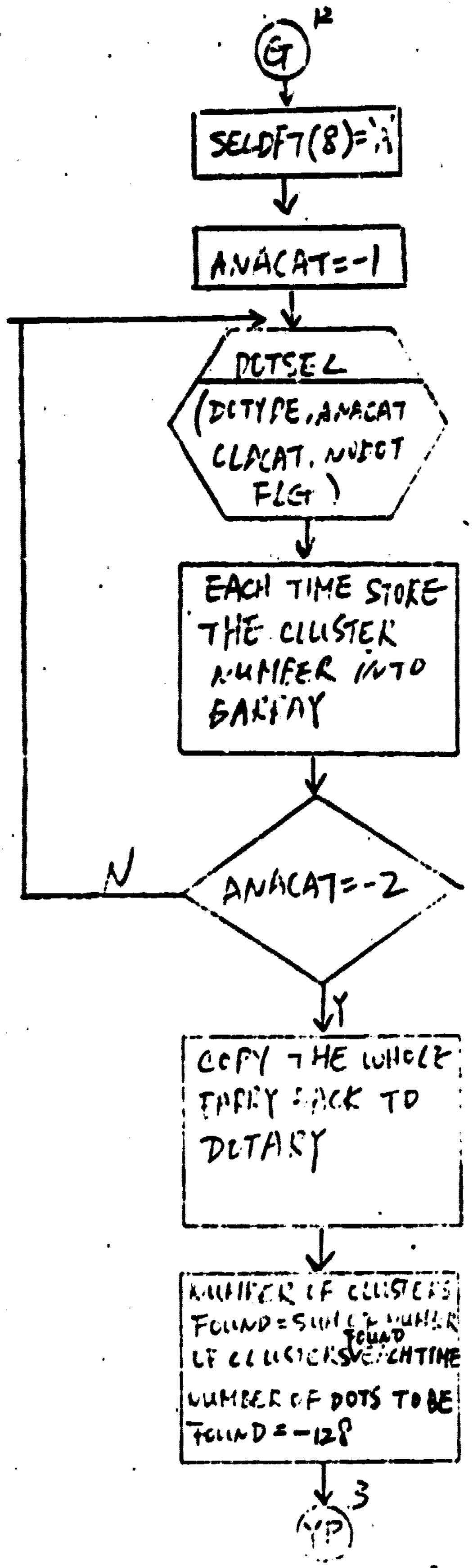


H4 10
EACH TIME STORE THE DOT # IN PART OF BARRAY
ALL CATEGORY BEEN DONE
Y
H3 10
N
COPY THE WHOLE BARRY BACK TO DOTARY
NUMBER OF DOTS FOUND = SUM OF THE DOTS FOUND FOR EACH CATEGORY
NUMBER OF DOTS TO BE FOUND =SELDFT (7)
YP 3
Y5 2
DOTYPE = 1
NODOT = SELDFT(6)
MESSAGE #6
INPUT (1 ~ NSTART)
EXIT
Y
u 1
N
BACKUP
Y
A 1
N
'CR'
Y
Y1 3
N
OUT OF BOUND
Y
Y5 11
N
REINITIALIZE DEFAULT VALUE & ASSIGN VALUE OF NUMBER OF DOTS TO BE FOUND
Y1 3



Y6
MESSAGE #23
INPUT (A, C, U OR CR)
Y8
EXIT
Y
N
U
BACKUP
A
CR
SELOFT(8) = 'C'
H5
SELOFT(8) = 'O'
ANACAT = -1
Y1
SELOFT(8) = 'U'
H6
SELOFT(8) = 'U'
ANACAT = -2
'A'
G
'O'
H7





G 12
SELDF7(8)='A'
ANACAT=-1
DCTSEL
(DCTYPE, ANACAT CLACAT, NUDOT FLG)
EACH TIME STORE THE CLUSTER NUMBER INTO BARRAY
ANACAT=-2
N
Y
COPY THE WHOLE ARRAY BACK TO DCTARY
NUMBER OF CLUSTERS FOUND = SUM OF NUMBER OF CLUSTERS FOUND EACH TIME
NUMBER OF DOTS TO BE FOUND = -128
3
YP

ORIGINAL PAGE IS
OF POOR QUALITY

```
            SUBROUTINE DOTIN(I?,KK)
      C BREAKING POINT
      C PROGRAM FOR INTERFACE
            IMPLICIT INTEGER (A-Z)
            INCLUDE 'SY:[300,3]CAMSCOMON.INC'
            INCLUDE 'SY:[300,3]CAMSPARAM.INC'
            PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
           1,MAXV=11,NDOTS=209,DLSKIP=1),DSSKIP=10,MAXACD=6,MAXACC=4,
           2NOSPWD=6,NODTWD=10
            EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
            INTEGER C1(469),C2(256),C3(71),C4(348),C5(629)
      C*
            INTEGER ACDATE,SUBCAT,SUBP?P,CATKNT,CATTH
            BYTE CHNVEC,NOCHAN,NOSUB,DOTCAT,DOTCLU
            COMMON/COM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NOCHAN,NOSUB,
           1SUBCAT(MAXSUB),SUBP2P(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NODO,
           2NODU,NOTH,DOTCAT(NDOTS),DOTCLU(NDOTS)
      C*
            INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DOTDAY,PDATE1,TDATE1
            INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDOM,GRID
            BYTE DELFLG,NOACQ,SOILGR,SUNEL,NSTART,NTYPE1,ALP,ALPO
            BYTE PCTCT,PCTCT?,VAR,VARO,DLABEL,TYPE
            COMMON/COM2/ISEG,DELFLG,NOACQ,ADATES(2,MAXACD),SOILGR(MAXACD),
           1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
           2DOTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
           3PDATE3(2),TDATE3(2),NOCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPO,
           4              PCTCT(MAXCAT),PCTCT2,VAR(MAXCAT),VARO
      C*
            INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
           1UFLAG4
            INTEGER PFLAG,DSKMNT
            COMMON/COM3/PFLAG,DSKMNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
           1,UFLAG2,UFLAG3,UFLAG4,NEWLAB(MAXSUB)
      C*
            INTEGER TX1,TY1,TX2,TY2,ACDISP,G,B,DTWIND,DOTARY,GMIN,GMAX,FUL
            INTEGER SPWIND,CLAWND,CLUWND
            COMMON/COM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
           1B(4),DTWIND(5,NODTWD),SPWIND(5,NOSPWD),IMWIND(4),NUMDOT,
           2DOTARY(NDOTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
            COMMON/COM5/DISKID,RANDOM(NDOTS),GRID(NDOTS),DLABEL(NDOTS),
           1TYPE(NDOTS),RECLOC
            REAL APC,AX,AR
            BYTE SELDFT(8),W(74),A,Q,C,D,AA,BB,CC,DD
            DIMENSION BARRY(NDOTS)
            COMMON/JCOM1/DOTYPE,ANACAT,CLACAT,NODOT,FLG
            NUMWD=1
      C INITIALIZE THE DEFAULT VALUE
            SELDFT(1)=1
            SELD2=25
            SELDFT(3)='1'
            CALL TABLE('W',' ',II)
            SELDFT(4)=II
            AA='W'
            BB=' '
            SELDFT(5)=II
            CC='W'
```

```
FØRTRAN IV-PLUS V02-04          14:15:57     31-AUG-77            PAGE 6
DTCL12.FTN      /TR:BLØCKS/WR
              DD=' '
              SELDFT(6)=NSTART
              SELDFT(7)=1
              SEL7=100
              SELDFT(8)='A'
        C PRINT ØUT ØPTIØN
        8     CØNTINUE
              CALL ØUTPUT(27,12)
              WRITE(IØ,7003)
     7003     FØRMAT(1X,T25,'***DØT SELECTIØN***',/)
              WRITE(IØ,100)
      100     FØRMAT(1X,'SELECT DØTS WITH THE FØLLØWING ØPTIØNS:')
              WRITE(IØ,799)
      799     FØRMAT(1X,'(1) ALL DØTS')
              WRITE(IØ,800)
      800     FØRMAT(1X,'(2) UNLABELED DØTS FRØM THE RANDØM SEQUENCE')
              WRITE(IØ,801)
      801     FØRMAT(1X,'(3) DØTS BY TYPE,ANALYST LABEL AND CLASSIFIER LABEL')
              WRITE(IØ,802)
      802     FØRMAT(1X,'(4) BIAS CØRRECTIØN DØTS BY THE CLASSIFICATIØN')
              WRITE(IØ,8002)
     8002     FØRMAT(5X,'PRØPØRTIØN')
              WRITE(IØ,803)
      803     FØRMAT(1X,'(5) STARTING DØTS')
              WRITE(IØ,804) SELDFT(1)
      804     FØRMAT('$(6) DØ/DU DØTS',5X,I1,1X,'>')
              CALL ØUTPUT (7)
              PØINT=1
              READ(IØ,1000) W
     1000     FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1) .EQ. 'X') GØ TØ 99
              IF(W(1) .EQ. 'R') GØ TØ 98
              IF(W(1) .EQ. ' ') GØ TØ 7
              IP=0
              CALL INTFF(IP,W,74,NUMWO)
              IF(NUMWO .LT. 1 .ØR. NUMWO .GT. 6) GØ TØ 666
              SELDFT(1)=NUMWO
        7     ANACAT=-128
              DØTYPE=-128
              CLACAT=-128
              NØDØT=-128
              CALL ØUTPUT(27,12)
              WRITE(IØ,7003)
              GØ TØ (80,13,19,155,12,121),SELDFT(1)
      666     WRITE(IØ,6666)
     6666     FØRMAT(1X,'ØUT ØF RANGE !!!',/)
              GØ TØ (8,4,150,122,11,125,121),PØINT
        C THIS IS FØR ØPTIØN 5 TØ SAY
       12     DØTYPE=1
              NØDØT=SELDFT(6)
      122     WRITE(IØ,148) SELDFT(6)
      148     FØRMAT(    '$INPUT NUMBER ØF STARTING DØTS:',2X,I3,1X,'>')
              CALL ØUTPUT (7)
              PØINT=4
              READ(IØ,1010)W
      1010 FØRMAT(74A1)
           CALL FRØNT(W,74)
           IF(W(1) .EQ. 'X') GØ TØ 99
           IF(W(1) .EQ. 'B') GØ TØ 8
           IF(W(1) .EQ. ' ') GØ TØ 88
           IP=0
           CALL INTFF(IP,W,74,NUMW4)
           IF(NUMW4 .LT.1 .ØR. NUMW4 .GT. NSTART) GØ TØ 666
           SELDFT(6)=NUMW4
           NØDØT=SELDFT(6)
           WRITE(IØ,1200) SELDFT(6)
      1200 FØRMAT(1X,'NUMBER ØF STARTING DØTS ',I4)
           GØ TØ 86
C THIS IS FØR ØPTIØN 2 TØ SAY
        13 ANACAT=0
           NØDØT=SELD2
         4 WRITE(IØ,147) SELD2
       147 FØRMAT('$INPUT NUMBER ØF UNLABELED DØTS!',2X,I3,1X,'>')
           CALL ØUTPUT (7)
           PØINT=2
           READ(IØ,1009) W
      1009 FØRMAT(74A1)
           CALL FRØNT(W,74)
           IF(W(1) .EQ. 'X') GØ TØ 99
           IF(W(1) .EQ. 'B') GØ TØ 8
           IF(W(1) .EQ. ' ') GØ TØ 88
           IP=0
           CALL INTFF(IP,W,74,NUMW5)
           IF(NUMW5 .LE. 0 .ØR. NUMW5 .GT. NDØTS) GØ TØ 666
           SELD2=NUMW5
           NØDØT=SELD2
           WRITE(IØ,1201) SELD2
      1201 FØRMAT(1X,'NUMBER ØF UNLABELED DØTS =',1X,I3)
           GØ TØ 86
C THIS IS FØR ØPTIØN 3 TØ SAY
        19 DØTYPE=SELDFT(7)
        11 WRITE(IØ,108) SELDFT(3)
       108 FØRMAT(     '$INPUT TYPE',2X,A1,1X,'>')
           CALL ØUTPUT (7)
           P1=1
           PØINT=5
           READ(IØ,1001)W
      1001 FØRMAT(74A1)
           CALL FRØNT(W,74)
           IF(W(1) .EQ. 'X') GØ TØ 99
           IF(W(1) .EQ. 'B') GØ TØ 8
           IF(W(1) .EQ. 'A') GØ TØ 23
           IF(W(1) .EQ. ' ') GØ TØ 21
           IF(W(1) .EQ. '0') GØ TØ 335
           IF(W(1) .EQ. '1') GØ TØ 336
           IF(W(1) .EQ. '2') GØ TØ 337
           GØ TØ 666
       335 SELDFT(3)='0'
           SELDFT(7)=0
           GØ TØ 338
       336 SELDFT(3)='1'
```

```
              SELDFT(7)=1
              GØ TØ 338
       337    SELDFT(3)='2'
              SELDFT(7)=2
       338    DØTYPE=SELDFT(7)
              WRITE(12,1202) SELDFT(3)
      1202    FØRMAT(1X,'TYPE = ',A1)
              GØ TØ 201
       231    SELDFT(3)='A'
              GØ TØ 202
        21    IF(SELDFT(3) .EQ. 'A') GØ TØ 202
              GØ TØ 201
       202    DØTYPE=-128
              P1=2
       201    WRITE(12,1777)
      1777    FØRMAT(1X,'"BACKUP" AND "EXIT" ØPTIØNS ARE NØT ACCEPTED HERE')
      7781    CØNTINUE
              WRITE(12,7001)
      7001    FØRMAT(1X,'AVAILABLE CATEGØRY NAMES :')
              WRITE(12,7002) (CATNAM(IRT),IRT=1,NØCAT)
      7002    FØRMAT(1X,20(A2,1X))
              WRITE(12,7780)
      7780    FØRMAT(50X)
              WRITE(12,103) AA,BB
       103    FØRMAT('$INPUT ANALYST LABEL',2X,2A1,1X,'>')
              P2=1
              CALL ØUTPUT (7)
              READ(12,1002)W
      1002    FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1) .EQ. 'Z' .AND. W(2) .EQ. 'Z') GØ TØ 232
              IF(W(1) .EQ. ' ') GØ TØ 200
              A=W(1)
              Q=W(2)
              CALL TABLE(A,Q,I)
              IF(I .EQ. 0) GØ TØ 209
              BB=Q
              AA=A
              WRITE(12,1203) AA,BB
      1203    FØRMAT(1X,'ANALYST LABEL =',1X,2A1)
              GØ TØ 203
       232    AA='Z'
              BB='Z'
              GØ TØ 238
       200    IF(AA .EQ. 'Z' .AND. BB .EQ. 'Z') GØ TØ 238
              GØ TØ 2011
       238    ANACAT=-128
              P2=2
              GØ TØ 239
      2011    A=AA
              Q=BB
              CALL TABLE(A,Q,I)
              IF(I .EQ. 0) GØ TØ 209
       203    ANACAT=I
       239    WRITE(12,1777)
      7782    CØNTINUE
```

```
                WRITE(IO,7001)
                WRITE(IO,7002) (CATNAM(IRX),IRX=1,NØCAT)
                WRITE(IO,7700)
                WRITE(IO,104) CC,DD
          104   FØRMAT('$INPUT CLASSIFIER LABEL',2X,2A1,1X,'>')
                CALL OUTPUT (7)
                P3=1
                READ(IP,1003)W
         1003   FØRMAT(74A1)
                CALL FRONT(W,74)
                IF(W(1) .EQ. 'Z' .AND. W(2) .EQ. 'Z') GØ TØ 234
                IF(W(1) .EQ. ' ') GØ TØ 233
                C=W(1)
                D=W(2)
                CALL TABLE(C,D,J)
                IF(J .EQ. 0) GØ TØ 273
                CC=C
                DD=D
                WRITE(IO,1204) CC,DD
         1204   FØRMAT(1X,'CLASSIFIER LABEL =',1X,2A1)
                GØ TØ 204
          234   CC='Z'
                DD='Z'
                GØ TØ 261
          233   IF(CC .EQ. 'Z' .AND. DD .EQ. 'Z') GØ TØ 261
                GØ TØ 237
          261   CLACAT=-128
                P3=2
                GØ TØ 9
          237   C=CC
                D=DD
                CALL TABLE(C,D,J)
                IF(J .EQ. 0) GØ TØ 273
          204   CLACAT=J
            9   CØNTINUE
                CALL OUTPUT(27,12)
                WRITE(IO,105)
          105   FØRMAT(1X,'DOT SELECTIØN BY TYPE,ANALYST LABEL AND CLASSIFIER')
                WRITE(IO,106)
          106   FØRMAT(1X,'LABEL')
                GØ TØ (400,401),P1
          400   WRITE(IO,164) SELDOT(3)
          164   FØRMAT(3X,'TYPE',12X,'!',A1)
                GØ TØ 180
          401   WRITE(IO,165)
          165   FØRMAT(3X,'TYPE',12X,'!ALL')
          180   GØ TØ (402,403),P2
          402   WRITE(IO,166)AA,BB
          166   FØRMAT(3X,'ANALYST LABEL',3X,'!',2A1)
                GØ TØ 181
          403   WRITE(IO,167)
          167   FØRMAT(3X,'ANALYST LABEL',3X,'!ALL')
          181   GØ TØ (404,405),P3
          404   WRITE(IO,168) CC,DD
          168   FØRMAT(3X,'CLASSIFIER LABEL!',2A1)
                GØ TØ 25
```

```
  209 WRITE(I2,1019) A,Q
 1019 FØRMAT(1X,'ANALYST LABEL ',2A1,' NØT FØUND !!!',/)
      GØ TØ 7701
  273 WRITE(I2,1018) C,D
 1018 FØRMAT(1X,'CLASSIFIER LABEL ',2A1,' NØT FØUND !!!',/)
      GØ TØ 7702
  405 WRITE(I2,169)
  169 FØRMAT(3X,'CLASSIFIER LABEL:ALL')
  25  WRITE(I2,107)
  107 FØRMAT(    '$PRØCEED (Y)ES/(N)Ø? >')
      CALL ØUTPUT (7)
      READ(I2,1005)W
 1005 FØRMAT(74A1)
      CALL FRØNT(W,74)
      IF(W(1) .EQ. 'Y') GØ TØ 38
      IF(W(1) .EQ. 'N') GØ TØ 11
      IF(W(1) .EQ. 'Ø') GØ TØ 239
      IF(W(1) .EQ. 'X') GØ TØ 99
      GØ TØ 25
C THIS IS FØR ØPTIØN 4 TØ SAY
C CHECK FØR CLASSIFICATIØN RESULT
  125 DØ 215 JI=1,NØSUB
      IF(PCTCT(JI) .GT. 0) GØ TØ 150
  215 CØNTINUE
      WRITE(I2,471)
  471 FØRMAT(1X,'NØ CLASSIFICATIØN RESULT !!!',/)
      GØ TØ 8
  150 WRITE(I2,4001) SEL7
 4001 FØRMAT('$INPUT NUMBER ØF BIAS CØRRECTIØN DØTS:',2X,I3,1X,'>')
      CALL ØUTPUT (7)
      DØTYPE=2
      PØINT=3
      READ(I2,488) W
  488 FØRMAT(74A1)
      CALL FRØNT(W,74)
      IF(W(1) .EQ. 'X') GØ TØ 99
      IF(W(1) .EQ. 'Ø') GØ TØ 8
      IF(W(1) .EQ. ' ')GØ TØ 96
      IP=0
      CALL INTFF(IP,W,74,NUMW7)
      IF(NUMW7 .LE.0 .ØR. NUMW7 .GT.NDØTS) GØ TØ 666
      SEL7=NUMW7
 1400 FØRMAT(1X,'NUMBER ØF BIAS CØRRECTIØN DØTS =',1X,I3)
      NØDØT=NUMW7
      WRITE(I2,1400) NUMW7
  96  INDXP=1
      CNUM=0
      DØ 94 IS=1,NØCAT
      APC=PCTCT(IS)
      CALL TABLE('X','X',K)
      IF(K .EQ. 0) GØ TØ 67
      IF(IS .EQ. K) GØ TØ 94
      AX=PCTCT(K)
      GØ TØ 677
  67  AX=0
  677 AR=(APC+AX*APC/(100.-AX))*SEL7/100.
```

```
      VALUE=AR*100
      IT1=VALUE/10*10
      DIFF1=VALUE-IT1
      IF(DIFF1 .LT. 5) GØ TØ 33
      IT1=IT1+10
   33 IT2=IT1/100*100
      DIFF2=(IT1-IT2)/10
      IF(DIFF2 .LT. 5) GØ TØ 41
      IT2=IT2+100
   41 NØDØT=IT2/100
      CLACAT=15
      CALL DØTSEL(DØTYPE,ANACAT,CLACAT,NØDØT,FLG)
      CNUM=CNUM+NUMDØT
      ISP2=0
      DØ 57 IRY=INDXP,CNUM
      ISP2=ISP2+1
      BARRY(IRY)=DØTARY(ISP2)
   57 CØNTINUE
      INDXP=INDXP+NUMDØT
   94 CØNTINUE
      DØ 56 ISP=1,CNUM
      DØTARY(ISP)=BARRY(ISP)
   56 CØNTINUE
      NUMDØT=CNUM
      NØDØT=SEL7
      GØ TØ 178
C THIS IS FØR ØPTIØN 6 TØ SAY
  121 WRITE(IØ,1299) SELDFT(8)
 1299 FØRMAT(' SELECT (A)LL DØ/DU DØTS, D(Ø) DØTS ØR D(U) DØTS'/
     1'$',2X,A1,1X,'>')
      CALL ØUTPUT (7)
      PØINT=7
      READ(IØ,129) W
  129 FØRMAT(74A1)
      CALL FRØNT(W,74)
      IF(W(1) .EQ. 'X') GØ TØ 99
      IF(W(1) .EQ. 'H') GØ TØ 6
      IF(W(1) .EQ. ' ') GØ TØ 125
      IF(W(1) .EQ. 'A') GØ TØ 141
      IF(W(1) .EQ. 'Ø') GØ TØ 126
      IF(W(1) .EQ. 'U') GØ TØ 127
      GØ TØ 566
  125 IF(SELDFT(8) .EQ. 'Ø') GØ TØ 126
      IF(SELDFT(8) .EQ. 'U') GØ TØ 127
  141 DNUM=0
      JNXP=1
      SELDFT(8)='A'
      ANACAT=-1
  569 CALL DØTSEL(DØTYPE,ANACAT,CLACAT,NØDØT,FLG)
      DNUM=DNUM+NUMDØT
      JSP=0
      DØ 567 JRY=JNXP,DNUM
      JSP=JSP+1
      BARRY(JRY)=DØTARY(JSP)
  567 CØNTINUE
      IF(ANACAT .EQ. -2) GØ TØ 570
```

```
                ANACAT=-2
                JNXP=JNXP+NUMDØT
                GØ TØ 569
          570   DØ 571 KK1=1,DNUM
                DØTARY(KK1)=PARRY(KK1)
          571   CØNTINUE
                NUMDØT=DNUM
                NØDØT=-128
                GØ TØ 178
          126   SELDFT(8)='Ø'
                ANACAT=-1
                GØ TØ 88
          127   SELDFT(8)='U'
                ANACAT=-2
          88    CØNTINUE
                CALL DØTSEL(DØTYPE,ANACAT,CLACAT,NØDØT,FLG)
          178   KK=1
                WRITE(IØ,7780)
                WRITE(IØ,1111)
         1111   FØRMAT(1X,'DØT SELECTIØN REPØRT:')
                WRITE(IØ,1502)
         1502   FØRMAT(1X,'DØT GRID NUMBER:')
                WRITE(IØ,1095) (DØTARY(JJ),JJ=1,NUMDØT)
         1095   FØRMAT(1X,10I5)
                IF(NØDØT .EQ. -128) GØ TØ 723
                WRITE(IØ,1112) NØDØT
         1112   FØRMAT(1X,'NUMBER ØF DØTS TØ BE SELECTED=',I4)
                GØ TØ 724
          723   WRITE(IØ,1119)
         1119   FØRMAT(1X,'NUMBER ØF DØTS TØ BE SELECTED= ALL')
          724   WRITE(IØ,1113) NUMDØT
         1113   FØRMAT(1X,'NUMBER ØF DØTS SELECTED      =',I4,/)
          61    CØNTINUE
                WRITE(IØ,7780)
                WRITE(IØ,1004)
         1004   FØRMAT(    '$PRØCEED (Y)ES/(N)Ø? >')
                CALL ØUTPUT (7)
                READ(IØ,190) W
          190   FØRMAT(74A1)
                CALL FRØNT(W,74)
                IF(W(1) .EQ. 'Y') GØ TØ 999
                IF(W(1) .EQ. 'N') GØ TØ 8
                IF(W(1) .EQ. 'R') GØ TØ 7
                IF(W(1) .EQ. 'X') GØ TØ 99
                GØ TØ 61
          99    KK=2
                GØ TØ 999
          98    KK=3
          999   RETURN
                END
```

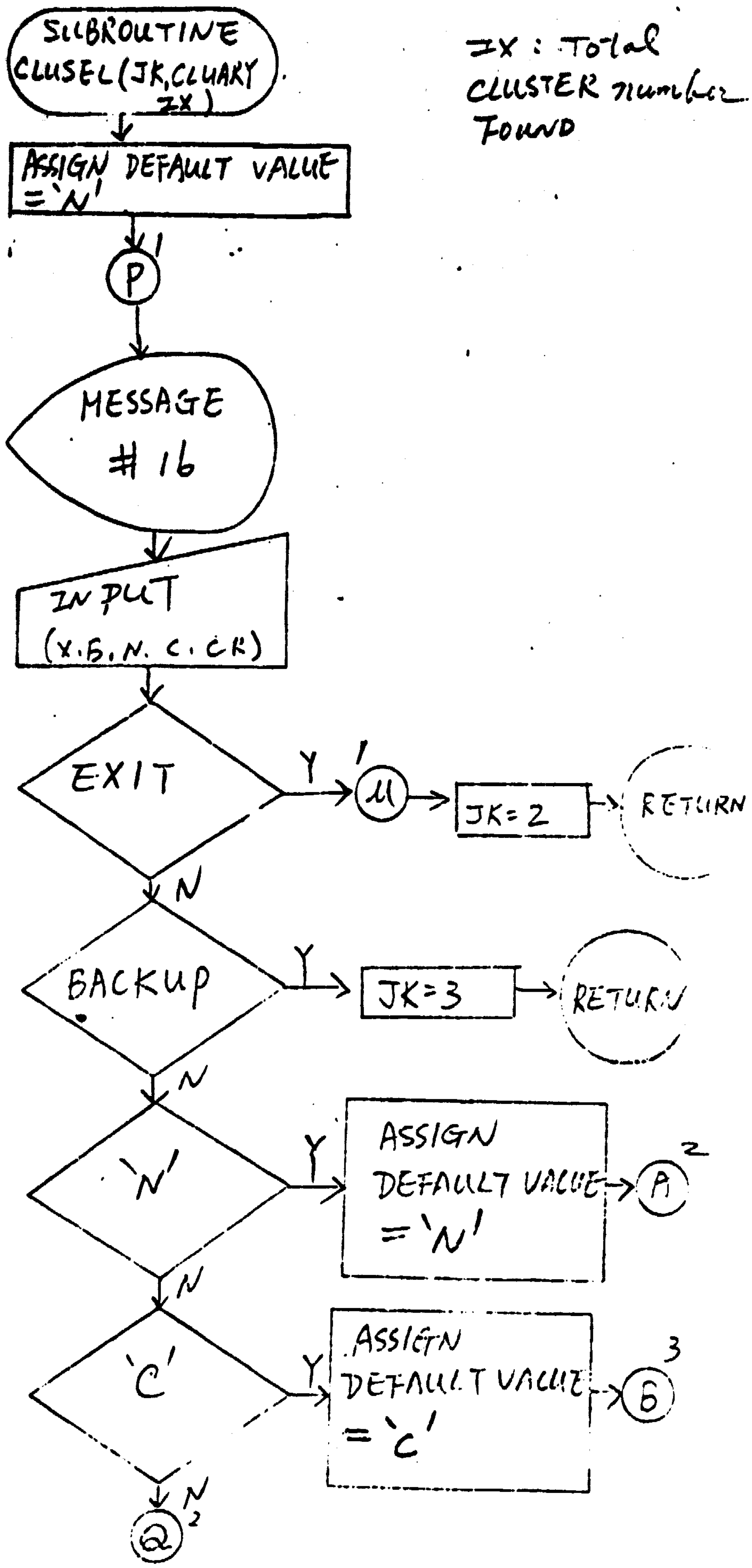
## 4.2 ENTRY POINT - CLUSEL

The subroutine DTCLIO uses the subroutine CLUSEL for cluster selection. It will allow the analyst to select the clusters according to category names or cluster numbers.

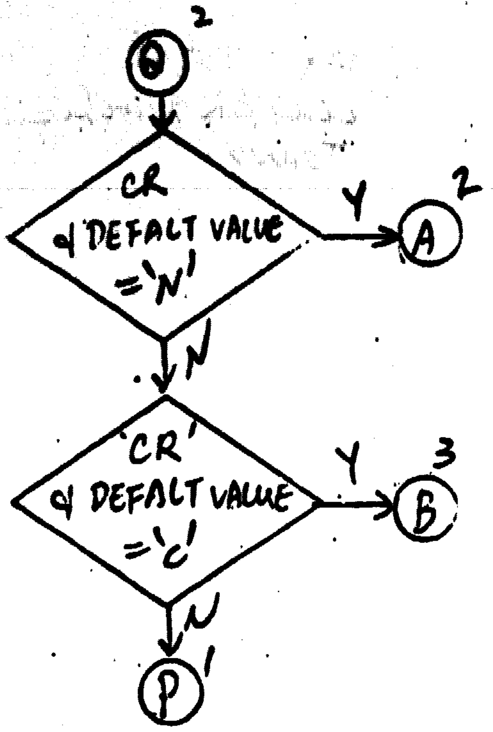
- Calling sequence

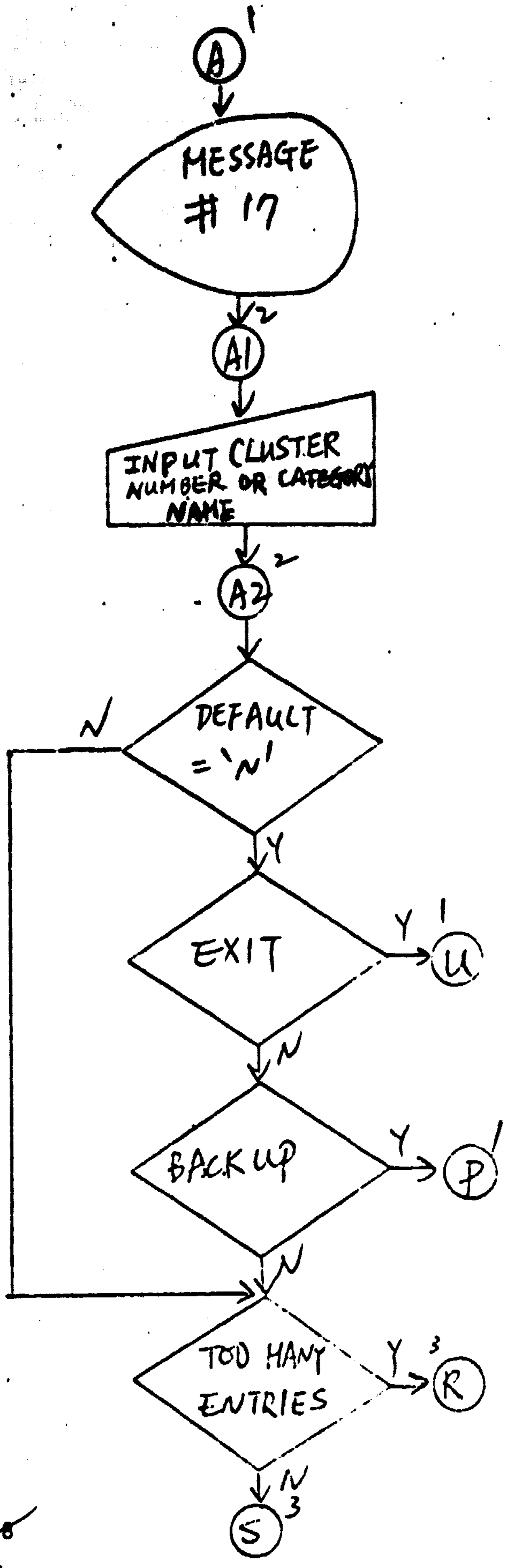
CALL CLUSEL (IO,JK,CLUARY,IX)

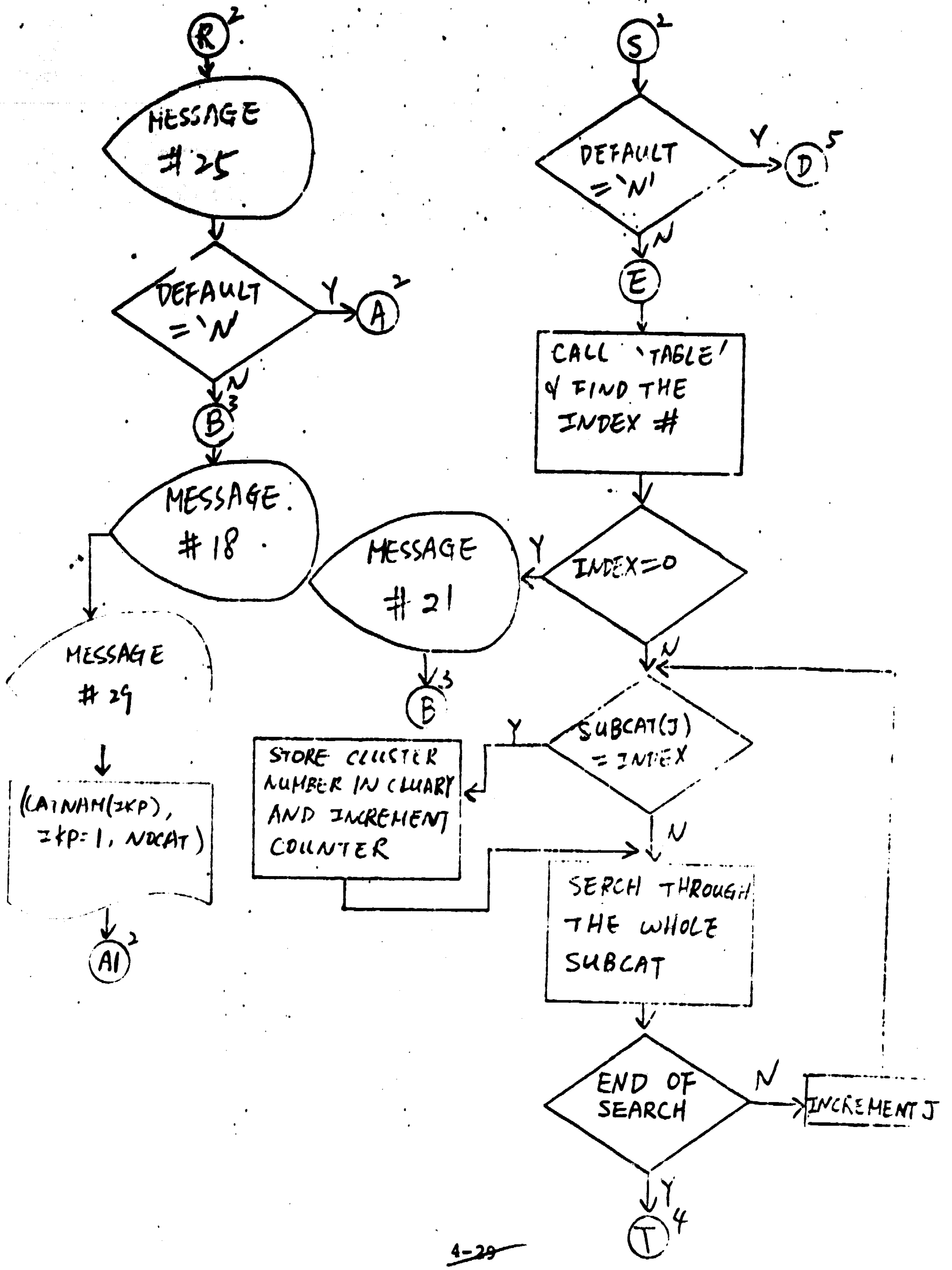
| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| IO | Integer | - | In | Input/output unit |
| dK | Integer | - | Out | See KK in DOTIN |
| CLUARY | Integer | MAXSUB | Out | Cluster array |
| IX | Integer | - | Out | Number of clusters |

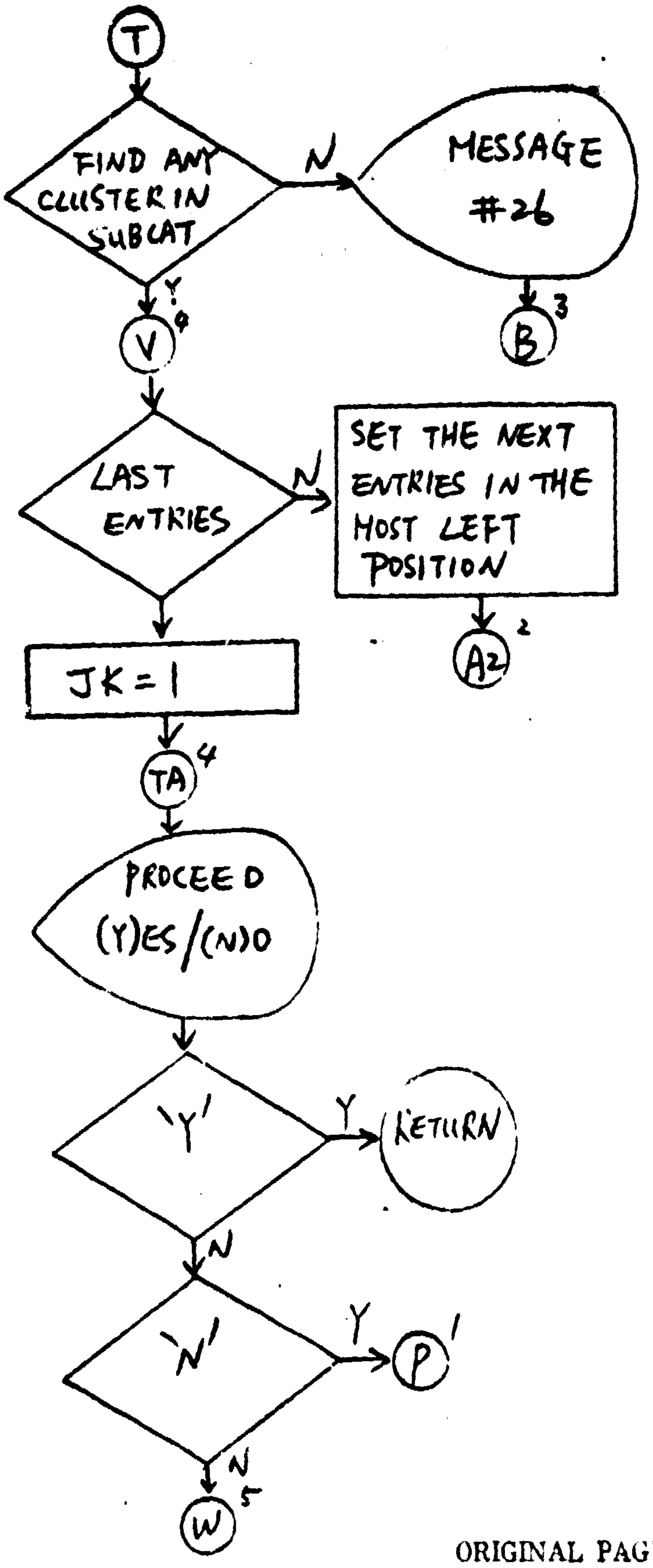

SUBROUTINE CLUSEL(JK,CLUARY, ZX)
ASSIGN DEFAULT VALUE ='N'
P
1
MESSAGE #16
INPUT (X.B.N.C.CR)
EXIT
Y
N
1
M
JK=2
RETURN
BACKUP
JK=3
RETURN
'N'
ASSIGN DEFAULT VALUE ='N'
A
2
'C'
ASSIGN DEFAULT VALUE ='C'
B
3
Q
2

ZX : Total CLUSTER number FOUND

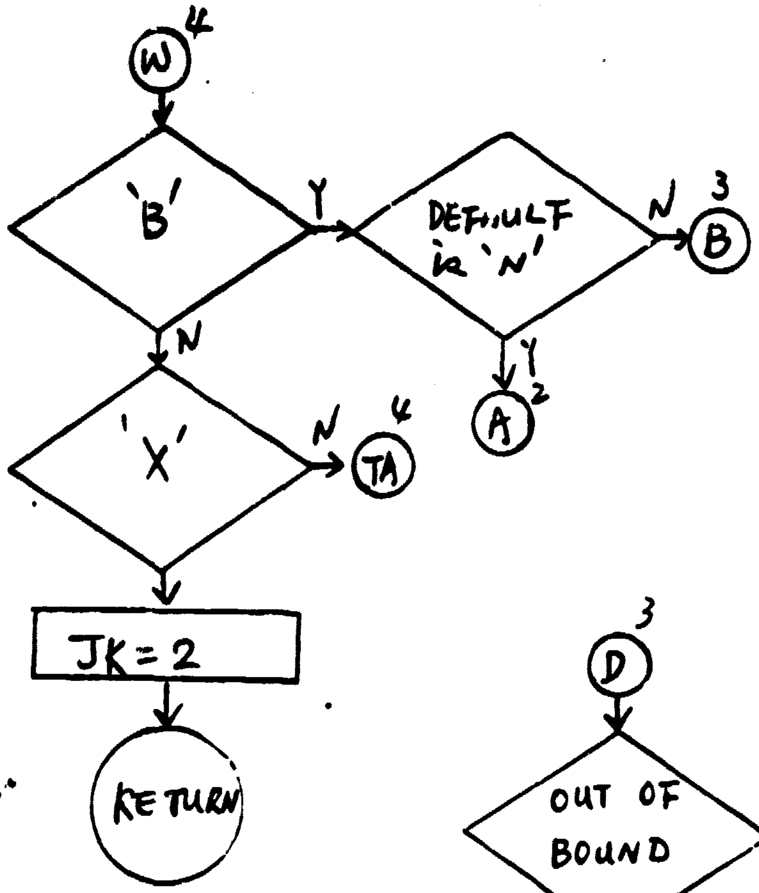

Q 2
CR
& DEFALT VALUE
='N'
Y
A 2
N
'CR'
& DEFALT VALUE
='C'
Y
B 3
N
P 1

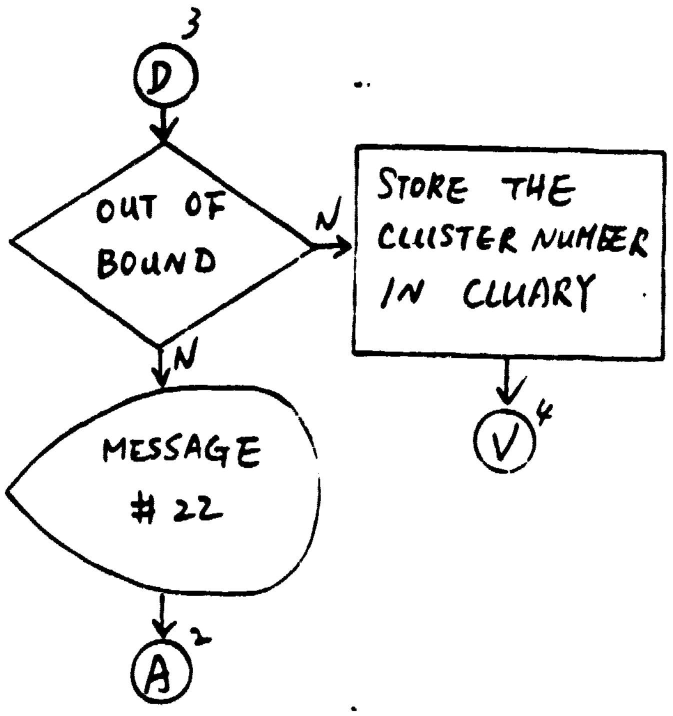

A 1
MESSAGE
# 17
A1 2
INPUT CLUSTER NUMBER OR CATEGORY NAME
A2 2
DEFAULT
='N'
N
Y
EXIT
Y
U 1
N
BACKUP
Y
P 1
N
TOO MANY ENTRIES
Y
R 3
N
S 3


R 2
MESSAGE #25
DEFAULT ='N
Y
A 2
N
B 3
MESSAGE #18
MESSAGE #29
(CATNAM(IKP), IKP=1, NOCAT)
A1 2
MESSAGE #21
B 3
S 2
DEFAULT ='N'
Y
D 5
N
E
CALL 'TABLE' & FIND THE INDEX #
INDEX=0
Y
N
SUBCAT(J) = INDEX
Y
STORE CLUSTER NUMBER IN CLUARY AND INCREMENT COUNTER
N
SERCH THROUGH THE WHOLE SUBCAT
END OF SEARCH
N
INCREMENT J
Y
T 4



ORIGINAL PAGE I
OF POOR QUALITY


W 4
'B'
Y
DEFAULT
is 'N'
N
B 3
Y
A 2
N
'X'
N
TA 4
JK = 2
RETURN


D 3
OUT OF
BOUND
N
STORE THE
CLUSTER NUMBER
IN CLUARY
V 4
N
MESSAGE
# 22
A 2

```
              SUBRØUTINE CLUSEL(IW,JK,CLUARY,IX)
              IMPLICIT INTEGER (A-Z)
              INCLUDE 'SY:[300,3]CAMSCØMMØN.INC'
     •        INCLUDE 'SY:[300,3]CAMSPARAM.INC'
     •        PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
     •       1,MAXV=11,NDØTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
     •       2NØSPWD=6,NØTWHD=10
     •        EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
     •        INTEGER C1(469),C2(256),C3(71),C4(348),C5(629)
     •  C•
     •        INTEGER ACDATE,SUBCAT,SUBPØP,CATKNT,CATTH
     •        BYTE CHNVEC,NØCHAN,NØSUB,DØTCAT,DØTCLU
     •        CØMMØN/CØM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NØCHAN,NØSUB,
     •       1SUBCAT(MAXSUB),SUBPØP(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NØDØ,
     •       2NØDU,NØTH,DØTCAT(NDØTS),DØTCLU(NDØTS)
     •  C•
     •        INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DØTDAY,PDATE1,TDATE1
     •        INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDØM,GRID
     •        BYTE DELFLG,NØACD,SØILGR,SUNEL,NSTART,NTYPE1,ALP,ALPØ
     •        BYTE PCTCT,PCTCTØ,VAR,VARØ,DLABEL,TYPE
     •        CØMMØN/CØM2/ISEG,DELFLG,NØACD,ADATES(2,MAXACD),SØILGR(MAXACD),
     •       1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
     •       2DØTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
     •       3PDATE3(2),TDATE3(2),NØCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPØ,
     •       4              PCTCT(MAXCAT),PCTCTØ,VAR(MAXCAT),VARØ
     •  C•
     •        INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
     •       1UFLAG4
     •        INTEGER PFLAG,DSKMNT
     •        CØMMØN/CØM3/PFLAG,DSKMNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
     •       1,UFLAG2,UFLAG3,UFLAG4,NENLAB(MAXSUB)
     •  C•
     •        INTEGER IX1,IY1,IX2,IY2,ACDISP,G,B,DTWIND,DØTARY,GMIN,GMAX,FUL
     •        INTEGER SPWIND,CLAWND,CLUWND
     •        CØMMØN/CØM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
     •       1B(4),DTWIND(5,NØTWHD),SPWIND(5,NØSPWD),IMWIND(4),NUMDØT,
     •       2DØTARY(NDØTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
     •        CØMMØN/CØM5/DISKID,RANDØM(NDØTS),GRID(NDØTS),DLABEL(NDØTS),
     •       1TYPE(NDØTS),RECLØC
              BYTE W(74),A(3)
              DIMENSIØN CLUARY(MAXSUB)
              ENTRY=30
              ENTRC=20
              DFT='N'
              DEFALT=1
         60   CØNTINUE
              CALL ØUTPUT (27,12)
       6051   CØNTINUE
              WRITE(IW,1600) DFT
       1600   FØRMAT(    '$SELECT CLUSTERS BY (N)UMBER ØR (C)ATEGØRY NAME',
             12X,A1,1X,'>')
              CALL ØUTPUT (7)
              READ(JK,162) W
        162   FØRMAT(74A1)
              CALL FRØNT (W,74)
              IF(W(1) .EQ. 'X') GØ TØ 72
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
              IF(W(1) .EQ. 'R') GØ TØ 720
              IF(W(1) .EQ. 'N') GØ TØ 601
              IF(W(1) .EQ. 'C') GØ TØ 602
              IF(W(1) .EQ. ' ') GØ TØ 603
              GØ TØ 6061
        601   DFT='N'
              DEFALT=1
              GØ TØ 603
        602   DFT='C'
              DEFALT=2
              GØ TØ 603
        603   CØNTINUE
              CALL ØUTPUT (27,12)
              WRITE(IØ,2103)
       2103   FØRMAT(1X,T25,'***CLUSTER SELECTIØN***',/)
              GØ TØ (61,62),DEFALT
        C THIS IS FØR SELECTING CLUSTERS BY NUMBER
         61   WRITE(IØ,1610)
       1610   FØRMAT(     '$INPUT CLUSTER NUMBERS >')
         81   IX=0
              CØUNT=0
              CALL ØUTPUT (7)
              READ(IØ,161) W
        161   FØRMAT(74A1)
              DØ 104 IZ1=1,NØSUB
              CLUARY(IZ1)=0
        104   CØNTINUE
              PX=1
         28   CALL FRØNT(W,74)
              GØ TØ (35,36),DEFALT
         35   IF(W(1) .EQ. 'X') GØ TØ 72
              IF(W(1) .EQ. 'R') GØ TØ 60
         36   IF(W(1) .EQ. ' ' .AND. PX .EQ. 1 .AND. DEFALT .EQ. 1)GØ TØ 61
              IF(W(1) .EQ. ' ' .AND. PX .EQ. 1 .AND. DEFALT .EQ. 2)GØ TØ 62
              IF(W(1) .EQ. ' ' .AND. PX .EQ. 2) GØ TØ 73
              IH1=1
         26   IF(W(IH1) .EQ. ' ' .ØR. W(IH1) .EQ. ',') GØ TØ 20
              A(IH1)=W(IH1)
              IH1=IH1+1
              IF(IH1 .GE. 4) GØ TØ 595
              GØ TØ 26
         20   GØ TØ (24,25,48),IH1
         24   W(IH1)=' '
              GØ TØ 28
         25   A(2)=' '
              GØ TØ 48
         48   CØUNT=CØUNT+1
        C CHECK WHETHER THERE ARE TØØ MANY ENTRIES
              GØ TØ (667,668),DEFALT
        667   IF(CØUNT .GE. ENTRN) GØ TØ 76
              GØ TØ 670
        668   IF(CØUNT .GE. ENTRC) GØ TØ 76
              GØ TØ 671
        672   IPP=0
              CALL INTFF(IPP,A,2,BNUM)
              IF(BNUM .LT. 1 .ØR. BNUM .GT. NØSUB) GØ TØ 598
```

```
             PX=2
             IX=IX+1
             CLUARY(IX)=BNUM
         299 DØ 29 IH2=1,IH1
             W(IH2)=' '
          29 CØNTINUE
             GØ TØ 28
         596 WRITE(IØ,1624) (A(JH3),JH3=1,2)
        1624 FØRMAT(1X,'CLUSTER NUMBER',1X,2A1,' ØUT ØF RANGE !!!',/)
             GØ TØ 61
         599 WRITE(IØ,1623) (A(JH2),JH2=1,3)
        1623 FØRMAT(1X,'CLUSTER NUMBER',1X,3A1,' ØUT ØF RANGE!!!',/)
             GØ TØ 61
         595 GØ TØ (599,596),DEFALT
      C THIS IS FØR SELECTING CLUSTERS BY CATEGØRY NAME
         62  WRITE(IØ,1660)
        1660 FØRMAT(1X,'"BACKUP" AND "EXIT" ØPTIØNS ARE NØT ACCEPTED HERE')
        6662 CONTINUE
             WRITE(IØ,2101)
        2101 FØRMAT(1X,'AVAILABLE CATEGØRY NAMES :')
             WRITE(IØ,2102)(CATNAM(IKP),IKP=1,NØCAT)
        2102 FØRMAT(1X,20(A2,1X))
             WRITE(IØ,2115)
        2115 FØRMAT(50X)
             WRITE(IØ,1612)
        1612 FØRMAT(    '$INPUT CATEGØRY NAME >')
             GØ TØ 81
         671 CALL TABLE(A(1),A(2),IK)
             IF(IK .NE. 0) GØ TØ 699
             WRITE(IØ,1627)A(1),A(2)
        1627 FØRMAT(1X,'CATEGØRY NAME',1X,2A1,1X,'NØT FØUND !!!',/)
             GØ TØ 6662
         59  WRITE(IØ,1628) (A(III),III=1,3)
        1628 FØRMAT(1X,'CATEGØRY NAME',1X,3A1,1X,'NØT FØUND !!!',/)
             GØ TØ 6662
         699 FIND=0
             DØ 69 J=1,NØSUB
             IF(SUBCAT(J) .NE. IK) GØ TØ 69
             PX=2
             IX=IX+1
             FIND=FIND+1
             CLUARY(IX)=J
         69  CØNTINUE
             IF(FIND .EQ. 0) GØ TØ 39
             GØ TØ 299
         39  WRITE(IØ,1630) A(1),A(2)
        1630 FØRMAT(1X,'NØ CLUSTER FØR CATEGØRY NAME',1X,2A1,1X,'!!!',/)
             PX=2
             GØ TØ 6662
         73  JK=1
             WRITE(IØ,1620)
        1620 FØRMAT(1X,'CLUSTER SELECTIØN REPØRT',/)
             WRITE(IØ,1621)
        1621 FØRMAT(1X,'CLUSTER NUMBERS:')
             WRITE(IØ,1622)(CLUARY(IC),IC=1,IX)
        1622 FØRMAT(1X,10I5)
```

```
FØRTRAN IV-PLUS V02-04            14:16:50    31-AUG-77              PAGE 19
DTCLI0.FTN       /TR:BLØCKS/WR

              WRITE(I0,1640) IX
        1640  FØRMAT(1X,'NUMBER ØF CLUSTERS SELECTED ='I3)
         71   CØNTINUE
              WRITE(I0,2115)
              WRITE(I0,1611)
        1611  FØRMAT(     '$PRØCEED (Y)ES/(N)Ø? >')
              CALL ØUTPUT (7)
              READ(I0,166)W
         166  FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1) .EQ. 'Y') GØ TØ 722
              IF(W(1) .EQ. 'N') GØ TØ 60
              IF(W(1) .EQ. 'P') GØ TØ 603
              IF(W(1) .EQ. 'X') GØ TØ 72
              GØ TØ 71
         76   WRITE(I0,1616)
        1616  FØRMAT(1X,'TØØ MANY ENTRIES IN CLUSTER SELECTIØN !!!',/)
              GØ TØ (61,6662),DEFALT
         72   JK=2
              GØ TØ 722
         720  JK=3
         722  RETURN
              END
```

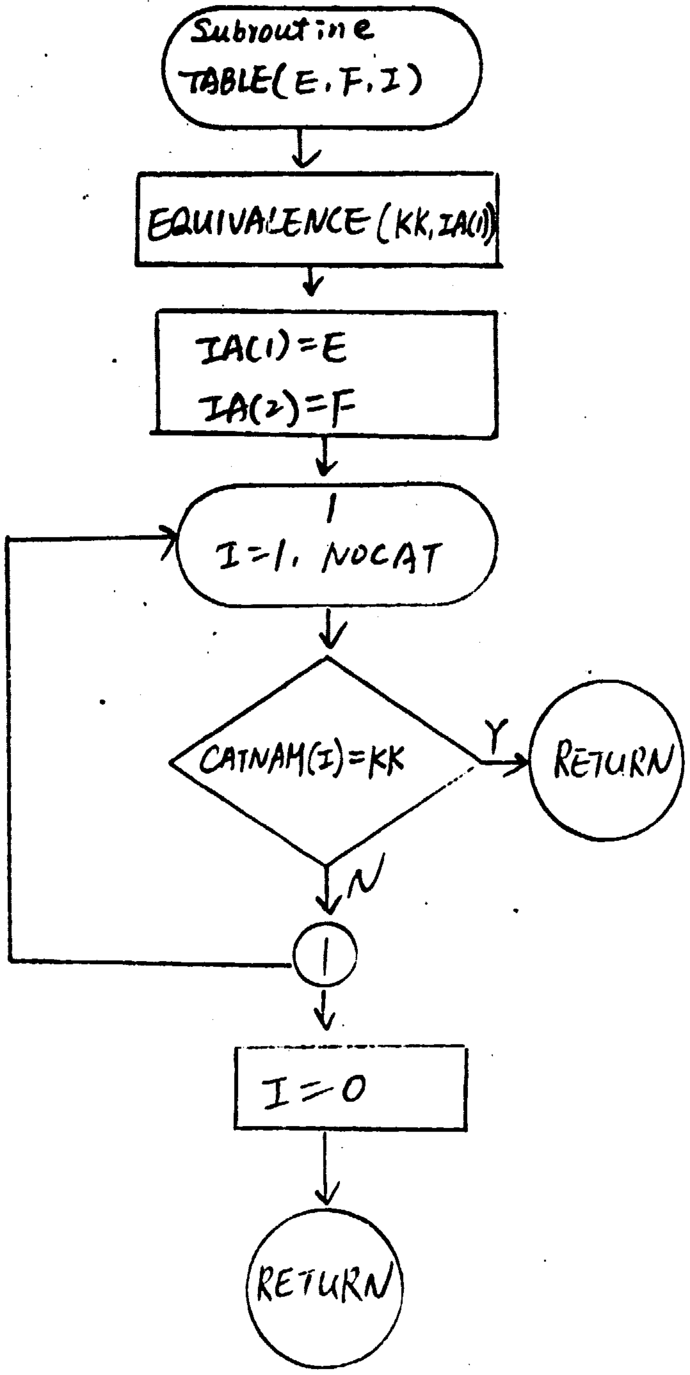
## 4.3 ENTRY POINT - TABLE

The subroutine DTCLIO uses the subroutine TABLE to search for the numerical sequence of a specific category name.

- Calling sequence

  CALL TABLE (E,F,I)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| E | Alphanumeric | - | In | First byte of category none |
| F | Alphanumeric | - | In | |
| I | Integer | - | Out | |


Subroutine TABLE(E,F,I)
EQUIVALENCE (KK,IA(1))
IA(1)=E
IA(2)=F
1
I=1, NOCAT
CATNAM(I)=KK
Y
RETURN
N
1
I=0
RETURN

```
                SUBROUTINE TABLE(E,F,I)
                IMPLICIT INTEGER (A-Z)
                INCLUDE 'SY:[300,3]CAMSCØMMØN.INC'
 *              INCLUDE 'SY:[300,3]CAMSPARAM.INC'
 *              PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
 *             1,MAXV=11,NDØTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
 *             2NØSPWD=6,NØDTWD=10
 *              EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
 *              INTEGER C1(469),C2(256),C3(71),C4(348),C5(629)
 *     C*
 *              INTEGER ACDATE,SUBCAT,SUBPØP,CATKNT,CATTH
 *              BYTE CHNVEC,NØCHAN,NØSUB,DØTCAT,DØTCLU
 *              CØMMØN/CØM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NØCHAN,NØSUB,
 *             1SUBCAT(MAXSUB),SUBPØP(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NØDØ,
 *             2NØDU,NØTH,DØTCAT(NDØTS),DØTCLU(NDØTS)
 *     C*
 *              INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DØTDAY,PDATE1,TDATE1
 *              INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDØM,GRID
 *              BYTE DELFLG,NØACQ,SØILGR,SUNEL,NSTART,NTYPE1,ALP,ALPØ
 *              BYTE PCTCT,PCTCTØ,VAR,VARØ,DLABEL,TYPE
 *              CØMMØN/CØM2/ISEG,DELFLG,NØACQ,ADATES(2,MAXACD),SØILGR(MAXACD),
 *             1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
 *             2DØTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
 *             3PDATE3(2),TDATE3(2),NØCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPØ,
 *             4             PCTCT(MAXCAT),PCTCTØ,VAR(MAXCAT),VARØ
 *     C*
 *              INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
 *             1UFLAG4
 *              INTEGER PFLAG,DSKMNT
 *              CØMMØN/CØM3/PFLAG,DSKMNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
 *             1,UFLAG2,UFLAG3,UFLAG4,NEWLAB(MAXSUB)
 *     C*
 *              INTEGER TX1,TY1,TX2,TY2,ACDISP,G,B,DTWIND,DØTARY,GMIN,GMAX,FUL
 *              INTEGER SPWIND,CLAWND,CLUWND
 *              CØMMØN/CØM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
 *             1B(4),DTWIND(5,NØDTWD),SPWIND(5,NØSPWD),IMWIND(4),NUMDØT,
 *             2DØTARY(NDØTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
 *              CØMMØN/CØM5/DISKID,RANDØM(NDØTS),GRID(NDØTS),DLABEL(NDØTS),
 *             1TYPE(NDØTS),RECLØC
                BYTE E,F,IA(2)
                EQUIVALENCE (KK,IA(1))
                IA(1)=E
                IA(2)=F
                DØ 1 I=1,NØCAT
                IF(CATNAM(I) .EQ. KK) GØ TØ 2
          1     CØNTINUE
                I=0
          2     RETURN
                END
```

ORIGINAL PAGE IS
OF POOR QUALITY

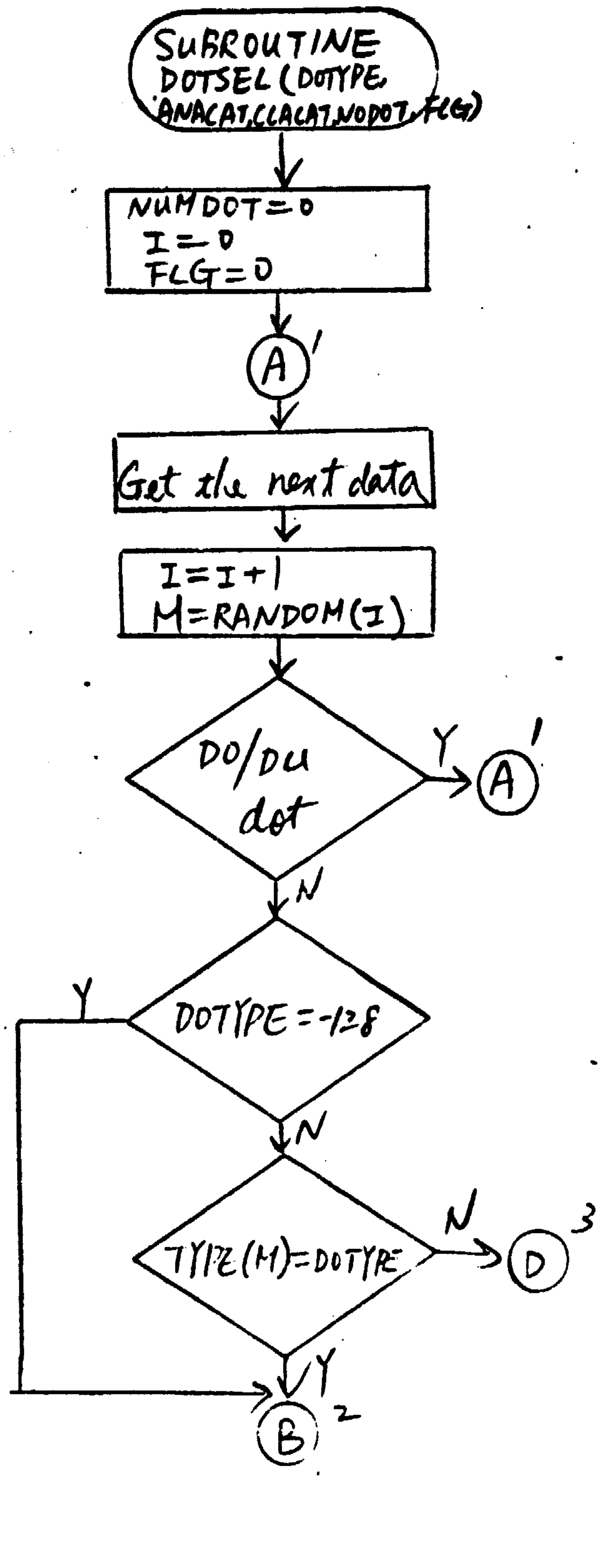
## 4.4 ENTRY POINT - DOTSEL

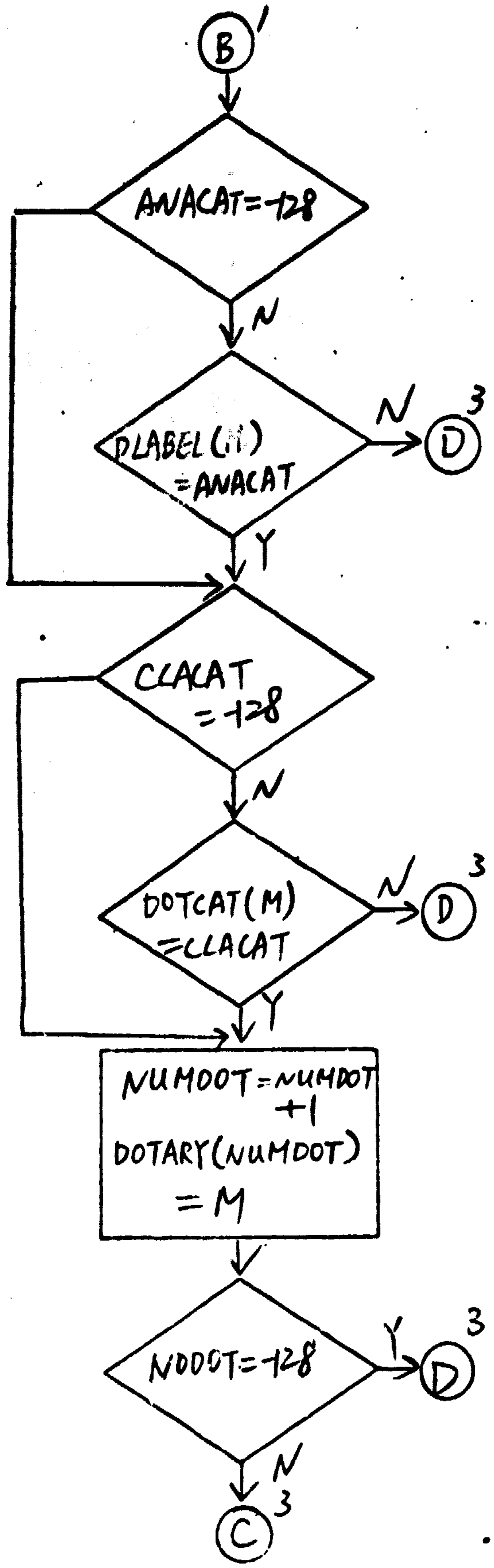
**The subroutine DTCLIO uses the subroutine DOTSEL to select the dot number and total number of dots according to the conditions specified in the calling arguments.**

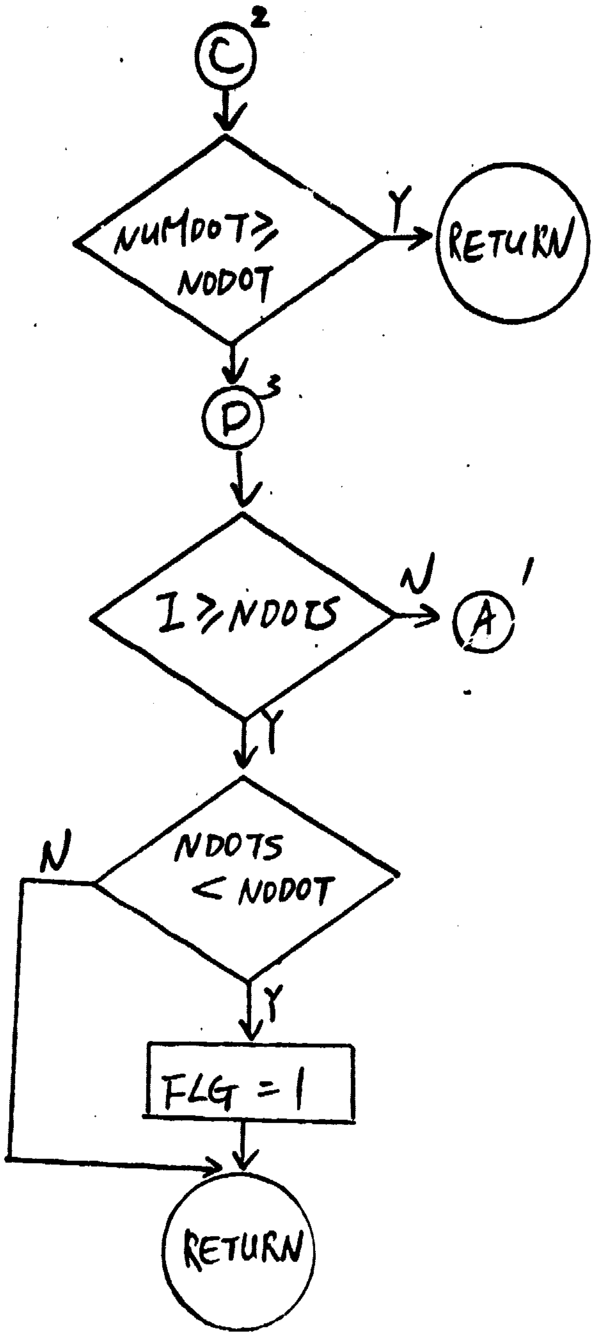
- Calling sequence

  CALL DOTSEL (DOTYPE,ANACAT,CLACAT,NODOT,FLAG)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| DOTYPE | Integer | - | In | Typed dot |
| ANACAT | Integer | - | In | Analyst labeled value |
| CLACAT | Integer | - | In | Classifies labeled value |
| NODOT | Integer | - | In | Number of dots specified by the analyst |
| FLAG | Integer | - | - | Summary variable |


SUBROUTINE DOTSEL (DOTYPE, ANACAT, CLACAT, NODOT, FLG)
NUMDOT=0
I=0
FLG=0
A 1
Get the next data
I=I+1
M=RANDOM(I)
DO/DU dot
Y
A 1
N
DOTYPE=-128
Y
N
TYPE(M)=DOTYPE
N
D 3
Y
B 2


B 1
ANACAT=-128
N
DLABEL(M) =ANACAT
N
D 3
Y
CLACAT =-128
N
DOTCAT(M) =CLACAT
N
D 3
Y
NUMDOT=NUMDOT +1
DOTARY(NUMDOT) =M
NODOT=-128
Y
D 3
N
C 3


C 2
NUMDOT ≥ NODOT
Y
RETURN
D 3
I ≥ NDOTS
N
A 1
Y
NDOTS < NODOT
N
Y
FLG = 1
RETURN

```
FØRTRAN IV-PLUS V02-04                    14115148    31-AUG-77             PAGE 1
DTCLIB.FTN         /TR:BLOCKS/WR
          SUBRØUTINE DØTSEL(DOTYPE,ANACAT,CLACAT,NØDØT,FLG)
          IMPLICIT INTEGER (A-Z)
          INCLUDE 'SY:[300,3]CAMSCØMØN.INC'
     *    INCLUDE 'SY:[300,3]CAMSPARAM.INC'
     *    PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
     *   1,MAXV=11,NDØTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
     *   2NØSPWD=6,NØDTWD=10
     *    EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
     *    INTEGER C1(469),C2(256),C3(71),C4(346),C5(629)
     * C*
     *    INTEGER ACDATE,SUBCAT,SUBPØP,CATKNT,CATTH
     *    BYTE CHNVEC,NØCHAN,NØSUB,DØTCAT,DØTCLU
     *    CØMMØN/CØM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NØCHAN,NØSUB,
     *   1SUBCAT(MAXSUB),SUBPØP(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NØDØ,
     *   2NØDU,NØTH,DØTCAT(NDØTS),DØTCLU(NDØTS)
     * C*
     *    INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DØTDAY,PDATE1,TDATE1
     *    INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDØM,GRID
     *    BYTE DELFLG,NØACQ,SØILGR,SUNEL,NSTART,NTYPE1,ALP,ALPØ
     *    BYTE PCTCT,PCTCTØ,VAR,VARØ,DLABEL,TYPE
     *    CØMMØN/CØM2/ISEG,DELFLG,NØACQ,ADATES(2,MAXACD),SØILGR(MAXACD),
     *   1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
     *   2DØTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
     *   3PDATE3(2),TDATE3(2),NØCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPØ,
     *   4          PCTCT(MAXCAT),PCTCTØ,VAR(MAXCAT),VARØ
     * C*
     *    INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
     *   1UFLAG4
     *    INTEGER PFLAG,DSKMNT
     *    CØMMØN/CØM3/PFLAG,DSKMNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
     *   1,UFLAG2,UFLAG3,UFLAG4,NEWLAB(MAXSUB)
     * C*
     *    INTEGER TX1,TY1,TX2,TY2,ACDISP,G,B,DTWIND,DØTARY,GMIN,GMAX,FUL
     *    INTEGER SPWIND,CLAWND,CLUWND
     *    CØMMØN/CØM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
     *   1B(4),DTWIND(5,NØDTWD),SPWIND(5,NØSPWD),IMWIND(4),NUMDØT,
     *   2DØTARY(NDØTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
     *    CØMMØN/CØM5/DISKID,RANDØM(NDØTS),GRID(NDØTS),DLABEL(NDØTS),
     *   1TYPE(NDØTS),RECLØC
          NUMDØT=0
          FLG=0
          NP1=1
          NP2=1
          NP3=1
          NP4=1
          DØ 99 II=1,NDØTS
          DØTARY(II)=0
   99     CØNTINUE
          IF(DØTYPE .EQ.-128) NP1=2
          IF(ANACAT .EQ. -128) NP2=2
          IF(CLACAT .EQ. -128) NP3=2
          IF(DØTYPE .EQ. -128 .AND. ANACAT .EQ. -128 .AND. CLACAT .EQ. -128
         1) NP4=2
          IF( ANACAT .EQ. -1 .ØR. ANACAT .EQ. -2) NP4=2
          DØ 5 I=1,NDØTS
          M=RANDØM(I)
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
FORTRAN IV-PLUS V02-04          14:15:48    31-AUG-77          PAGE 2
DTCLI2.FTN        /TR:BLOCKS/WR
              GO TO (67,68),NP4
        67    IF(DLABEL(M) .EQ. -1 .OR. DLABEL(M) .EQ. -2) GO TO 5
        68    GO TO (10,20),NP1
        10    IF(TYPE(M) .NE. DOTYPE) GO TO 5
        20    GO TO (11,21),NP2
        11    IF(DLABEL(M) .NE. ANACAT) GO TO 5
        21    GO TO (12,22),NP3
        12    IF(DOTCAT(M) .NE. CLACAT) GO TO 5
        22    NUMDOT=NUMDOT+1
              DOTARY(NUMDOT)=M
              IF(NODOT .EQ. -128) GO TO 5
              IF(NUMDOT .GE. NODOT) GO TO 4
         5    CONTINUE
              IF(NUMDOT .LT. NODOT) GO TO 8
              GO TO 4
         8    FLG=1
         4    RETURN
              END
```

4-44
62

## 5. [300.6] DSET.FTN

This routine will open file from specified areas of a multi-spectral image. The imagery may be in UNIVERSAL, LARSYS, or ERTS formats.

The imagery may also be either on the 'foreign' tape as received from some other system or a preprocessed PDP 11/45 named file.

- Calling sequence

```
CALL DSET (LUN,F11,FILE,MTXTFG,UNIT,EN,RWD)
```

| Argument | Dimension | In/Out | Definition |
|---|---|---|---|
| LUN | 1 (word) | In | Logical unit number. |
| F11 | 1 (word) | In | File indicator.<br>0=FILES - 11<br>1=Foreign (tape only) |
| FILE | 33 (bytes) | In | If F11=0, this is the 33 byte character string defining the complete file specification.<br>Example:<br>DK1:[100,2]XXXXX.EEE;1<br>If F11≠0, this argument is ignored. |

The following arguments are ignored if F11=0.

| | | | |
|---|---|---|---|
| MTXTFG | 1 (word) | In | Tape drive on which the operator mounted the tape.<br>0=MT tape drive<br>1=XT tape drive |
| UNIT | 1 (word) | In | Physical unit assignment made by the operator when he mounted the tape.<br>(0-3) 800 bpi<br>(4-7) 1600 bpi. We do not obtain imagery tapes in this density. |
| FN | 1 (word) | In | File number ±(0→N) |

| Argument | Dimension | In/Out | Definition |
|---|---|---|---|
| FN | 1 (word) | In | File number ±(O→N) The program will skip over FN (FN-1 if FN<0) end of file marks to position the tape to the correct file. NOTE: This relative to the current file, not the BOT. The effect is to position the tape to the FNthe file from the current position. |
| RWD | 1 (word) | In | Flag to cause rewinding of tape. 0=rewind {FN is relative to BOT FN≥) only; 1=no rewind - leave tape where is; 2=no initialization, space tape only |

```
HFØRTRAN IV-PLUS V02-04                       14117139    31-AUG-77            PAGE 1
DSET.FTN          /TR:BLØCKS/WR
                  SUBRØUTINE DSET (LUN,F11,FILE,MTYTFG,UNIT,FN,RWD)
                  IMPLICIT  INTEGER (A - Z )
                    INTEGER FFUNC
                  BYTE HEADER(1)
                  BYTE FILE(1)
                  CØMMØN /HCØM/ SS,SE,LS,LE,NRPDS,NDSPR,NCFR, NRPC,ANCL,NC,NS,
                 1 NBIT,DSI,NCAR,SVD,RSIZ,PSKIP,HSIZ,CALP,CERR
                    CØMMØN /IØLP/IØ,LP
                    BYTE BUF(1)
                  INTEGER IB(64),JR(64)
                  BYTE RUFF(1)
                  INTEGER HSIZZ(3),LSLØC(2)
                  DATA HSIZZ/ 3060,800,40/
                  DATA LSLØC/71,1 /
                    WBUF(I)=FFUNC(BUF(I+13))*256+FFUNC(BUF(I+12))
                    RBUF(I)=FFUNC(BUF(I+12))*256+FFUNC(BUF(I+13))
                  READY = 0
                    EEØF = 1
                  IF(F11.NE.0) GØ TØ 1
                    CERR = 0
                    DØ 34 I = 1,33
                    IF(FILE(I).EQ.0) GØ TØ 35
                    IF(FILE(I).EQ.' ') GØTØ 35
         34         CØNTINUE
         35         I = I-1
         D          WRITE(LP,9006)(FILE(J),J=1,I)
         D9006      FØRMAT(' ',32A1)
                    CALL FVØPEN(1,LUN,FILE,I,ISTAT,20+LUN)
                    CALL FVWAIT(LUN)
                    BUF(1)=1
                    BUF(2)=0
                    BUF(3)=0
                    BUF(4)=0
                    BUF(9)=0
                    BUF(10)=0
                    BUF(11)=0
                    BUF(12)=0
                    IF(ISTAT.EQ.0)RETURN
                    CERR = 1
                    WRITE(IØ,1006)ISTAT
         1006       FØRMAT(' FVØPEN ERRØR, ISTAT = ',I5,' DSET TERMINATES'///)
                  RETURN
         1          IF(RWD.EQ.2) GØ TØ 7
                    CALL TINIT(LUN,MTXTFG,UNIT)
                  CALL TATCH,LUN)
                  IF(RWD.NE.0)GØ TØ 7
                  CALL TRWD(LUN)
                  CALL TFILE(LUN,FN)
                    CERR = 0
                  RETURN
         7          I = FN
                    IF(I.LT.0) I = I - 1
                    CALL TFILE(LUN,I)
                    IF(I.GE.0) RETURN
                    CALL TSTAT(LUN,FNCT,I)
                    IF(IAND(FUNC,"400).NE.0)RETURN
                 CALL TFILE(LUN,1)
                 RETURN
               ENTRY DUSET (LUN,F11,RWD)
               IF(F11.NE.0) GØ TØ 2
                 CALL FVCLØS(LUN)
               RETURN
         2     IF(RWD.NE.0) GØ TØ 3
               CALL TRWD(LUN)
               GØ TØ 4
         3       I = -1
                 IF(EEØF.EQ.0)I = -2
                 EEØF = 1
                 CALL TFILE,LUN,-1)
               CALL TSTAT,LUN,FNCT,I)
               IF(IAND(FNCT,"400).NE.0) GØ TØ 4
               CALL TFILE(LUN,1)
         4     CALL TRLSE(LUN)
               RETURN
               ENTRY HREAD(LUN,HEADER,BUF,BUFSZ,FØRMAT,F11,PRTY,EØF,SHFG)
                 IF(CERR.NE.0) RETURN
                 IF(BUFSZ.GE.HSIZZ(FØRMAT))GØ TØ 42
                 CERR = 13
                 WRITE(I0,1008)BUFSZ,HSIZZ(FØRMAT)
         1008    FØRMAT(' BUFSZ = ',I5,' EED ',I5,'  READ TERMINATES.'//)
                 RETURN
         42      CØNTINUE
               CALL RREAD(LUN,F11,BUF,HSIZZ(FØRMAT),BCT,EØF,PRTY)
                 IF(EEØF.NE.0)EEØF=EØF
                 IF(EØF.EQ.0)RETURN
                 J=HSIZZ(FØRMAT)
                 DØ 32 I = 1,J
         32      HEADER(I)=BUF(12+I)
         D       J12 = J + 12
         D       WRITE(LP,5002)J,(BUF(X),X=1,J12)
                 IF(F11.NE.0) GØ TØ 33
                 TF=WBUF(-7)
                 IF(TF.EQ.FØRMAT) GØ TØ 33
                 WRITE(IØ,1007)TF,FØRMAT
         1007    FØRMAT(' FILE FØRMAT IS ',I5,' REQUESTED FØRMAT IS ',I5/
                 1 ' HREAD TERMINATES'///)
                 CERR = 2
                 RETURN
         33      CØNTINUE
               IF(BCT.EQ.HSIZZ(FØRMAT)) GØ TØ 5
               WRITE(IØ,1000)HSIZZ(FØRMAT),BCT
         1000  FØRMAT(' HEADER RECØRD SHØULD BE ',I5,' BYTES LØNG.'/
              1 ' IT IS ',I5,' BYTES LØNG.  HREAD TERMINATES.'///)
                 CERR = 3
                 RETURN
         5     CALL HPRØS(LUN,F11,HEADER,FØRMAT,EØF,PRTY,SHFG)
                 IF(CERR.NE.0)RETURN
         6       Q = RSIZ
                 IF(RSIZ.GT.BUFSZ) Q = BUFSZ
                 CALL RREAD(LUN,F11,BUF,Q,BCT,EØF,PRTY)
                 IF(EEØF.NE.0)EEØF=EØF
                 IF(FØRMAT.EQ.3)GØ TØ 6
```

```
                  LS=LSL8C(FØRMAT)
                  LS=FFUNC(BUF(LS+12))*256+FFUNC(BUF(LS+13))
        8         CALL HVFY(HEADER,FØRMAT)
        D         RS12 = RS12 + 12
        D         WRITE(LP,5002)LS,(BUF(X),X=1,RS12)
        D5002     FØRMAT(' ',I10/(' ',8(2(O3,2X),1X)))
                  IF(CERR.NE.0)RETURN
                  FSCAN=LS
                ØNEST=LS
                  CL=LS
                DSL = ANCL + NC*(NS+CALP)
                  CCAN = 1
                RETURN
                ENTRY FFIND(LUN,TX1,TY1,TX2,TY2,FØRMAT,BUF,BUFSZ)
                  IF(CERR.NE.0)RETURN
                TY = TY1
                  IF(TY1.GT.TY2)TY=TY2
                  X1=TX1
                  X2=TX2
                  IF(X1.LT.X2) GØ TØ 41
                  X1=TX2
                  X2 = TX1
        41        IF(X1.LT.SS)X1=SS
                  IF(X2.GT.SE)X2=SE
                  X1 = X1-SS+1
                  X2 = X2-SS+1
                IF(TY.LT.ØNEST) TY = ØNEST
                  FLIN=TY-MØD(TY-ØNEST,NDSPR)
                  LSKIP=((FLIN-FSCAN)/NDSPR-1)*NRPDS
                  IF(NRPDS.GT.1)LSKIP=LSKIP+NRPDS-1
                  CL=TY
                  TY=FLIN
                  ADD=(CL-TY)*DSL
                  IF(LSKIP.LT.0) GØ TØ 9
                  CL=FLIN-NDSPR
                  IF(CL.LT.ØNEST)CL=ØNEST
                  TY = CL
                  CALL RSKIP(LUN,F11,LSKIP,BUF)
        9       ANC = ANCL+SVD
        D         WRITE(LP,1500)TY,FLIN,LSKIP,NDSPR,CL,TY1,ØNEST,FSCAN
                IF(FØRMAT.EQ.1) ANC = ANC + 2
                IF(FØRMAT.EQ.3) GØ TØ 11
        10      FC = 1
                LC = NCAR
                I = 1
                  IR = 1
                DØ 12 CHAN = 1,NC
        13        CØNTINUE
                IF(IR.GT.1) ANC = J+SVD
                IF(CHAN.GE.FC.AND.CHAN.LE.LC) GØ TØ 14
                IF(CHAN.LE.LC.OR.IR.GE.NRPDS) GØ TØ 15
                FC = LC + 1
                LC = LC + NCPR
                  IR = IR + 1
                  GØ TØ 13
        14      IB(I)=(CHAN    -FC)*(NS+CALP)+ANC-1
```

```
                JR(I)=IR
        16      I = I + 1
        12      CØNTINUE
        11      READY = 1
        D         IF(FØRMAT.NE.3)WRITE(LP,5000)(IB(K),JR(K),K=1,I)
        D5000     FØRMAT('    IB        JR'/(' ',I5,3X,I5))
                RETURN
        15        WRITE(IØ,1001)CHAN
        1001      FØRMAT(' ILLEGAL CHANNEL REQUEST, ',
               1 'FFIND TERMINATES.  ( ',I5,' ) ***'/,.)
                  CERR = 4
                  RETURN
                ENTRY LREED(LUN,BUFF,BUF,BUFSZ,DLIN,F11,PRTY,EØF,FØRMAT,RCHAN)
                  IF(CERR.NE.0)RETURN
                CHAN = RCHAN
                  IF(RCHAN.GT.NC) GØ TØ 30
                IF(READY.NE.0) GØ TØ 31
                  WRITE(IØ,1005)
        1005      FØRMAT(' IMPRØPER CALLING SEQUENCE TØ LREED,  LREED TERMINATES'///)
                  CERR = 5
                  RETURN
        30      WRITE(IØ,1004)     RCHAN,NC
        1004    FØRMAT(' REQUESTED CHANNEL IS ',I5/
                  1 ' NUMBER ØF CHANNELS AVAILABLE IS ',I5,' LREED TERMINATES'///)
                  CERR=6
                  RETURN
        31      CØNTINUE
        D         WRITE(LP,1500)CL,DLIN
                IF(CL-DLIN)29,17,18
        18      WRITE(IØ,1002)CL,DLIN
        1002    FØRMAT(' CURRENT LINE IS ',I5/
               1 ' DESIRED LINE IS',I5/
               2 ' CANNØT BACKUP FILE, LREED TERMINATES.'///)
                  CERR=7
                  RETURN
        29      FLIN=DLIN-MØD(DLIN-ØNEST,NDSPR)
                LSK=((FLIN-TY)/NDSPR-1)*NRDS
        D         WRITE(LP,1500)FLIN,LSK,TY
        D1500     FØRMAT(' ',10I10)
                IF(LSK.LT.0) GØ TØ 19
                CALL RSKIP(LUN,F11,LSK,BUF)
                  IF(RSIZ.LE.BUFSZ) GØ TØ 43
                  CERR = 12
                  WRITE(IØ,1009)BUFSZ,RSIZ
        1009      FØRMAT(' BUFSZ = ',I5,' RSIZ NEEDS ',I5,' LREED TERMINATES..'//)
                  RETURN
        43        CØNTINUE
                CALL RREAD(LUN,F11,BUF,RSIZ,RCT,EØF,PRTY)
                  IF(ERØF.NE.0)ERØF=EØF
                IF(EØF.EQ.0) RETURN
                IF(2-FØRMAT)21,20,20
        20        TY=LSLØC(FØRMAT)
                  TY=FFUNC(BUF(LS+12))*256+FFUNC(BUF(TY+13))
        D         WRITE(LP,6000)TY,LSLØC(FØRMAT),(BUF(Q),Q=1,3060)
        D6000     FØRMAT(' ',2I10/(' ',8(2(Z3,2X),1X)))
                GØ TØ 22
```

```
FØRTRAN IV-PLUS V02-04          14:17:39    31-AUG-77          PAGE 5
DSET.FTN                 /TR:BLØCKS/WR
        21      TY = FLIN
        D         RS12 = RSIZ + 12
        D         WRITE(LP,6000)TY,RS12,(BUF(Q),Q=1,RS12)
        22      IF(TY.EQ.FLIN) GØ TØ 19
                  IF(IAND(FLIN,255).NE.IAND(TY,255))GØ TØ 83
                  TY = FLIN
                  GØ  TØ 19
        83        CØNTINUE
                  WRITE(IØ,1003)FLIN,TY
        1003      FØRMAT(' SCAN LINE NUMBER ERRØR'/' DESIRED LINE IS ',I10/
                  1 ' ACTUAL LINE IS ',I10,' LNEED TERMINATES'///)
                  CERR = 8
                  RETURN
        19      ADD =(DLIN-TY)*DSL
                  CL=DLIN
                CCAN = 1
        17      IF(NRPDS.LE.1) GØ TØ 23
        D         WRITE(LP,6001)NRPDS
        D6001     FØRMAT(' HØW DID I GET HERE?????  'I5)
                I = JR(CHAN)-JR(CCAN)
                  CCAN = CHAN
                IF(I.EQ.0) GØ TØ 24
                CALL RSKIP(LUN,F11,I-1,BUF)
                CALL RREAD(LUN,F11,BUF,RSIZ,RCT,EØF,PRTY)
                  IF(EØF.NE.0)EEØF=EØF
                IF(EØF.EQ.0) RETURN
        24      ADD= 0
        23      IF(FØRMAT .EQ.3) GØ TØ 25
                B = ADD+IB(CHAN)+ X1
                E = B+ X2- X1
        D         WRITE(LP,5001)B,E,ADD,X1,X2,SS
        D5001     FØRMAT(' ',6I5)
                K = 0
                DØ 26 I = B,E
                K = K + 1
        26      BUFF(K)=BUF(I+12)
                RETURN
        25      B=(( X1-1)/2)*8+CHAN*2-MØD( X1,2)
                E=(( X2-1)/2)*8+CHAN*2-MØD( X2,2)
                J = 1+MØD( X1,2)
                K = 0
        28      DØ 27 I = 1,J
                K = K + 1
                BUFF(K)=BUF(B+12)
                B = B + 1
        27      IF(B.GT.E)RETURN
                J = 2
                B = B + 6
                GØ TØ 28
                END
```

## 6. [300,6] DSKCHK.FTN

**This subroutine validates the existence of the current segment number and determines whether or not the current disk pack will accommodate the segment.**

- **Calling sequence**

CALL DSKCHK (SEGNO,PTR,DSKID,FLAG)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| SEGNO | Integer | 1 | In | Current segment number |
| PTR | Integer | 1 | Out | If flg=1, disk pack to be mounted |
| DSKID | Integer | 1 | Out | Available disk pack |
| FLAG | Integer | 1 | Out | 0=good segment, right disk pack<br>1=valid segment wrong disk<br>2=invalid segment |

```
FØRTRAN IV-PLUS V02-04                    14:18:35    31-AUG-77           PA
DSKCHK.FTN          /TR:BLØCKS/WR
                    SUBRØUTINE DSKCHK(SEGNØ,PTR,DSKID,FLAG)
                    IMPLICIT INTEGER (A-Z)
                    DIMENSIØN KPAT(2,200),BUFF1(23)
                    ØPEN(UNIT=1,NAME='DB0:[300,300]DSKTBL.DAT',ACCESS='DIRECT',
               *    TYPE='ØLD',MAXREC=1,RECØRDSIZE=200)
                    ØPEN(UNIT=2,NAME='DB2:[310,300]HDRREC.DAT',
               *      ACCESS='SEQUENTIAL',TYPE='ØLD',READØNLY)
                    FLAG=0
                    PTR=-9
                    READ(2,100) BUFF1
          100       FØRMAT(23A2)
                    DECØDE(4,101,BUFF1) CURDSK
          101       FØRMAT(I4)
                    READ (1'1) KPAT
                    DØ 2 J=1,200
                    IF(SEGNØ .EQ. KPAT(1,J)) GØ TØ 10
            2       CØNTINUE
                    FLAG=2
                    GØ TØ 25
           10       IF(KPAT(2,J) .EQ. CURDSK) GØ TØ 20
                    FLAG=1
                    DSKID=KPAT(2,J)
                    GØ TØ 25
           20       PTR=J
                    DSKID=KPAT(2,J)
           25       CLØSE(UNIT=1,DISPØSE='SAVE')
                    CLØSE(UNIT=2,DISPØSE='SAVE')
                    RETURN
                    END
```

## 7. [300,6] ELAPSE.FTN

### 7.1 ENTRY POINT - ELAPSE

The subroutine prints out the elapse time between the initial and final calls.

- Calling sequence:

  CALL ELAPSE(II)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| II | I | 1 | In | II=1 Initial call<br>II=2 Final call |

```
HFØRTRAN IV-PLUS V02-04                      14:19:28    31-AUG-77            PAGE 1
ELAPSE.FTN      /TR:BLØCKS/WR
          SUBRØUTINE ELAPSE(II)
      C
      C   SUBRØUTINE TØ PRINT ELAPSED TIME
      C
      C
      C   FUNCTIØN:PRINT ØUT ELAPSED TIME
      C   DATE:APRIL 6,1977
      C   PRØGRAMMER:PAUL LIN
      C   CALLING SEQUENCE:CALL ELAPSE(II)
      C                   WHERE II=1 INITIAL CALL
      C                         II=2 FINAL CALL
      C   ØUTPUT:TØTAL ELAPSED TIME BETWEEN INITIAL AND FINAL CALLS
      C   EXAMPLE:C
      C           C    MAIN PRØGRAM
      C           C
      C                II=1
      C                CALL ELAPSE(II)
      C           C
      C           C    MAIN LØGIC
      C           C
      C                II=2
      C                CALL ELAPSE(II)
      C                STØP
      C                END
          REAL T1,DELTA
          REAL SCR
          INTEGER JSCR,ISEC,IMIN,IHR
          GØ TØ (1,2),II
      C
      C   INITIAL CLALL
      C
        1 T1=SECNDS(0.)
          GØ TØ 9999
      C
      C   FINAL CALL
      C
        2 DELTA=SECNDS(T1)
          JSCR=DELTA/60.
          SCR=JSCR*60.
          ISEC=DELTA-SCR+0.5
          IHR=JSCR/60
          IMIN=JSCR-IHR*60
          TYPE 1000,IHR,IMIN,ISEC
     1000 FØRMAT(' TØTAL ELAPSED TIME= ',I2,':',I2,':',I2)
     9999 RETURN
          END
```

ORIGINAL PAGE IS
OF POOR QUALITY

## 8. ERRMES

See appendix A for a description of the program.

```
MERRMES -- PRINTS DIRECTIVE, I/Ø        MACRØ D10    31-AUG-77 14:20
TABLE OF CØNTENTS

    1-  2        INTRØDUCTIØN
    1- 49        DATA AREA
    1- 89        DIRERR -- PRINTS DIRECTIVE ERRØR MESSAGES,
    1-120        IØERR -- PRINTS I/Ø ERRØR MESSAGES,
    1-168        FCSERR -- PRINTS ERRØR MESSAGES FØR FCS ERRØRS,
        .TITLE  ERRMES -- PRINTS DIRECTIVE, I/Ø, AND FCS ERRØR MESSAGES.
        .SBTTL  INTRØDUCTIØN
;
;               THIS PACKAGE ØF SUBRØUTINES PRINTS MESSAGES FØR DIRECTIVE,
;               I/Ø, AND FILE PRØCESSING (FCS) ERRØRS. THE APPRØPRIATE MESSAGE IN
;               [1,2]QIØSYM.MSG IS PRINTED ALØNG WITH THE PRØGRAM CØUNTER
;               (PC) AT THE PØINT FRØM WHICH THE SUBRØUTINE WAS CALLED.
;               THE MØ (MESSAGE ØUTPUT) HANDLER IS USED FØR ALL
;               ØPERATIØNS.
;
;               THE FØLLØWING ENTRY PØINTS ARE PRØVIDED:
;
;               DIRERR - PRINTS THE MESSAGE FØR THE ERRØR INDICATED
;                        IN THE DIRECTIVE STATUS WØRD (DSW) AT VIRTUAL
;                        LØCATIØN 0, ALØNG WITH THE VALUE ØF THE PC
;                        ØBTAINED FRØM THE STACK. "DIRERR" MAY BE
;                        USED AS THE ERRØR HANDLING SUBRØUTINE ADDRESS IN
;                        DIRECTIVE CALLS. AFTER THE MESSAGE IS PRINTED,
;                        THE TASK IS CØNTINUED.
;
;               IØERR  - PRINTS THE MESSAGE FØR THE ERRØR INDICATED IN THE
;                        I/Ø STATUS BLØCK (PRØVIDED AS AN ARGUMENT),
;                        ALØNG WITH THE PC ØBTAINED FRØM THE STACK, AND THE
;                        THE I/Ø STATUS BLØCK. THE TASK EXECUTIØN CØNTINUES
;                        IN THE CASE ØF AN END-ØF-FILE; ØTHERWISE THE TASK
;                        IS CØNTINUED.
;
;               FCSERR - PRINTS THE MESSAGE FØR THE ERRØR INDICATED BY THE
;                        FILE DESCRIPTØR BLØCK (FDB) PØINTED TØ BY R0,
;                        ALØNG WITH THE PC ØBTAINED FRØM THE STACK AND THE
;                        ERRØR CØDE, FILENAME, AND LUN. IF THE ADDRESS ØF AN
;                        I/Ø STATUS BLØCK IS PRØVIDED IN THE FDB, IT TØØ IS
;                        PRINTED. AFTER THE MESSAGE IS PRINTED, THE TASK IS
;                        CØNTINUED. "FCSERR" MAY BE USED AS THE ERRØR HANDLING
;                        SUBRØUTINE ADDRESS IN FCS CALLS.
;
;               NØTE: IF THESE SUBRØUTINES ARE INCLUDED IN A TASK THAT
;                     CØNTAINS ØNLY MACRØ-11 GENERATED CØDE, THE GLØBAL
;                     SYMBØL ".MØLUN" MUST BE DEFINED ELSEWHERE (FØR EXAMPLE,
;                     BY INCLUDING "MØDEF.ØBJ" AS INPUT TØ THE TASK BUILD).
;
;                     WHENEVER THE TASK IS CØNTINUED, IT MAY BE RESUMED BY
;                     ENTERING:
;                                         CØN "TASKNAME"
;                     TØ MCR.
;
;                       JØHN T. DALTØN, CØDE 933            29 ØCT 1975
;
        .SBTTL  DATA AREA
        .MCALL  MØDF$,MØUT$,WTSE$,FDØF$L,DIR$
        MØDF$                           ; DEFINE MØ FLAGS.
        FDØF$L                          ; DEFINE FDB ØFFSETS.
WTSE:   WTSE$   30.                     ; DPB TØ WAIT FØR MØ HANDLER CØMPLETIØN
MØUT:   MØUT$   IØESTR,PARAM,0,CØNT,SYLØC,BUF,134,,MIØST,0
MIØST:  .WØRD   0,0
;       FILE NAME STRING FØR ERRØR MESSAGE SØURCE.
MFILE:  .WØRD   20$-10$                 ; LENGTH ØF FILE NAME STRING.
        .WØRD   10$                     ; @(FILE NAME STRING)
10$:    .ASCIZ  /SY:[1,2]QIØSYM.MSG/
20$:
;       FØRMAT STRING FØR DIRECTIVE ERRØRS.
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        .EVEN
DIRSTR: .WØRD   20$-10$              ; LENGTH ØF FØRMAT STRING.
        .WØRD   10$                  ; @(FØRMAT STRING)
10$:    .ASCIZ  /DIRECTIVE ERRØR -- PC = %1P%1N%VA/
20$:
;       FØRMAT STRING FØR I/Ø ERRØRS.
        .EVEN
IØESTR: .WØRD   20$-10$
        .WØRD   10$                  ; @(FØRMAT STRING)
10$:    .ASCII  &%VA%1NPC = %1P%1NI/Ø STATUS BLØCK:&
        .ASCIZ  / %1P,%1D %1D (%1P)Ø/
20$:
;       FØRMAT STRING FØR FCS ERRØRS.
        .EVEN
FCSSTR: .WØRD   20$-10$              ; LENGTH ØF FØRMAT STRING.
        .WØRD   10$                  ; @(FØRMAT STRING)
FCSL1=20$-10$            ; LENGTH ØF FMT STR W/Ø I/Ø STATUS BLK
FCSL2=30$-10$            ; LENGTH ØF FMT STR W/ I/Ø STATUS BLK
10$:    .ASCII  /FCS ERRØR: PC = %1P/
        .ASCII  /%1N%VA%1N%2P %2A%1D:%1X, LUN=%1D/
20$:
        .ASCIZ  /%1N%2P/
30$:
        .EVEN
PARAM:  .BLKW   20.                  ; PARAMETER AREA.
SAVE:   .BLKW   2                    ; TEMPØRARY STØRAGE FØR REGS.
BUF:    .BLKB   134.                 ; BUFFER FØR MESSAGE.
.SBTTL  DIRERR -- PRINTS DIRECTIVE ERRØR MESSAGES.
;
;               ENTRY: CALL DIRERR
;
DIRERR::
        MØV     @#0,MØUT+M,ØNUM             ; MØVE ERRØR CØDE TØ DPB.
        BGE     10$                  ; RETURN IF NØ ERRØR.
        NEG     MØUT+M,ØNUM          ;CØNVERT TØ RECØRD NUMBER.
        ADD     #128.,MØUT+M,ØNUM           ; DIRECTIVE ERRØRS ØFFSET BY 128.
        MØV     #MFILE,MØUT+M,ØSTR          ; FILE NAME STRING ADDRESS
        MØV     .MØLUN,MØUT+M,ØLUN          ; LUN
        CLR     MØUT+M,ØPRM          ; NØ PARAMETERS
        MØVB    #BUSFFR,MØUT+M,ØDST         ; GET MESSAGE IN BUFFER.
        MØV     #BUF,MØUT+M,ØBUF
        MØV     #134.,MØUT+M,ØSIZ
        MØVB    #CSØNT,MØUT+M,ØACT          ; CØNTINUE AFTER ØPERATIØN
        DIR$    #MØUT                ; GET ERRØR MESSAGE IN BUFFER
        DIR$    #WTSE                ;   AND WAIT FØR CØMPLETIØN.
        MØV     (SP),PARAM           ; MØVE PC TØ PARAMETER LIST.
;       MØVE MESSAGE STRING PARAMETERS TØ PARAMETER LIST
;       (ASSUME A ØNE-RECØRD MESSAGE).
        MØV     BUF+2,PARAM+2        ; MESSAGE LENGTH IN BYTES
        MØV     #BUF+4,PARAM+4       ; @(MESSAGE STRING).
        CLR     MØUT+M,ØNUM          ; FØRMAT STRING IS IN CØRE.
        MØV     #DIRSTR,MØUT+M,ØSTR         ; FØRMAT STRING ADDRESS
        MØV     #PARAM,MØUT+M,ØPRM          ; PARAMETER STRING ADDRESS
        MØVB    #SY$STM!HE$ADR,MØUT+M,ØDST        ; PRINT MESS W/ HDR
        MØVB    #CSØNT,MØUT+M,ØACT          ; CØNTINUE AFTER ØPERATIØN
        DIR$    #MØUT
        DIR$    #WTSE
10$:    RTS     PC
.SBTTL  IØERR -- PRINTS I/Ø ERRØR MESSAGES.
;
;               ENTRY:  CALL IØERR(IØST)
;
;       WHERE IØST IS THE TWØ-WØRD I/Ø STATUS BLØCK.
;
IØERR::
        MØV     R0,SAVE              ; SAVE REGISTERS USED.
        MØV     R1,SAVE+2
        MØV     2(R5),R0        ; @(IØST)
        MØVB    (R0),R1         ; GET ERRØR CØDE
        BGE     20$             ; RETURN IF +< ERRØR.
        NEG     R1              ; RECØRD # FØR QIØSYM.MSG
;       SET UP MØ DPB (MØUT) TØ GET MESSAGE FRØM QIØSYM.MSG
        MØV     R1,MØUT+M,ØNUM  ; RECØRD NUMBER
        MØV     .MØLUN,MØUT+M,ØLUN      ;LUN
        CLR     MØUT+M,ØPRM     ;NØ PARAMETERS
        MØV     #MFILE,MØUT+M,ØSTR      ; FILE NAME STRING ADDRESS
        MØVB    #BUSFFR,MØUT+M,ØDST     ; GET MESSAGE IN BUFFER
        MØV     #BUF,MØUT+M,ØBUF        ; @(BUFFER)
        MØV     #134.,MØUT+M,ØSIZ       ; SIZE ØF BUFFER
        MØVB    #C$ØNT,MØUT+M,ØACT      ;CØNTINUE
        DIR$    #MØUT
        DIR$    #WTSE           ; WAIT FØR MESSAGE CØMPLETIØN
        MØV     BUF+2,PARAM     ; LENGTH ØF MESSAGE (BYTES)
        MØV     #BUF+4,PARAM+2  ; @(MESSAGE TEXT)
        MØV     (SP),PARAM+4    ; PC FRØM STACK
        CLR     PARAM+8.                ; ERRØR CØDE TØ PARAMETER LIST
        MØVB    (R0),R1
        MØV     R1,PARAM+8.
        CLR     PARAM+6         ; MØVE BYTE 1 ØF WØRD 1 TØ
        MØVB    1(R0),PARAM+6   ; PARAMETER LIST.
        MØV     2(R0),PARAM+10. ;MØVE WØRD 2 ØF IØST FØR PRINTING
        MØV     2(R0),PARAM+12. ;  IN DECIMAL AND ØCTAL.
        CLR     MØUT+M,ØNUM     ; INDICATE FØRMAT STRING IN CØRE.
;       SET UP MØ DPB FØR MESSAGE PRINTING.
        MØV     #IØESTR,MØUT+M,ØSTR     ; FØRMAT STRING ADDRESS
        MØV     #PARAM,MØUT+M,ØPRM      ; PARAMETER STRING ADDRESS.
        MØVB    #SY$STM!HE$ADR,MØUT+M,ØDST      ; PRT ØN SYLØG W/ HDR
;       IF EØF, CØNTINUE AFTER PRINTING, ØTHERWISE CØNTINUE.
        CMPB    #-10.,(R0)      ; EØF?
        BEQ     10$
        MØVB    #C$ØNT,MØUT+M,ØACT      ; NØ - CØNTINUE.
10$:    DIR$    #MØUT
        DIR$    #WTSE           ;WAIT FØR MESSAGE CØMPLETIØN
        MØV     SAVE,R0         ; RESTØRE REGISTERS
        MØV     SAVE+2,R1
20$:    RTS     PC              ;RETURN
.SBTTL  FCSERR -- PRINTS ERRØR MESSAGES FØR FCS ERRØRS.
;
;               ENTRY:  CALL FCSERR
;
;       THE ADDRESS ØF THE FILE DESCRIPTØR BLØCK (FDB) IN USE IS
;       ØBTAINED FRØM R0.
;
;       THE FØLLØWING MESSAGE IS PRINTED ØN TI:
;
;               FCS ERRØR: PC = <ADDRESS>
;               <ERRØR MESSAGE TEXT>
;               <F.ERR> <F.ERR+1> <FILE NAME>, LUN=<LUN>
;               <2-WØRD I/Ø STATUS BLØCK (ØNLY IF ADDRESS IN FDB)>
;
;       THE TASK IS THEN CØNTINUED.
;
FCSERR::
        MØV     R1,SAVE         ; SAVE R1
        MØVB    F.ERR(R0),R1    ;GET ERRØR CØDE
        BGE     20$             ; RETURN IF +< ERRØR
        NEG     R1              ; GET RECØRD NUMBER ØF MESSAGE
        TSTB    F.ERR+1(R0)     ;CHECK FØR DIRECTIVE ERRØR
        BGE     10$
        ADD     #128.,R1        ;DIRECTIVE ERRØR - ADD 128 TØ REC #
10$:    MØV     R1,MØUT+M,ØNUM
        MØV     #MFILE,MØUT+M,ØSTR      ;FILENAME
        MØV     .MØLUN,MØUT+M,ØLUN          ;LUN
        CLR     MØUT+M,ØPRM        ;NØ PARAMETERS
        MØVB    #BUSFFR,MØUT+M,ØDST       ; GET MESSAGE IN BUFFER
        MØVB    #CSØNT,MØUT+M,ØACT        ; AND CØNTINUE
        MØV     #BUF,MØUT+M,ØBUF          ; @(BUFFER)
        MØV     #134.,MØUT+M,ØSIZ         ; SIZE ØF BUFFER
        DIRS    #MØUT              ; GET MESSAGE
        DIRS    #WTSE              ; WAIT FØR CØMPLETIØN
        MØV     (SP),PARAM         ; MØVE PC TØ PARAMETER LIST
        CLR     PARAM+6            ; MØVE F.ERR
        MØVB    F.ERR(R0),PARAM+6
        CLR     PARAM+8.           ;     AND F.ERR+1 TØ PARAMETER LIST.
        MØVB    F.ERR+1(R0),PARAM+8.
        MØV     R0,PARAM+10.              ; CØMPUTE @(DEVICE NAME)
        ADD     #F.FNB,PARAM+10.
        ADD     #N.DVNM,PARAM+10.
        MØV     F.FNB+N.UNIT(R0),PARAM+12.       ; UNIT #
;       MØVE FILENAME FRØM FILENAME BLØCK TØ PARAMETER LIST.
        MØV     F.FNB+N.FNAM(R0),PARAM+14.       ;FILENAME
        MØV     F.FNB+N.FNAM+2(R0),PARAM+16.
        MØV     F.FNB+N.FNAM+4(R0),PARAM+18.
        MØV     F.FNB+N.FTYP(R0),PARAM+20.       ; FILE TYPE
        MØV     F.FNB+N.FVER(R0),PARAM+22.       ; VERSIØN NUMBER
        CLR     PARAM+24.
        MØVB    F.LUN(R0),PARAM+24.              ; LUN
        MØV     #FCSL1,FCSSTR      ; SET LENGTH ØF SHØRT FMT STR
;   IF I/Ø STATUS BLØCK ADDRESS IS PRØVIDED IN FDB, PRINT IT TØØ.
        TST     F.BKST(R0)         ; I/Ø STATUS BLK PRØVIDED?
        BEQ     15$
        MØV     #FCSL2,FCSSTR.     ; LENGTH ØF LØNG FMT STR
        MØV     F.BKST(R0),R1      ; @(I/Ø STAT BLK)
        MØV     (R1)+,PARAM+26.
        MØV     (R1),PARAM+28.
;       SET UP MØ DPB (MØUT) FØR MESSAGE PRINTING.
15$:    CLR     MØUT+M.ØNUM        ;FØRMAT STRING IN CØRE
        MØV     BUF+2,PARAM+2      ; LENGTH ØF MESSAGE TEXT.
        MØV     #BUF+4,PARAM+4     ; @(MESSAGE TEXT)
        MØV     #FCSSTR,MØUT+M.ØSTR        ;FØRMAT STRING ADDRESS
        MØV     #PARAM,MØUT+M.ØPRM
        MØVB    #SYSSTM!HESADR,MØUT+M.ØDST
        MØVB    #CSØNT,MØUT+M.ØACT        ;CØNTINUE
        DIRS    #MØUT              ; PRINT MESSAGE
        DIRS    #WTSE              ;WAIT FØR CØMPLETIØN
        MØV     SAVE,R1            ; RESTØRE R0
20$:    RTS     PC
        .END
```

## 9. [300,6] FFFPI.FTN

The subroutine FFFPI converts and returns in real format the first number set off by blanks or commas in array A starting after byte I.

- Calling sequence

```
I = 0

CALL FFFPI (I,A,N,V)
```

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| I | I | 1 | In/Out | Pointer, zero before usage first time |
| A | B | N | In | Input array to convert |
| N | I | 1 | In | Size of A |
| V | R | 1 | Out | Returned data |

```
FØRTRAN IV-PLUS V02-04                14:21:58    31-AUG-77           PAGE 1
FFFPI.FTN            /TR:BLOCKS/WR
                     SUBRØUTINE FFFPI(P,A,N,FPN)
                     INTEGER FA,FB
                     INTEGER P,N,FØRM(7),SKIP,FX,PP1,IP1
                     BYTE A(74)
                     REAL FPN
                     DATA FØRM/'( ',' ',' ',' F',' ','.0',' )'/
                     IDP = 0
                     PP1 = P+ 1
                     IF(PP1.GT.N) GØ TØ 7
                     DØ 1 I = PP1,N
                     IF(A(I).EQ."56) GØ TØ 22
                     IF(A(I).EQ."53) GØ TØ 2
                     IF(A(I).EQ."55) GØ TØ 2
          C          IF(A(I).EQ."54) GØ TØ 77
                     IF(A(I).LT."60) GØ TØ 1
                     IF(A(I).LE."71) GØ TØ 2
          1          SKIP = I
          77         P = I
          7          FPN = 0
                     RETURN
          22         IDP = 1
          2          SKIP = I-1
                     IP1 = SKIP + 2
                     I10 = 0
                     IF(IP1.LE.N) GØ TØ 8
                     J = IP1
                     GØ TØ 4
          8          DØ 3 J = IP1,N
                     JJ = J
          C          IF(A(J).EQ."40) GØ TØ 44
          C          IF(A(J).EQ."54) GØ TØ 44
                     IF(A(J).EQ.' ') GØ TØ 4
                     IF(A(J).EQ.',') GØ TØ 4
                     IF(A(J).EQ."53) GØ TØ 4
                     IF(A(J).EQ."55) GØ TØ 4
                     IF(A(J).NE."56) GØ TØ 3
                     IF(IDP.EQ.1) GØ TØ 4
                     IDP = 1
          3          CØNTINUE
          4          P = JJ - 1
                     FX = P-SKIP-I10
                     IF(SKIP.LE.0) GØ TØ 5
                     FØRM(3)='X,'
                     FA = SKIP/10
                     FB=SKIP-FA*10
                     FØRM(2)=IØR((FB+"60)*256,FA+"60)
                     GØ TØ 6
          C44        JJ = JJ + 1
          C          I10 = 1
          C          GØ TØ 4
          5          FØRM(2) = '  '
                     FØRM(3)= '  '
          6          FA = FX/10
                     FB = FX-FA*10
                     FØRM(5)=IØR((FB+"60)*256,FA+"60)
                     DECØDE(P,FØRM,A,ERR=9)FPN
```

~~9-2~~
87

```
FORTRAN IV-PLUS V02-04             14121158    31-AUG-77           PAGE 2
FFFPI.FTN          /TRIBLOCKS/WR
0051               RETURN
0052     9         FPN = 0.
0053               RETURN
0054               END
```

~~9-3~~
82

# 10. FFUNC.FTN

## 10.1 ENTRY PERIOD - FFUNC

Given a byte it converts to integer, masks off any sign extension and returns it to caller as function.

- Calling sequence

  I = FFUNC(B)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| B | B | 1 | In | Byte to be converted |

```
FORTRAN IV-PLUS V02-04                          14:22:47    31-AUG-77            PAGE 1
FFUNC.FTN            /TR:BLOCKS/WR
                     INTEGER FUNCTION FFUNC(A)
                     INCLUDE 'TOP.INC'
      *  C:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:C
      *  C                                                                         C
      *  C                                                                         C
      *  C                                                                         C
      *  C                                                                         C
         C                CONVERT BYTE TO NON-SIGN EXTEND INTEGER
         C                T. KELL/LEC/4/77
         C                FFUNC.FTN                                                  C
                     INCLUDE 'BOT.INC'
      *  C                                                                         C
      *  C                                                                         C
      *  C                                                                         C
      *  C                                                                         C
      *  C:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:CC:C
                     BYTE A
                     INTEGER I
                     I=A
                     I=IAND(I,"377)
                     FFUNC = I
                     RETURN
                     END
```

## 11. [300,6]FLGDOT.FTN

### 11.1 ENTRY POINT - FLGDOT

The subroutine FLGDOT uses the subroutine FDLINT to determine and flag all dots lying within predetermined designated 'other' and 'unidentifiable' fields.

- Calling Sequence
  CALL FLGDOT(NOFLD,NV,VERTEX,FLDLAB,DLABEL)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| NOFLD | I | | IN | Number of fields. |
| NV | I | MAXFLD | IN | Number of vertices in each field. |
| VERTEX | I | (2,MAXV, MAXFLD) | IN | Spatial (pixel, line) coordinate of each vertex. |
| FLDLAB | I | MAXFLD | IN | Label or 'type' designator for each field. -1=DØ field -2=DU field |
| DLABEL | I | NDOTS | IN/OUT | Analyst labels for each dot. Category index numbers. |

```
FLGDOT.FTN      /TR:BLOCKS/WR
            SUBROUTINE FLGDØT (NØFLD,NV,VERTEX,FLDLAB,DLABEL)
      C        FLAGS DØTS WHICH LIE WITHIN DØ/DU FIELDS
      C        WRITTEN BY RUTH MINTER
      C        NØFLD  - NØ. ØF FIELDS
      C        NV     - NØ. ØF VERTICES IN EACH FIELD
      C        VERTEX - SPATIAL (PIXEL,LINE) CØØRDINATE ØF EACH VERTEX
      C        FLDLAB - TWØ CHARACTER LABEL (DØ ØR DU) FØR EACH FIELD
      C        DLABEL - CATEGØRY INDICATØR FØR EACH DØT
            IMPLICIT INTEGER (A-Z)
              INCLUDE 'SY:[300,3]CAMSPARAM.INC'
            PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
           1,MAXV=11,NDØTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
           2NØSPWD=6,NØDTWD=10
            DIMENSIØN NV(MAXFLD), VERTEX(2,MAXV,MAXFLD),
           1  FLDLAB(MAXFLD),DLABEL(NDØTS),FL(8),LB(MAXFLD),LE(MAXFLD)
            DIMENSIØN SB(MAXFLD),SE(MAXFLD)
      C      RESET ALL DØTS WITH DØ ØR DU LABEL
            DØ 10 I=1,NDØTS
            IF (DLABEL(I) .EQ. -1) DLABEL(I)=0
            IF (DLABEL(I) .EQ. -2) DLABEL(I)=0
         10 CØNTINUE
              DØ 15 I=1,MAXFLD
              LB(I)=1000
              LE(I)=0
              SB(I)=1000
              SE(I)=0
         15 CØNTINUE
            DØ 20 J=1,NØFLD
            NVT=NV(J)
            DØ 20 K=1,NVT
            LB(J)=MIN(LB(J),VERTEX(2,K,J))
            LE(J)=MAX(LE(J),VERTEX(2,K,J))
            SB(J)=MIN(SB(J),VERTEX(1,K,J))
            SE(J)=MAX(SE(J),VERTEX(1,K,J))
         20 CØNTINUE
              NP=NPIX/DSSKIP
            DØ 50 J=1,NØFLD
            DØT=0
            DØ 40 LINE=10,NLIN,DLSKIP
            DØ 30 IS=10,NPIX,DSSKIP
              DØT=(LINE/DLSKIP-1)*NP+IS/DSSKIP
              IF (LINE .LT. LB(J)) GØ TØ 40
              IF (LINE .GT. LE(J)) GØ TØ 40
            IF (IS .LT. SB(J)) GØ TØ 30
            IF (IS .GT. SE(J)) GØ TØ 30
            CALL FDLINT (VERTEX(1,1,J),NV(J),FL,
           1  LINE,NSAMP,JJ)
              DØ 30 I=1,JJ,2
              IF (IS .LT. FL(I)) GØ TØ 30
              IF (IS .GT. FL(I+1)) GØ TØ 30
              DLABEL(DØT)=FLDLAB(J)
         30 CØNTINUE
         40 CØNTINUE
         50 CØNTINUE
            RETURN
            END
```

## 11.2 <u>ENTRY POINT - FDLINT</u>

This subroutine returns the pixel intercepts on a given scan line for an irregular field with a maximum of 10 vertices.

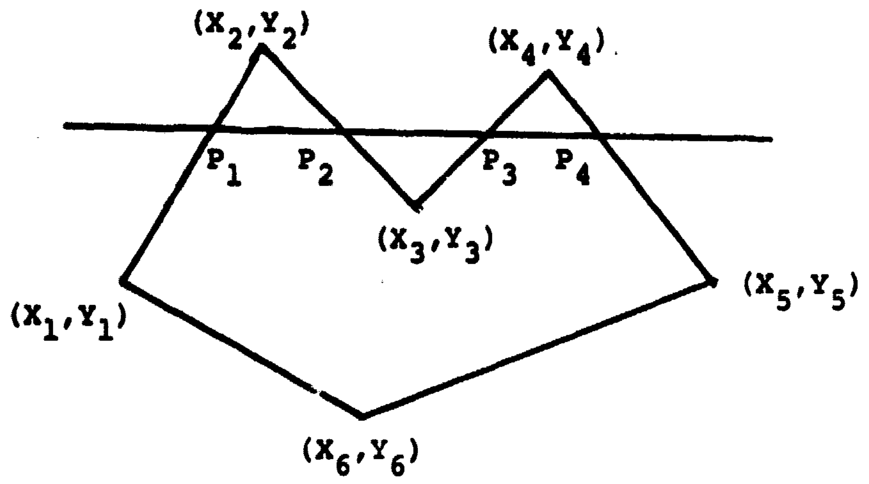
The pixel intercept, X, with the scan line L and the side defined by vertices $(X_1,Y_1)$ and $(X_2,Y_2)$ is calculated by the equation:

$$X = \frac{(L-Y_1)(X_2-X_1)}{(Y_2-Y_1)} + X_1$$

The value of X is computed as a floating point number. However, the actual pixel intercept must be an integer number. Therefore, if the fractional part of X is greater than one half, then the pixel intercpet is the next higher integer number. If the fractional part of X is less than one half, then the pixel intercept is the next lower integer number. When the fractional part of X is exactly one half, the integer pixel intercept depends on the direction of movement from the point $(X_1,Y_1)$ to $(X_2,Y_2)$. If $Y_1$ is less than $Y_2$, the pixel intercept is the next higher integer. If $Y_1$ is greater than $Y_2$, the pixel intercept is the next lower integer number.

After all intercepts for a given scan line have been determined, the intercepts are taken in pairs and all pixels bet[illegible] and including the pair of intercepts are included in th[illegible]ld.

**Example:**



For scan line L, all pixels between, and including $P_1$, and $P_2$ are included and all pixels between, and including $P_3$ and $P_4$ are included.

o **Calling Sequence**
```
CALL FDLINT(FIELD,NPTS,FL,YLINE,NSAMP,JJ)
```

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| FIELD | I | (2,NPTS) | IN | The vertices defining the field. Pixel number followed by line number. The first vertex must equal the last vertex for field closure. Vertices must be defined in a clockwise order. |
| NPTS | I | | IN | Number of vertices (including closure). |
| YLINE | I | | IN | Scan line number. |
| FL | I | 8 | OUT | Array containing the ordered pixel intercepts. |
| NSAMP | I | | OUT | Total number of pixels in the field on scan line. |
| JJ | I | | OUT | The length of the array FL. |

```
FØRTRAN IV-PLUS V02-04          14:23:13    31-AUG-77          PAGE 4
FLODØT.FTN        /TR:BLØCKS/WR
              SUBRØUTINE FDLINT(FIELD,NPTS,FL,YLINE,NSAMP,JJ)
        C
        C     THIS SUBRØUTINE WILL RETURN THE PIXEL NUMBERS ØF THØSE
        C     PIXELS ØN A A GIVEN LINE THAT ARE CØNTAINED WITHIN THE
        C     BØUNDARIES ØF A  NØN-RECTANGULAR FIELD
        C
        C
        C     INPUT     FIELD -  NØN-RECTANGULAR FIELD TABLE
        C                        ALL THE VERTICES MUST BE IN CLØCKWISE
        C                        ØRDER AND THE LAST VERTEX HAS TØ BE EQUAL
        C                        TØ THE FIRST VERTEX FØR FIELD CLØSURE
        C                        THE FIRST VERTEX MUST HAVE MINIMUM
        C                        PIXEL VALUE
        C               NPTS   - NØ ØF PØINTS ØF THE N-R FIELD
        C               YLINE  - SCAN LINE NUMBER
        C
        C     ØUTPUT    FL     - ARRAY CØNTAINING THE ØRDERED PIXEL INTERCEPTS
        C               NSAMP  - NØ ØF SAMPLES CØNTAINED IN THE FIELD ØF
        C                        A GIVEN SCAN LINE
        C               JJ     - THE LENGTH ØF THE ARRAY FL
        C
              PARAMETER MAXI=6
              DIMENSIØN FIELD(2,NPTS),FL(MAXI)
              INTEGER X1,X2,Y1,Y2,XX,FL,FIELD,YLINE
              INTEGER XNM1,YNM1,XNP2,YNP2
              IF(NPTS.EQ.2)GØ TØ 35
        C*    ØNE VERTEX FIELD
              L= YLINE
              DØ 7 N = 1,MAXI
            7 FL(N) = 0
              NPTSE = NPTS-1
              I = 1
              JJ = 0
          100 X1=FIELD(1,I)
              Y1=FIELD(2,I)
              J = I+1
              X2=FIELD(1,J)
              Y2=FIELD(2,J)
              IF ( I .EQ. 1 ) GØ TØ 200
              IM1 = I-1
              XNM1=FIELD(1,IM1)
              YNM1=FIELD(2,IM1)
              GØ TØ 300
          200 XNM1=FIELD(1,NPTSE)
              YNM1=FIELD(2,NPTSE)
          300 IP1 = I+1
              XNP1=FIELD(1,IP1)
              YNP1=FIELD(2,IP1)
              IF ( I .EQ. NPTSE) GØ TØ 400
              IP2 = I+2
              XNP2=FIELD(1,IP2)
              YNP2=FIELD(2,IP2)
              GØ TØ 500
          400 XNP2=FIELD(1,2)
              YNP2=FIELD(2,2)
          500 IF ( Y1 .EQ. Y2 ) GØ TØ 1000
```

~~115~~
89

```
              IF((L.EQ.Y2).AND.(Y2.EQ.YNP2)) GØ TØ 2000
              IF((L.EQ.Y1).AND.(Y1.EQ.YNM1)) GØ TØ 2000
              RL = L
                RX1 = X1
                RX2 = X2
                RY1 = Y1
                RY2 = Y2
                RXX =(((RL-RY1)*(RX2-RX1))/(RY2-RY1))+RX1
                XX = RXX+.5
              IF(Y1.LT.Y2)  GØ TØ 510
              XX=RXX
              IF((RXX-XX).GT..5) XX=XX+1
        510   CONTINUE
              IF ((XX.GE. X1) .AND. (XX .LE. X2) ) GØ TØ 600
              IF ((XX.LE. X1) .AND. (XX .GE. X2) ) GØ TØ 600
       2000   I = I+1
              IF ( I .GT. NPTSE ) GØ TØ 5
              GØ TØ 100
        600   IF(L.LE.Y1.AND.L.GE.Y2) GØ TØ 700
              IF(L.LE.Y2.AND.L.GE.Y1) GØ TØ 700
              GØ TØ 2000
        700   JJ = JJ+1
              FL(JJ) = XX
              IF ( JJ .EQ. 1 ) GØ TØ 2000
              IF ( I .NE. NPTSE ) GØ TØ 3000
              IF(L.NE.Y2) GØ TØ 3000
              XNM1=X1
              YNM1=Y1
              X1=X2
              Y1=Y2
              X2=FIELD(1,2)
              Y2=FIELD(2,2)
              GØ TØ 3001
       3000   IF ( L .NE. Y1 ) GØ TØ 2000
       3001     IF ((Y1.LT. YNM1) .AND. (Y1 .GT. Y2 )) GØ TØ 4000
              IF ((Y1 .GT. YNM1) .AND. (Y1 .LT. Y2)) GØ TØ 4000
              GØ TØ 2000
       4000   FL(JJ) = 0
              JJ = JJ-1
              GØ TØ 2000
       1000   IF(L.NE.Y1) GØ TØ 2000
              IF(X1.GT.X2) GØ TØ 5000
              IF(YNM1.LT.Y1) GØ TØ 6000
              IF ( YNP2 .GT. Y2 ) GØ TØ 7000
              JJ = JJ+1
              FL(JJ) = X1
              GØ TØ 2000
       7000   JJ = JJ+1
              FL(JJ) = X1
              MM = JJ+1
              FL(MM) = X2
              JJ = MM
              GØ TØ 2000
       6000   IF ( YNP2 .LT. Y2 ) GØ TØ 2000
              JJ = JJ+1
              FL(JJ) = X2
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
          GO TO 2000
     5000 IF ( YNM1 .LT. Y1 ) GO TO 9000
          IF ( YNP2 .GT. Y2 ) GO TO 2000
          JJ = JJ+1
          FL(JJ) = X2
          IF(NPTSL.EQ.2)FL(JJ)=X1
          GO TO 2000
     9000 IF ( YNP2 .GT. Y2 ) GO TO 8000
          JJ = JJ+1
          FL(JJ) = X1
          MM = JJ+1
          FL(MM) = X2
          JJ = MM
          GO TO 2000
     8000 JJ = JJ+1
          FL(JJ) = X1
          GO TO 2000
        5 NPTS1 = JJ-1
          DO 29 NI = 1,NPTS1
          NP1 = NI+1
          DO 29 NJ = NP1,JJ
          IF ( FL(NI) - FL(NJ) ) 29,29,28
       28 NTEMP = FL(NI)
          FL(NI) = FL(NJ)
          FL(NJ) = NTEMP
       29 CONTINUE
          NSAMP = 0
          DO 30 N = 1,JJ,2
          NN = N+1
          NSAMP = NSAMP+(FL(NN) -FL(N)+1)
       30 CONTINUE
          RETURN
       35 IF(YLINE.NE.FIELD(2,1))RETURN
          FL(1)=FIELD(1,1)
          FL(2)=FIELD(1,1)
          NSAMP=1
          JJ=2
          RETURN
          END
```

## 12. [300,6] FSTVID.MAC

This program is documented in reference 1 and in appendix A. It has the following entry points FVOPEN, FVDSET, FVREAD, FVWRIT, FVCLOS, FVDLTE, FVWAIT, FVRWND.

```
HFSTVID -- FAST VIDEØ SUPPØRT SU          MACRØ D10    31-AUG-77 14:23
TABLE ØF CØNTENTS

     1-  2          INTRØDUCTIØN
     1- 44          DATA AREA
     1-169          FVØPEN - ASSIGNS LUN AND ØPENS FILE.
     1-353          FVDSET -- SETS PØINTER TØ NEXT RECØRD.
     1-367          FVDLST  -- GET VBN ØF LAST RECØRD
     1-383          FVDGET  -- RETURNS PØINTER TØ NEXT RECØRD
     1-402          FVNXRC  -- CALCULATE NEW VALUE ØF VBN
     1-429          FVREAD/FVWRIT -- READS/WRITES BLØCKS FRØM/TØ DISK ØR TAPE.
     1-675          FVCLØS -- CLØSES FILE ØN DISK ØR TAPE.
     1-790          FVDLTE -- DELETES AN ØPEN DISK FILE.
     1-841          FVWAIT -- WAITS FØR CØMPLETIØN ØF I/Ø.
     1-868          FVRWND -- REWINDS TAPE ØR PØINTS TØ START ØF DISK FILE.
     1-905          PRSFNM -- PARSES FILE NAME AND STØRES IT IN DATASET DESCRIPTØR.
```

~~12-2~~
93

C-2

```
        .TITLE  FSTVID -- FAST VIDEØ SUPPØRT SUBRØUTINES.
        .SBTTL  INTRØDUCTIØN
;
;       THIS PACKAGE ØF SUBRØUTINES PRØVIDES FAST VIDEØ I/Ø FUNCTIØNS
;       USING MAG TAPE ØR DISK STØRAGE MEDIUM.
;
;       THE FØLLØWING ENTRY PØINTS ARE PRØVIDED:
;
;               FVØPEN - ASSIGNS A LUN TØ THE DESIGNATED DEVICE AND,
;                        IF NØT MAG TAPE, ØPENS THE DESIGNATED FILE ØN
;                        IT. THE NUMBER ØF THE NEXT BLØCK TØ BE WRITTEN
;                        IS SET TØ 1.
;
;               FVDSET - SETS THE NUMBER ØF THE NEXT VIRTUAL BLØCK
;                        (DISK ØNLY) TØ READ/WRITTEN. (ALLØWS AN ØFFSET
;                        FØR USER-WRITTEN LABEL BLØCKS.)
;
;               FVREAD - READS THE DESIGNATED NUMBER ØF BYTES BEGINNING
;                        WITH THE NEXT VIRTUAL BLØCK.
;
;               FVWRIT - WRITES THE DESIGNATED NUMBER ØF BYTES BEGINNING
;                        WITH THE NEXT VIRTUAL BLØCK.
;
;                        (FVREAD AND FVWRIT BØTH INCREMENT THE "NEXT VIRTUAL
;                        BLØCK NUMBER BY THE APPRØPRIATE AMØUNT, DEPENDING ØN
;                        THE NUMBER ØF BYTES TRANSFERRED.)
;
;               FVCLØS - IF THE DESIGNATED LUN IS ASSIGNED TØ MAG TAPE, AN
;                        END ØF FILE IS WRITTEN (IF ØPENED FØR ØUTPUT).
;                        IF THE LUN IS ASSIGNED TØ DISK, THE FILE IS CLØSED.
;
;               FVDLTE - IF THE DESIGNATED LUN INDICATES AN ØPEN DISK FILE,
;                        THE FILE IS DELETED. ØTHERWISE, NØ ACTIØN ØCCURS.
;
;               FVWAIT - WAITS FØR CØMPLETIØN ØF THE LAST I/Ø ØPERATIØN ØN THE
;                        SPECIFIED LUN.
;
;               FVRWND - IF THE SPECIFIED LUN DESIGNATES MAG TAPE, THE TAPE
;                        IS REWØUND. IF THE LUN DESIGNATES DISK, THE VIRTUAL
;                        BLØCK NUMBER ØF THE NEXT RECØRD IS SET TØ 1.
;
;                      JØHN T. DALTØN  CØDE 933        24 FEB 1976
;
        .SBTTL  DATA AREA
                .MCALL  QIØSYS,FDØFSL,QIØ$,DIR$,ASTX$S,ALUN$,ALUN$S
                .MCALL  FDØDF$,FDAT$A,FDRC$A,FDBK$A,FDØP$A,FSRSZ$
                .MCALL  NMBLK$,WTSE$,MØUT$,ØPEN$,CLØSE$
                .IDENT  /X01/
                QIØSYS                  ; DEFINE QIØ SYMBØLS
                FDØFSL                  ; DEFINE FDB ØFFSETS LØCALLY
NLUNSC: .WØRD   NLUNS
NLUNS=20                        ; DEFINE NUMBER ØF LUNS ALLØWED.
NFDBS=10.                       ; DEFINE NUMBER ØF FDB'S ALLØCATED (= MAXIMUM
                                ; NUMBER ØF FILES ØPEN AT ØNE TIME).
;
DSDESC: .WØRD   0,0,0,0,0,0     ; DATA SET DESCRIPTØR.
DFNAM:  NMBLK$  GØØDSTUFF,LEC           ; DEFAULT FILE NAME BLØCK
;       ARGUMENT LIST FØR PRSFNM SUBRØUTINE.
PRSFNA: .WØRD   5,0,0,DSDESC,IØSTAT,0
;       THE FØLLØWING TABLE CØNSISTS ØF FLAG BYTES (ØNE BYTE PER LUN)
;       AND THEIR BIT ASSIGNMENTS.
;
FVBT.Ø=200                      ; BIT 7 = 1 IF FILE IS ØPEN.
FVBT.A=001                      ; BIT 0 = 1 FØR WRITE, 0 FØR READ.
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
FVBT.D=002                  ; BIT 1 = 1 FØR MAG TAPE, 0 FØR DISK.
FVBT.B=004                  ; BIT 2 = 1 IF FILE HAS BEEN ACCESSED.
;
;   DEFINE FLAG BYTE FØR EACH LUN.
FVBT:    .REPT    NLUNS
         .BYTE    0
         .ENDM
;
ALUN:    ALUN$    1,SY,0              ; DPB TØ ASSIGN LUN
ERRARG:  .WØRD    1,0                 ; ARGUMENT LIST FØR ERRØR SUBRØUTINE
WAITAR:  .WØRD    0,0                 ; ARGUMENT LIST FØR FVWAIT
QIØ:     QIØ$     0,0,0,0,IØSTAT,QIØAST,<0,0,0,0,0,0>     ; DPB FØR QIØ
FUNC:    .WØRD    0                   ; TEMP STØRAGE FØR I/Ø FUNCTIØN CØDE
NBYTES:  .WØRD    0                   ; TEMP STØRAGE FØR TRANSFER LENGTH
WTSE:    WTSE$    0                   ; DPB TØ WAIT FØR EVENT FLAG
WTSE30:  WTSE$    30.                 ; DPB TØ WAIT FØR MØ CØMPLETIØN
MØUT:    MØUT$    STR1,PARAM,0,CØNT,SYLØG,0,0,IØSTAT,0
PARAM:   .WØRD    0,0                 ; PARAMETER LIST FØR MØ MESSAGES.
;   MESSAGE ØUPUT FØRMAT STRING DESCRIPTØR PØINTERS.
STR1:    .WØRD    STRS2-STRS1
         .WØRD    STRS1
STR2:    .WØRD    STRS3-STRS2
         .WØRD    STRS2
STR3:    .WØRD    STRS4-STRS3
         .WØRD    STRS3
STR4:    .WØRD    STRSE-STRS4
         .WØRD    STRS4
STRS1:   .ASCIZ   /FVØPEN - FILENAME SYNTAX ERRØR: %VA/
STRS2:   .ASCIZ   /FVØPEN - FILE ALREADY ØPEN FØR LUN %1D/
STRS3:   .ASCIZ   /FVØPEN - ALL %1D FDB'S ARE IN USE, LUN=%1D/
STRS4:   .ASCIZ   /FVØPEN - LUN %1D IS TØØ LARGE, %1D SUPPØRTED./
STRSE:
;
;   GENERATE I/Ø STATUS BLØCK FØR EACH LUN + ØNE EXTRA (FIRST ØNE) FØR
;   GENERAL USE.
IØSTAT:  .REPT    NLUNS+1
         .WØRD    0,0
         .ENDM
;
;   GENERATE TABLE CØNTAINING VIRTUAL BLØCK NUMBER ØF NEXT ACCESS FØR
;   EACH LUN.
NXTREC:  .REPT    NLUNS
         .WØRD    0,1
         .ENDM
;        GENERATE TABLE CØNTAINING VIRTUAL BLØCK NUMBER (VBN) ØF NEXT
;        ACCESS FØR EACH LUN.
;
LSTREC:  .REPT    NLUNS
         .WØRD    0,0
         .ENDM
;
;
;
         FSRSZ$   0                   ; NØ RECØRD I/Ø ØPERATIØNS.
;
;        FDB ALLØCATIØN TABLES.
;
;        LUNFDB IS AN INDEX TABLE (ØNE BYTE PER LUN). THE LUN IS USED TØ INDEX
;        THE TABLE. THE CØRRESPØNDING ENTRY CØNTAINS THE INDEX IN THE
;        FDBFLG AND FDBLST TABLES ØF THE FDB IN USE FØR THAT LUN. (THE WØRD
;        CØNTAINS -1 IF A FILE IS NØT CURRENTLY ØPENED FØR THAT LUN.)
;
;        FDBFLG IS A TABLE CØNTAINING ØNE BYTE FØR EACH FDB FØR WHICH STØRAGE
;        HAS BEEN RESERVED. EACH BYTE CØRRESPØNDS TØ AN FDB ADDRESS IN
;        FDBLST. EACH BYTE CØNTAINS 1 IF THE CØRRESPØNDING FDB IS IN
;        USE AND 0 ØTHERWISE.
```

ORIGINAL PAGE IS OF POOR QUALITY

```
;
;       FDBLST IS A TABLE ØF ØNE-WØRD FDB ADDRESSES CØRRESPØNDING TØ THE
;       FDB FLAG ENTRIES IN FDBFLG.
;
LUNFDB: .REPT   NLUNS
        .BYTE   -1
        .ENDM
FDBFLG: .REPT   NFDBS
        .BYTE   0
        .ENDM
;               DEFINE FDBLST ENTRY
        .MACRØ  FDBLF A
        .WØRD   FDB'A
        .ENDM
NN=1
FDBLST: .REPT   NFDBS
        FDBLF   \NN
NN=NN+1
        .ENDM
;               MACRØ TØ RESERVE AN FDB.
        .MACRØ  FDBDEF A
FDB'A:  FDBDF$                  ; ALLØCATE SPACE FØR FDB
        FDAT$A  R,FIX,FD.BLK,512.,0,12.         ; FILE ATTRIBUTE SECT.
        FDRC$A  FD.RWM,0,0      ;RECØRD ACCESS SECTIØN
        FDBK$A  0,512.,0,,,IØSTAT       ;BLØCK ACCESS SECTIØN
        FDØP$A  ,DSDESC,DFNAM   ;FILE ØPEN SECTIØN
        .ENDM
NN=1
        .REPT   NFDBS
        FDBDEF  \NN
NN=NN+1
        .ENDM
;
;       DEFINE TABLE ØF EVENT FLAGS FØR LUNS.
EFNTAB: .REPT   NLUNS
        .BYTE   0
        .ENDM
;
.SBTTL  FVØPEN - ASSIGNS LUN AND ØPENS FILE.
;
;       ENTRY:   CALL FVØPEN(ITYPE,LUN,FILE,NC,STAT,EF [,NBLKS] )
;
;       WHERE ITYPE = TYPE ØF ACCESS (1=READ,2=WRITE,3=MØDIFY,4=UPDATE,
;                       5=WRITE WITHØUT CREATING NEW VERSIØN IF FILE
;                         ALREADY EXISTS)
;             LUN   = LØGICAL UNIT NUMBER TØ ASSIGN TØ FILE
;             FILE  = ASCII DEVICE/FILENAME
;             NC    = NUMBER ØF CHARACTERS IN 'FILE'
;             STAT  = WØRD IN WHICH STATUS IS RETURNED FRØM THE FDB
;                       (ZERØ IF NØ ERRØR).
;             EF    = EVENT FLAG NUMBER TØ BE USED FØR SYNCHRØNIZATIØN
;                     ØF I/Ø (MUST BE UNIQUE FØR LUN).
;             NBLKS = NUMBER ØF 512-BYTE BLØCKS TØ ALLØCATE (REQUIRED ØNLY IF
;                     'ITYPE' = 2 AND DEVICE IS DISK).
;
;       THIS SUBRØUTINE CALLS 'PRSFNM' TØ PASS THE FILE NAME AND
;       CØNSTRUCT THE DATASET DESCRIPTØR. THE LUN IS ASSIGNED AND,
;       IF THE DEVICE IS DISK, THE FILE IS ØPENED. IF ØPENING A DISK
;       FILE FØR ØUTPUT, THE SUBRØUTINE ATTEMPTS TØ ALLØCATE CØNTIGUØUS
;       SPACE. IF THIS FAILS, IT ATTEMPTS TØ ALLØCATE NØN-CØNTIGUØUS
;       SPACE.
;
;
;       THE FØLLØWING ERRØR CØDES ARE RETURNED IN 'STAT'.
;
PRSERC==1.      ; FILENAME SYNTAX ERRØR.
```

```
FILØPC=-2.        ; FILE IS ALREADY ØPEN FØR LUN.
NØFDBC=-3.        ; ALL FDB'S ARE IN USE.
BIGLUC=-4.        ; LUN IS TØØ LARGE.
ØPERRC=-5.        ; ERRØR RETURN FRØM ØPEN.
;
;
FVØPEN::
;       TEST FØR FILE ALREADY ØPEN.
        MØV     @4(R5),R0          ; GET LUN
        CMP     R0,NLUNSC          ; TEST FØR GREATER THAN MAX ALLØWED.
        BLE     5$
        JMP     BIGLUN             ; LUN TØØ LARGE.
5$:     BITB    #FVBT,0,FVBT-1(R0)           ; ØPEN?
        BEQ     10$                          ; BRANCH IF NØ.
        JMP     FILØPN                       ; FILE ALREADY ØPEN
10$:    MØV     4(R5),PRSFNA+10.             ; SET UP ARGUMENT LIST FØR
        MØV     6(R5),PRSFNA+2               ;  PRSFNM.
        MØV     8.(R5),PRSFNA+4
        MØV     R5,-(SP)           ;SAVE R5 ØN STACK
        MØV     #PRSFNA,R5         ;CALL PRSFNM TØ PARSE FILE NAME.
        JSR     PC,PRSFNM
        MØV     (SP)+,R5           ;RESTØRE R5.
        TST     IØSTAT             ;CHECK FØR SYNTAX ERRØR.
        BGE     20$                ;BRANCH IF NØNE.
        JMP     PRSERR
20$:
30$:    CMP     @DSDESC+2,#"MT               ; IS DEVICE MAG TAPE?
        BNE     40$
        JMP     FVBTST
40$:    CMP     @DSDESC+2,#"MM
        BNE     DISK
        JMP     FVBTST
;
;       DISK DEVICE ØR NØ DEVICE SPECIFIED.
;
DISK:   MØV     @4(R5),R1          ;GET LUN
;       SEARCH LIST FØR AVAILABLE FDB.
        CLR     R4                 ;LØØP INDEX
20$:    TSTB    FDBFLG(R4)         ;TEST FDB FLAG
        BEQ     30$                ;IF 0, FDB IS AVAILABLE.
        INC     R4                 ; NØT AVAILABLE - TEST NEXT ØNE.
        CMP     R4,#NFDBS          ; HAVE ALL FDB'S BEEN TESTED?
        BLT     20$                ; IF NØT, REPEAT.
        JMP     NØFDBS             ; NØ FDB AVAILABLE
;       FDB FØUND - R4 CØNTAINS INDEX.
30$:    INCB    FDBFLG(R4)         ;INDICATE FDB IN USE.
        MØV     @4(R5),R1          ;GET LUN
        MØVB    R4,LUNFDB-1(R1)    ;SET INDEX ØF FDB FØR LUN
        ASL     R4                 ; GET WØRD INDEX ØF FDB
        MØV     FDBLST(R4),R0      ;ADDRESS ØF FDB.
;       SET UP REMAINING FDB PARAMETERS.
        MØVB    R1,F.LUN(R0)       ; MØVE LUN
        MØVB    #FØ.RD,F.FACC(R0)            ;SET UP FØR READ,WRITE, ØR MØDIFY
        CMP     @2(R5),#3          ; ARE WE ØPENING TØ MØDIFY?
        BNE     40$                ; IF NØT, TEST FØR MØDIFY.
        MØVB    #FØ.MFY,F.FACC(R0)           ; YES. MØVE MØDIFY CØDE TØ FDB
        BR      95$                ; ØPEN FILE
40$:    CMP     @2(R5),#4          ; ARE WE ØPENING TØ UPDATE?
        BNE     50$                ; IF NØT, TEST FØR WRITE.
        MØVB    #FØ.UPD,F.FACC(R0)           ; YES. MØVE UPDATE CØDE TØ FDB.
        BR      95$                ; ØPEN FILE.
50$:    CMP     @2(R5),#2          ; ARE WE ØPENING TØ WRITE?
        BNE     60$
55$:    MØVB    #FØ.WRT,F.FACC(R0)           ; MØVE WRITE CØDE TØ FDB
        BR      80$
60$:    CMP     @2(R5),#5                    ; UPDATE ØR WRITE A NEW FILE
```

```
        BNE     95$
        MØVB    #FO.UPD,F.FACC(R0)      ; YES, MØVE CØDE TØ FDB.
        BR      95$                     ; ØPEN FILE.
80$:    CMPB    (R5),#7                 ;TEST FØR NBLKS ARGUMENT
        BNE     95$
;
;       'NBLKS' SPECIFIED AND ØPENING FØR ØUTPUT, ATTEMPT TØ ALLØCATE
;       CØNTIGUØUS FILE.
;
        MØV     @14.(R5),F.CNTG(R0)
        ØPEN$   R0
;       TEST FØR ERRØR. IF NØNE, RETURN AND INDICATE SUCCESS.
        TSTB    F.ERR(R0)
        BGE     DØPND
        CMP     @2(R5),#5               ; IF ERRØR CAUSED BY ITYPE=5 AND
        BNE     90$                     ; FILE ALREADY IN EXISTENCE,
        CMPB    #IE.DUP,F.ERR(R0)       ; IGNØRE.
        BEQ     DØPND
;       ERRØR. ATTEMPT TØ ALLØCATE NØN-CØNTIGUØUS FILE.
90$:    MØV     @14.(R5),F.CNTG(R0)     ; RESTØRE # BLØCKS
        NEG     F.CNTG(R0)              ;   AND NEGATE FØR NØN-CØNTIG
95$:    ØPEN$   R0
        TSTB    F.ERR(R0)       ; TEST FØR ERRØR
        BGE     DØPND
        CMP     @2(R5),#5               ; IF ERRØR CAUSED BY ITYPE=5 AND
        BNE     100$                    ; FILE DØES NØT EXIST,
        CMPB    #IE.NSF,F.ERR(R0)       ; ØPEN FILE FØR WRITE
        BEQ     55$
100$:   MØV     #ØPERRC,@10.(R5)        ;INDICATE ØPEN ERRØR
        JSR     PC,FCSERR       ; CALL ERRØR MESSAGE SUBRØUTINE
        RTS     PC              ; RETURN
;
;       MAG TAPE - SET FLAG BYTE BITS AND RETURN.
FVBTST: MØV     @4(R5),R1       ; GET LUN
        BISB    #FVBT.D,FVBT-1(R1)      ; INDICATE MAG TAPE
;
;       FILE SUCCESSFULLY ØPENED.
DØPND:  BISB    #FVBT.Ø,FVBT-1(R1)      ;SET BITS IN FVBT
        CMP     @2(R5),#2
        BNE     RET
        BISB    #FVBT.A,FVBT-1(R1)      ; INDICATE ØUTPUT
RET:    CLR     @10.(R5)                ; INDICATE SUCCESS.
        ASH     #2,R1           ; GET DØUBLE-WØRD INDEX
        CLR     LSTREC-2(R1)
        CLR     LSTREC-4(R1)
        CLR     NXTREC-4(R1)
        MØV     #1,NXTREC-2(R1)         ; SET VIRTUAL BLØCK # ØF 1ST REC
        RØR     R1              ; GET LUN
        RØR     R1
        MØVB    @12.(R5),EFNTAB-1(R1)   ; MØVE EVENT FLAG NUMBER TØ TABLE
        RTS     PC              ;RETURN
;
;       FVØPEN - ERRØR HANDLING.
;
;       SYNTAX ERRØR IN FILENAME FRØM PRSFNM
;
PRSERR: MØV     @8.(R5),PARAM   ; NUMBER ØF CHARACTERS
        MØV     6(R5),PARAM+2   ; @(FILENAME)
        MØV     #STR1,MØUT+M.ØSTR       ;FØRMAT STRING ADDRESS
        MØV     #PRSERC,@10.(R5)        ;SET ERRØR CØDE.
        JMP     MØPRT           ; PRINT MESSAGE
;
;       FILE ALREADY ØPEN FØR LUN,
;
FILØPN: MØV     @4(R5),PARAM    ;LUN
        MØV     #STR2,MØUT+M.ØSTR       ;FØRMAT STRING ADDRESS
```

~~12T~~
98

```
        MØV     #FILØPC,@10,(R5)         ;SET ERRØR CØDE
        JMP     MØPRT                ;PRINT MESSAGE
;
;       ALL FDB'S IN USE.
;
NØFDBS: MØV     #NFDBS,PARAM             ; NUMBER ØF FDB'S ALLØWED.
        MØV     @4(R5),PARAM+2           ; LUN
        MØV     #STR3,MØUT+M,ØSTR
        MØV     #NØFDBC,@10,(R5)         ; SET ERRØR CØDE.
        JMP     MØPRT
;
;
;       LUN TØØ LARGE.
;
BIGLUN: MØV     @4(R5),PARAM     ;LUN
        MØV     #NLUNS,PARAM+2   ;MAX LUN
        MØV     #STR4,MØUT+M,ØSTR        ;FØRMAT STRING ADDRESS
        MØV     #BIGLUC,@10,(R5)         ;SET ERRØR CØDE.
;       PRINT MESSAGE AND WAIT.
MØPRT:  MØV     ,MØLUN,MØUT+M,ØLUN       ; MØVE LUN TØ DPB
        DIR$    #MØUT,DIRERR
        DIR$    #WTSE$0,DIRERR
        RTS     PC                       ; RETURN.
;
.SBTTL  FVDSET -- SETS PØINTER TØ NEXT RECØRD.
;
;               ENTRY: CALL FVDSET(LUN,LSW,MSW)
;
;    THE VIRTUAL BLØCK NUMBER ØF THE NEXT RECØRD TØ BE ACCESSED
;    ØN "LUN" IS SET TØ "LSW/MSW".
;
FVDSET::
        MØV     @2(R5),R1        ; LUN
        ASH     #2,R1            ; GET DØUBLE-WØRD INDEX
        MØV     @6(R5),NXTREC-4(R1)
        MØV     @4(R5),NXTREC-2(R1)      ; SET PØINTER.
        RTS     PC
;
.SBTTL  FVDLST  -- GET VBN ØF LAST RECØRD
;
;
;       ENTRY:  CALL FVDLST(LUN,LSW,MSW)
;
;               GETS STARTING VBN ØF LAST RECØRD WRITTEN.
;
;
FVDLST::
        MØV     @2(R5),R1
        ASH     #2,R1
        MØV     LSTREC-2(R1),@4(R5)      ;GET
        MØV     LSTREC-4(R1),@6(R5)      ;VBN
        RTS     PC
;
;
.SBTTL  FVDGET  -- RETURNS PØINTER TØ NEXT RECØRD
;
;       ENTRY:  CALL FVDGET(LUN,LSW,MSW)
;
;               WHERE   LUN=LØGICAL UNIT NUMBER
;                       LSW=LEAST SIG. PART ØF VBN PØINTER
;                       MSW=MØST SIG. PART ØF VBN PØINTER
;
;               LSW/MSW IS SET TØ VBN ØF STARTING BLØCK ØF NEXT RECØRD.
;
;
FVDGET::
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        MØV     @2(R5),R1
        ASH     #2,R1   ;GET TABLE PØINTER
        MØV     NXTREC-2(R1),@4(R5)     ;GET LSW
        MØV     NXTREC-4(R1),@6(R5)     ;GET MSW
        RTS     PC
;
;
.SBTTL  FVNXRC  -- CALCULATE NEW VALUE ØF VBN
;
;
;       ENTRY:  CALL FVNXRC(LUN,NBYT,LSW,MSW)
;
;               WHERE NBYT = NUMBER ØF BYTES THAT ARE ABØUT TØ BE WRITTEN,
;
;               RETURNS VBN ØF RECØRDFØLLØWING ØNE THAT IS ABØUT TØ
;               BE WRITTEN.
FVNXRC::
        MØV     @2(R5),R1       ;LUN
        ASH     #2,R1   ;PØINTER
        MØV     NXTREC-2(R1),R3 ;CURRENT VBN
        MØV     NXTREC-4(R1),R2
        MØV     @4(R5),R0       ;GET BYTE CØUNT
        ADD     #511.,R0        ;CØNVERT
        RØR     R0              ;TØ
        CLRB    R0              ;BLØCK
        SWAB    R0              ;CØUNT
        ADD     R0,R3           ;ADD TØ
        ADC     R2              ;VBN
        MØV     R3,@6(R5)       ;RETURN
        MØV     R2,@10(R5)      ;NEXT PØINTER
        RTS     PC
;
;
;
.SBTTL  FVREAD/FVWRIT -- READS/WRITES BLØCKS FRØM/TØ DISK ØR TAPE.
;
;               ENTRY: CALL FVREAD(LUN,BUFFER,NBYTES)
;                      CALL FVWRIT(LUN,BUFFER,NBYTES)
;
;   "NBYTES" BYTES ARE READ INTØ (WRITTEN FRØM) ARRAY BUFFER FRØM
;   (TØ) THE DEVICE ASSØCIATED WITH LØGICAL UNIT NUMBER "LUN".
;   FØR DISK, THE READ (WRITE) IS ISSUED STARTING AT THE VIRTUAL
;   BLØCK NUMBER CØNTAINED IN THE "NXTREC" TABLE (SEE "FVDSET").
;   THE "NXTREC" ENTRY IS THEN INCREMENTED BY THE NUMBER ØF 512-BYTE
;   BLØCKS REQUIRED TØ CØNTAIN "NBYTES" BYTES.
;
FVREAD::
        MØV     #IØ.RVB,FUNC                    ; SET UP I/Ø FUNCTIØN CØDE.
        BR      FVRWCM
FVWRIT::
        MØV     #IØ.WVB,FUNC                    ; SET UP I/Ø FUNCTIØN CØDE.
FVRWCM: MØV     @2(R5),R1                       ; GET LUN
        BITB    #FVMT.D,FVDT-1(R1)              ; MAG TAPE?
        BEQ     10$                             ; BRANCH IF NØT.
        JMP     FVTAPE                          ; MAG TAPE...
;
;   DISK DEVICE, CHECK FØR FILE ØPEN.
;
10$:    BITB    #FVBT.D,FVDT-1(R1)
        BNE     20$
;   NØ FILE ØPEN, FØRCE "NØ FILE ACCESSED ØN LUN" MESSAGE.
        MØV     #IE.NLN,IØSTAT                  ; MØVE ERRØR CØDE TØ STATUS BLK.
        MØV     R5,-(SP)                        ; SAVE R5.
        MØV     #IØSTAT,ERRARG+2                ; SET UP ARG LIST.
        MØV     #ERRARG,R5
        JSR     PC,IØERR                        ; PRINT ERRØR MESSAGE.
        MØV     (SP)+,R5                ; RESTØRE R5.
        RTS     PC                      ; RETURN - NØ ØPERATIØN PERFØRMED
;
;   FILE ØPEN ØN DISK, WAIT FØR ANY PREVIØUS I/Ø CØMPLETIØN.
;
2$:     MØV     2(R5),WAITAR+2          ; LUN ADDRESS
        MØV     R5,-(SP)                ; SAVE R5
        MØV     #WAITAR,R5
        JSR     PC,FVWAIT               ; WAIT FØR I/Ø CØMPLETIØN
        MØV     (SP)+,R5                ; RESTØRE R5.
;   PERFØRM QIØ - SET UP DPB.
        MØV     FUNC,QIØ+Q.IØFN
        MØV     @2(R5),R1               ; GET LUN
        BISB    #FVBT,R,FVRT-1(R1)      ; INDICATE FILE ACCESSED.
        MØV     R1,QIØ+Q.IØLU           ; LUN TØ DPB
        MØVB    EFNTAB-1(R1),QIØ+Q.IØEF ; EVENT FLAG TØ DPB
        MØV     #IØSTAT,QIØ+Q.IØSB      ; CØMPUTE @(I/Ø STATUS BLØCK)
        ASH     #2,R1
        ADD     R1,QIØ+Q.IØSB
        MØV     4(R5),QIØ+Q.IØPL        ;START ADDRESS.
        MØV     @6(R5),QIØ+Q.IØPL+2     ; LENGTH ØF TRANSFER
        MØV     @6(R5),NBYTES           ; SAVE FØR AST USE
        MØV     NXTREC-4(R1),QIØ+Q.IØPL+6       ; SET UP DØUBLE PRECISIØN
        MØV     NXTREC-2(R1),QIØ+Q.IØPL+10      ; VIRTUAL BLØCK NUMBER.
;
;       CØNVERT (# ØF BYTES TØ BE TRANSFERRED) TØ (# ØF BLØCKS).
;
        MØV     NBYTES,R0               ; # ØF BYTES TØ BE TRANSFERRED
        ADD     #511.,R0                ; RØUND UP BYTE CØUNT TØ NEXT BLØCK
        RØR     R0                      ; RECØVER CARRY
        CLRB    R0                      ; CLEAR FRACTIØNAL PART
        SWAB    R0                      ; CØNVERT TØ BLØCK CØUNT
;
;       CØMPUTE (TØTAL BLØCKS ACCESSED IN FILE) + 1 AFTER THIS ØPERATIØN
;       IN R2,R3 (DØUBLE WØRD INTEGER).
;
        MØV     NXTREC-4(R1),R2         ; GET START VBN (VIRUAL BLØCK NUMBER)
        MØV     NXTREC-2(R1),R3         ; ØF ACCESS IN R2,R3.
        MØV     R2,-(SP)
        MØV     R3,-(SP)
        ADD     R0,R3                   ; ADD # BLØCKS TØ BE TRANSFERRED
        ADC     R2
        MØV     R2,-(SP)                ; SAVE ØN STACK FØR LATER USE.
        MØV     R3,-(SP)
;
;       GET FDB ADDRESS IN R0
;
        MØV     @2(R5),R0               ; GET LUN
        MØVB    LUNFDB-1(R0),R0         ; INDEX ØF FDB FØR LUN
        ASL     R0                      ; CØNVERT TØ WØRD INDEX
        MØV     FDBLST(R0),R0           ; ADDRESS ØF FDB
;
;       CHECK FØR ATTEMPTED ACCESS PAST END ØF FILE.
;
        CMP     #IØ.WVB,FUNC            ; BRANCH IF WRITING.
        BEQ     WRTDSK
;
;       READING FRØM DISK, IF REQUESTED ACCESS IS PAST END-ØF-FILE,
;       READ UP TØ END-ØF-FILE AND RETURN EØF ERRØR CØDE.
;
;       SUBTRACT # BLØCKS WRITTEN + 1 (FRØM FDB) FRØM # BLØCKS
;       ACCESSED AFTER THIS ØPERATIØN + 1 (IN R2,R3).
;
        SUB     F.EFBK+2(R0),R3         ; LØW ØRDER WØRD.
        SBC     R2                      ; SUBTRACT CARRY FRØM HIGH-ØRDER WØRD
        SUB     F.EFBK(R0),R2           ; HIGH-ØRDER WØRD.
        TST     F.FFBY(R0)              ; IF F.FFBY NØT ZERØ, THEN F.EFBK
        BEQ     30$                     ; CØNTAINS THE ACTUAL NUMBER ØF BLØCKS
        SUB     #1,R3                   ; WRITTEN, NEED TØ SUBTRACT ØNE MØRE .
        SBC     R2
30$:
        TST     R3                      ; IF DIFFERENCE > 0, ACCESS
        BGT     RDEØF                   ;   PAST END-ØF-FILE.
;
;       NØRMAL ACCESS, DØ QIØ AND SET NXTREC ENTRY TØ NEW ACCESSED BLØCK CØUNT
;
        DIR$    #QIØ,DIRERR             ;ISSUE QIØ
        MØV     (SP)+,NXTREC-2(R1)      ; PØP BLØCK CØUNT FRØM STACK.
        MØV     (SP)+,NXTREC-4(R1)
        MØV     (SP)+,LSTREC-2(R1)
        MØV     (SP)+,LSTREC-4(R1)
        RTS     PC                      ; RETURN.
;
;       READ PAST END-ØF-FILE ATTEMPTED, CØNVERT # BLØCKS PAST EØF TØ
;       # BYTES AND SUBTRACT FRØM REQUESTED TRANSFER LENGTH IN QIØ DPB.
;
RDEØF:  ASH     #9.,R3                  ; MULTIPLY BY 512. BYTES
        MØV     QIØ+Q.IØSB,R4           ; GET @(I/Ø STATUS BLØCK)
        CLR     2(R4)                   ; CLEAR BYTE CØUNT
        SUB     R3,QIØ+Q.IØPL+2         ; SUBTRACT FRØM LENGTH IN DPB
        BLE     10$                     ; IF NEGATIVE ØR ZERØ, NØ TRANSFER
        DIR$    #QIØ,DIRERR             ; ISSUE QIØ AND
        MØVB    QIØ+Q.IØEF,WTSE+W.TSEF  ;   WAIT FØR CØMPLETIØN.
        DIR$    #WTSE,DIRERR
10$:    MØV     F.EFBK(R0),NXTREC-4(R1) ; SET NXTREC PØINTER TØ PAST EØF.
        MØV     F.EFBK+2(R0),NXTREC-2(R1)
        MØV     #IE.EØF,@QIØ+Q.IØSB     ; SET EØF CØDE IN STATUS BLØCK
        MØV     R5,-(SP)                ; CALL IØERR FØR MESSAGE.
        MØV     QIØ+Q.IØSB,ERRARG+2     ; @(I/Ø STATUS BLØCK) TØ ARGUMENT LIST
        MØV     #ERRARG,R5
        JSR     PC,IØERR
        MØV     (SP)+,R5
        MØV     @2(R5),R1               ; GET LUN
        BICB    #FVBT.R,FVBT-1(R1)      ; INDICATE NØ I/Ø IN PRØGRESS
        TST     (SP)+                   ; CLEAR STACK
        TST     (SP)+
        TST     (SP)+
        TST     (SP)+
        RTS     PC                      ; RETURN
;
;       WRITING TØ DISK. IF REQUESTED ACCESS IS BEYØND ALLØCATED SPACE,
;       ATTEMPT TØ EXTEND FILE.
;
WRTDSK: SUB     F.HIBK+2(R0),R3         ; GET HIGHEST BLØCK TØ BE WRITTEN
        SBC     R2                      ;   - HIGHEST BLØCK ALLØCATED
        SUB     F.HIBK(R0),R2           ;     IN R2,R3.
        SUB     #1,R3
        SBC     R2
        BLT     30$             ; IF NØT > 0, ACCESS IS WITHIN ALLØCATIØN
        TST     R3              ; HIGH-ØRDER WØRD 0, TEST LØW-ØRDER WØRD
        BEQ     30$             ; IF ZERØ, NØ EXTEND.
;
;       WRITE REQUEST IS PAST ALLØCATED SPACE, ATTEMPT TØ EXTEND FILE.
;
        MØV     R3,R4                   ; SAVE # BLØCKS TØ EXTEND
        MØV     R3,R1                   ; SET UP REGS FØR .EXTND
        CMP     #22.,R1                 ; IF < 22. BLØCKS REQUIRED, EXTEND
        BLT     10$                     ;   BY 22.
        MØV     #22.,R1
10$:    MØV     #1,R2                   ; ATTEMPT CØNTIGUØUS EXTEND.
        JSR     PC,.EXTND
        BCC     30$                     ; BRANCH IF SUCCESSFUL
        MØV     #201,R2                         ; ATTEMPT NØN-CONTIGUØUS EXTEND
        JSR     PC,.EXTND
        BCC     30$                             ; BRANCH IF SUCCESSFUL
        CMP     R1,R3                           ; IF REQUEST WAS FØR ACTUAL BLØCKS
        BEQ     20$                             ;   NEEDED, NØ SPACE AVAILABLE.
        MØV     #201,R2                         ; REQUEST WAS FØR 22. BLØCKS, TRY
        MØV     R3,R1                           ;   FØR WHAT WE NEED.
        JSR     PC,.EXTND
        BCC     30$                             ; BRANCH IF SUCCESSFUL.
;
;       UNABLE TØ EXTEND FILE - REPØRT ERRØR.
;
20$:    JSR     PC,FCSERR                       ; CALL FCSERR FØR MESSAGE.
        MØVB    F.ERR(R0),R1                    ; MØVE ERRØR CØDE TØ I/Ø STATUS BLØCK
        MØV     R1,QIØ+G,IØSB
        MØV     @2(R5),R1                       ; GET LUN
        BICB    #FVBT.B,FVBT-1(R1)              ; INDICATE NØ I/Ø IN PRØGRESS.
        TST     (SP)+                           ; CLEAR STACK
        TST     (SP)+
        RTS     PC                              ; RETURN
;
;       EXTEND SUCCESSFUL ØR NØ EXTEND NEEDED.
;
30$:    DIR$    #QIØ,DIRERR                     ; DØ QIØ
        MØV     (SP)+,R3                        ; RESTØRE NEXT VBN.
        MØV     (SP)+,R2
;
;       IF # VIRTUAL BLØCKS WRITTEN > CURRENT VBN COUNT IN FDB, UPDATE
;       FDB VBN COUNT.
;
        CMP     F.EFBK(R0),R2                   ; IF HIGH-ØRDER WØRD ØF FDB VBN <
        BLT     40$                             ;   VBN CØUNT AFTER WRITE,
        BGT     50$
        CMP     F.EFBK+2(R0),R3                 ;   ØR IF = AND LØW-ØRDER WØRD IN
        BGE     50$                             ;     < LØW-ØRDER WØRD ATER WRITE,
40$:    MØV     R2,F.EFBK(R0)                   ;   SET FDB VBN TØ NEW CØUNT.
        MØV     R3,F.EFBK+2(R0)
50$:    MØV     @2(R5),R1                       ; GET LUN
        ASH     #2,R1                           ; GET DØUBLE-WØRD INDEX
        MØV     R2,NXTREC-4(R1)                 ; SET VBN FØR NEXT ACCESS
        MØV     R3,NXTREC-2(R1)
        MØV     (SP)+,LSTREC-2(R1)
        MØV     (SP)+,LSTREC-4(R1)
        RTS     PC                              ; RETURN
;
;   MAG TAPE READ/WRITE - WAIT FØR PREVIØUS I/Ø.
FVTAPE: MØV     2(R5),WAITAR+2                  ; LUN TØ WAIT ARG LIST
        MØV     R5,-(SP)                        ; SAVE R5
        MØV     #WAITAR,R5                      ; CALL FVWAIT(LUN)
        JSR     PC,FVWAIT
        MØV     (SP)+,R5                        ; RESTØRE R5
;   SET UP QIØ DPB.
        MØV     #IØ.RLB,QIØ+Q.IØFN              ; SET UP I/Ø FUNCTIØN CØDE.
        CMP     FUNC,#IØ.WVB
        BNE     10$
        MØV     #IØ.WLB,QIØ+Q.IØFN
10$:    MØV     @2(R5),R1                       ; GET LUN
        BISB    #FVBT.F,FVBT-1(R1)              ; INDICATE FILE HAS BEEN ACCESSED.
        MØV     R1,QIØ+Q.IØLU                   ; LUN TØ DPB
        MØVB    EFNTAB-1(R1),QIØ+Q.IØEF         ; EVENT FLAG TØ DPB
        MØV     #IØSTAT,QIØ+Q.IØSB              ; CØMPUTE @(I/Ø STATUS BLØCK)
        ASH     #2,R1                           ; DØUBLE WØRD ØFFSET.
        ADD     R1,QIØ+Q.IØSB
        ASH     #-2,R1
        MØV     4(R5),QIØ+Q.IØPL                ; START ADDRESS ØF BUFFER
        MØV     @6(R5),QIØ+Q.IØPL+2             ; # BYTES TØ TRANSFER
        DIRS    #QIØ,DIRERR                 ; ISSUE QIØ
        RTS     PC                          ; RETURN
;
;  QIØ AST RØUTINE FØR FVREAD/FVWRIT.
;
QIØAST:
        MØV     (SP)+,ERRARG+2              ; PØP @(I/Ø STATUS BLØCK)
        TSTB    @ERRARG+2                   ; TEST FØR I/Ø ERRØR.
        BGE     10$                         ; EXIT IF NØ ERRØR.
;  I/Ø ERRØR.
        MØV     R5,-(SP)                    ; SAVE R5
        MØV     #ERRARG,R5                  ; CALL I/Ø ERRØR RØUTINE
        JSR     PC,IØERR
        MØV     (SP)+,R5                    ; RESTØRE R5
10$:    ASTX$S                              ; EXIT AST RØUTINE
;
.SBTTL  FVCLØS -- CLØSES FILE ØN DISK ØR TAPE.
;
;                 ENTRY: CALL FVCLØS(LUN [,IRWFLG] )
;
;   WHERE LUN = LØGICAL UNIT #.
;         IRWFLG = ØPTIØNAL REWIND FLAG. IF THIS ARGUMENT IS PRESENT AND
;                  > 0 AND THE SPECIFIED LUN IS ASSIGNED TØ A MAG TAPE
;                  DEVICE, THE TAPE IS REWOUND AFTER THE 'CLØSE' ØPERATIØN
;                  HAS BEEN PERFØRMED.
;
;   THIS SUBRØUTINE CLØSES A FILE ØPENED BY FVØPEN.
;   IF THE LUN IS ASSIGNED TØ A MAG TAPE DEVICE AND THE DATA SET
;   IS ØPENED FØR ØUTPUT, AN END-ØF-FILE IS WRITTEN ØN THE UNIT.
;   IF THE LUN IS ASSIGNED TØ A DISK FILE, THE FILE IS CLØSED AND
;   THE FDB IS FREED.
;
FVCLØS::
        MØV     2(R5),WAITAR+2              ; MØV LUN TØ FVWAIT ARGUMENT LIST
        MØV     R5,-(SP)                    ; SAVE R5
        MØV     #WAITAR,R5
        JSR     PC,FVWAIT                   ; WAIT FØR CØMPLETIØN ØF ANY I/Ø
        MØV     (SP)+,R5                    ; RESTØRE R5
        MØV     @2(R5),R1                   ; GET LUN
        BITB    #FVBT.D,FVBT-1(R1)          ; CHECK IF MAG TAPE
        BNE     5$
        JMP     CLDISK                      ; NØT MAG TAPE
;
;   CLØSING MAG TAPE FILE.
;
5$:     BITB    #FVBT.A,FVBT-1(R1)          ; ØPENED FØR ØUTPUT?
        BNE     50$                         ; BRANCH IF YES.
;
;       READING MAG TAPE. IF THE LAST ØPERATIØN DID NØT ENCØUNTER AN EØF
;       AND THE TAPE IS NØT REWINDING ØR AT THE LØAD PØINT, SPACE PAST
;       THE NEXT EØF.
;
;       READ TAPE CHARACTERISTICS TØ DETERMINE PØSITIØN.
;
        MØV     #IØ.SEC,QIØ+Q.IØFN          ; FUNCTIØN CØDE TØ DPB
        BISB    #FVBT.B,FVBT-1(R1)          ; INDICATE I/Ø IN PRØGRESS
        MØV     R1,QIØ+Q.IØLU               ; LUN TØ DPB
        MØV     EFNTAB-1(R1),QIØ+Q.IØEF     ; EVENT FLAG NUMBER TØ DPB
        MØV     #IØSTAT,QIØ+Q.IØSB          ; I/Ø STATUS BLØCK ADDRESS TØ DPB
        DIRS    #QIØ,DIRERR                 ; ISSUE QIØ
        MØV     R5,-(SP)                    ; WAIT FØR CØMPLETIØN
        MØV     #WAITAR,R5
        JSR     PC,FVWAIT
        MØV     (SP)+,R5
        BIT     #21040,IØSTAT+2             ; TEST CHARACTERISTICS BITS
        BNE     10$
```

```
;
;       IF TAPE IS TØ BE REWØUND AFTER CLØSE, DØ NØT SPACE PAST EØF.
;
        CMPB    (R5),#2                 ; SECØND ARGUMENT PRESENT?
        BNE     7$                      ; BRANCH IF NØT.
        TST     @4(R5)                  ; YES, IS IT > 0?
        BGT     65$                     ; BRANCH IF YES.
;   SPACE PAST NEXT EØF MARK, SET UP QIØ DPB.
7$:     MØV     #IØ.SPF,QIØ+Q.IØFN      ; FUNCTIØN CØDE
        BISB    #FVØT,B,FVØT-1(R1)      ; INDICATE I/Ø IN PRØGRESS.
        MØV     R1,QIØ+Q.IØLU           ; LUN TØ DPB
        MØVB    EFNTAB-1(R1),QIØ+Q.IØEF ; EVENT FLAG NUMBER TØ DPB
        CLR     QIØ+Q.IØSB              ; CLEAR I/Ø STAT BLK ADDRESS
        MØV     #1,QIØ+Q.IØPL           ; MØVE 1 FILE MARK
        DIR$    #QIØ,DIRERR             ; ISSUE QIØ
        MØV     R5,-(SP)                ; WAIT FØR CØMPLETIØN
        MØV     #WAITAR,R5
        JSR     PC,FVWAIT
        MØV     (SP)+,R5                ; RESTØRE R5
10$:    BR      60$                     ; CLEAR FLAGS AND RETURN.
; . WRITING MAG TAPE, WRITE EØF.
50$:    MØV     #IØ.EØF,QIØ+Q.IØFN      ; FUNCTIØN CØDE TØ DPB.
        BISB    #FVØT,B,FVØT-1(R1)      ; INDICATE I/Ø IN PRØGRESS.
        MØV     R1,QIØ+Q.IØLU           ; LUN
        MØVB    EFNTAB-1(R1),QIØ+Q.IØEF ; EVENT FLAG # TØ DPB.
        CLR     QIØ+Q.IØSB              ; NØ I/Ø STATUS BLØCK.
        DIR$    #QIØ,DIRERR             ; ISSUE QIØ
        MØV     R5,-(SP)                ; CALL FVWAIT TØ WAIT FØR CØMPLETIØN
        MØV     #WAITAR,R5
        JSR     PC,FVWAIT
        MØV     (SP)+,R5
;
;       IF SECØND ARGUMENT PRESENT AND > 0, REWIND TAPE.
;
60$:    CMPB    (R5),#2                 ; SECØND ARGUMENT PRESENT.
        BNE     70$                     ; BRANCH IF NØT.
        TST     @4(R5)                  ; PRESENT, > 0?
        BLE     70$                     ; BRANCH IF NØT.
65$:    MØV     #IØ.RWD,QIØ+Q.IØFN      ; REWIND FUNCTIØN CØDE TØ DPB
        MØV     R1,QIØ+Q.IØLU           ; MØVE LUN TØ DPB
        MØV     EFNTAB-1(R1),QIØ+Q.IØEF ; EVENT FLAG NUMBER TØ DPB
        MØV     #IØSTAT,QIØ+Q.IØSB      ; #(I/Ø STATUS BLØCK) TØ DPB
        DIR$    #QIØ,DIRERR             ; ISSUE QIØ TØ REWIND
        MØV     EFNTAB-1(R1),WTSE+W.TSEF ; WAIT FØR CØMPLETIØN.
        DIR$    #WTSE,DIRERR
;   DEASSIGN LUN.
70$:    ALUN$S  R1,,,DIRERR
        BR      CLRFL1                  ; CLEAR FLAGS AND RETURN.
;
;       CLØSING DISK FILE.
;
CLDISK: CMPB    #-1,LUNFDB-1(R1)
        BNE     10$
        JMP     CLRFLG          ; NØ FILE ØPEN FØR LUN.
10$:    MØVB    LUNFDB-1(R1),R2 ; GET FDB INDEX
        ASL     R2              ; GET WØRD INDEX
        MØV     FDBLST(R2),R0   ; #(FDB).
        RØR     R2
        CLØSE$  R0,FCSERR       ; ISSUE CLØSE
; . CLEAR FLAGS FØR LUN AND RETURN.
CLRFLG:
        MØVB    #-1,LUNFDB-1(R1)
        CLRB    FDBFLG(R2)      ; INDICATE FDB NØT IN USE.
CLRFL1: CLRB    FVST-1(R1)              ; CLEAR FLAGS BYTE FØR LUN.
        RTS     PC              ; RETURN.
.SBTTL  FVDLTE -- DELETES AN ØPEN DISK FILE.
```

```
;
;               ENTRY:  CALL FVDLTE(LUN,ISTAT)
;
;       WHERE   LUN = LØGICAL UNIT NUMBER ØF AN ØPEN DISK FILE TØ BE DELETED.
;
;       IF THE SPECIFIED LUN DØES NØT INDICATE AN ØPEN DISK FILE, NØ ACTIØN
;       ØCCURS.
;
;       THE FØLLØWING ERRØR CØDES ARE RETURNED IN 'ISTAT':
;
SUCDEL=0                ; FILE WAS SUCCESSFULLY DELETED.
NDSKØP=-1.              ; LUN DID NØT SPECIFY AN ØPEN DISK FILE.
DELERR=-2.              ; ERRØR RETURN FRØM .DLFNB
;
;
FVDLTE::
        MØV     @2(R5),R1                       ; GET LUN
        CMP     R1,#LUNSC                       ; TEST FØR GREATER THAN MAX ALLØWED
        BGT     30$                             ; BRANCH IF TØØ LARGE.
        MØV     2(R5),WAITAR+2                  ; MØVE LUN TØ FVWAIT ARGUMENT LIST
        MØV     R5,-(SP)
        MØV     #WAITAR,R5                      ; WAIT FØR CØMPLETIØN ØF ANY I/Ø
        JSR     PC,FVWAIT                       ;   ØN THIS LUN.
        MØV     (SP)+,R5
        MØV     @2(R5),R1                       ; LUN
        BITB    #FVBT.D,FVBT-1(R1)              ; CHECK FØR DISK FILE
        BNE     30$                             ; BRANCH IF NØT DISK
        CMPB    #-1,LUNFDB-1(R1)                ; IS FILE ØPEN?
        BEQ     30$                             ; BRANCH IF NØT.
        MØVB    LUNFDB-1(R1),R2                 ; GET FDB INDEX.
        ASL     R2                              ; GET WØRD INDEX
        MØV     FDBLST(R2),R0                   ; ADDRESS ØF FDB
        RØR     R2
        JSR     PC,.DLFNB                       ; DELETE BY FILENAME BLØCK
        BCC     10$                             ; BRANCH IF SUCCESSFUL
        JSR     PC,FCSERR                       ; DELETE FAILED - REPØRT ERRØR.
        MØV     #DELERR,@4(R5)
        RTS     PC                              ; RETURN
;
;       FILE DELETED - CLEAR FLAGS FØR LUN AND RETURN.
;
10$:    MØVB    #-1,LUNFDB-1(R1)                ; INDICATE NØ FDB FØR LUN
        CLRB    FDBFLG(R2)                      ; INDICATE FDB NØT IN USE.
        CLRB    FVBT-1(R1)                      ; CLEAR FLAGS BYTE FØR LUN
        RTS     PC                              ; RETURN
;
;       LUN DID NØT SPECIFY AN ØPEN DISK FILE.
;
30$:    MØV     #NDSKØP,@4(R5)
        RTS     PC                              ; RETURN
.SBTTL  FVWAIT -- WAITS FØR CØMPLETIØN ØF I/Ø.
;
;               ENTRY:  CALL FVWAIT(LUN [,IØST] )
;
;    THIS SUBRØUTINE WAITS FØR CØMPLETIØN ØF ANY ØUTSTANDING I/Ø ØN THE
;    DESIGNATED LUN. THE FLAGS BYTE (FVBT ENTRY) IS CHECKED TØ DETERMINE
;    IF ANY I/Ø IS IN PRØGRESS. IF SØ, THE EVENT FLAG FØR THE LUN IS ØBTAINED
;    FRØM EFNTAB AND A WAIT IS ISSUED.
;
;    IF THE SECØND (ØPTIØNAL) ARGUMENT IS PRØVIDED, THE TWØ-WØRD I/Ø STATUS
;    BLØCK FØR THE LUN (INDICATING THE RESULT ØF THE LAST ØPERATIØN)
;    IS MØVED TØ THAT ADDRESS.
;
FVWAIT::
        MØV     @2(R5),R1       ; GET LUN
        BITB    #FVBT.R,FVBT-1(R1)      ; HAS THE FILE BEEN ACCESSED?
        BEQ     10$                             ; BRANCH IF NOT,
        MØVB    EFNTAB-1(R1),WTSE+W,TSEF                ; YES - WAIT.
        DIR$    #WTSE,DIRERR
10$:    CMP     (R5),#2                         ; 2ND ARGUMENT PRESENT?
        BNE     20$                             ; BRANCH IF NOT.
        MØV     4(R5),R2                        ; YES, GET @(RETURN IØST).
        ASH     #2,R1                           ; DØUBLE WØRD ØFFSET FØR LUN.
        MØV     IØSTAT(R1),(R2)+                        ; MØVE I/Ø STATUS BLØCK.
        MØV     IØSTAT+2(R1),(R2)
20$:    RTS     PC                              ; RETURN.
;
.SBTTL  FVRWND -- REWINDS TAPE ØR PØINTS TØ START ØF DISK FILE.
;
;                       ENTRY:  CALL FVRWND(LUN)
;
;       WHERE LUN = LØGICAL UNIT NUMBER ØF TAPE ØR DISK FILE.
;       IF 'LUN' DESIGNATES MAG TAPE, THE TAPE IS REWØUND.
;       IF 'LUN' DESIGNATES DISK, THE VIRTUAL BLØCK NUMBER FØR THE NEXT
;       ACCESS IS SET TØ 1.
;
FVRWND::
        MØV     @2(R5),R1                       ; GET LUN
        BITB    #FVBT.D,FVBT-1(R1)              ; MAG TAPE?
        BNE     10$                             ; BRANCH IF YES.
;       DISK - SET VIRTUAL BLØCK NUMBER TØ 1 FØR NEXT ACCESS ØN LUN.
        ASH     #2,R1                           ; GET DØUBLE-WØRD INDEX FØR LUN.
        CLR     NXTREC-4(R1)
        MØV     #1,NXTREC-2(R1)
        RTS     PC                              ; RETURN.
;       WAIT FØR ANY PREVIØUS I/Ø AND REWIND MAG TAPE.
10$:    MØV     2(R5),WAITAR+2                  ; LUN TØ FVWAIT ARGUMENT LIST
        MØV     R5,-(SP)                        ; SAVE R5
        MØV     #WAITAR,R5                      ; CALL FVWAIT(LUN)
        JSR     PC,FVWAIT
        MØV     (SP)+,R5                        ; RESTØRE R5.
;       SET UP QIØ DPB.
        MØV     #IØ.RWD,QIØ+Q.IØFN
        MØV     @2(R5),R1                       ; GET LUN
        BISB    #FVBT.B,FVBT-1(R1)              ; INDICATE I/Ø IN PRØGRESS
        MØV     R1,QIØ+Q.IØLU
        MØVB    EFNTAB-1(R1),QIØ+Q.IØEF         ; EVENT FLAG
        MØV     #IØSTAT,QIØ+Q.IØSB              ; CØMPUTE @(I/Ø STATUS BLØCK)
        ASH     #2,R1                           ; DØUBLE WØRD ØFFSET
        ADD     R1,QIØ+Q.IØSB
        ASH     #-2,R1
        DIR$    #QIØ,DIRERR
        RTS     PC                              ; RETURN.
;
.SBTTL  PRSFNM -- PARSES FILE NAME AND STØRES IT IN DATASET DESCRIPTØR.
;
;                       ENTRY:  CALL PRSFNM(FILE,NCHAR,DSDESC,ERRØR [,LUN] )
;
;       THIS SUBRØUTINE PARSES THE 'NCHAR' BYTE FILE NAME IN 'FILE' AND
;       SETS UP A DATA SET DESCRIPTØR IN THE 6-WØRD ARRAY 'DSDESC'.
;       THE DATA SET DESCRIPTØR PØINTS TØ THE APPRØPRIATE LØCATIØNS IN
;       'FILE' FØR THE START ØF THE DEVICE NAME, DIRECTØRY, AND FILENAME
;       FIELDS. AN ØMITTED FIELD RESULTS IN A ZERØ ENTRY FØR THAT FIELD.
;       IF THE DEVICE NAME IS PRESENT AND THE FIFTH ARGUMENT 'LUN' IS
;       PRØVIDED, THE LØGICAL UNIT NUMBER IN 'LUN' IS ASSIGNED TØ THE
;       DEVICE.
;       IF THE DEVICE NAME IS PRESENT AND > 5 CHARACTERS (INCLUDING THE ":")
;       ØR IF A '[' IS FØUND WITHØUT A ']' ( BUT NØT VICE VERSA),
;       'ERRØR' IS SET TØ -1, ØTHERWISE, 'ERRØR' IS SET TØ 0 UPØN RETURN.
;
;       DATA AREA
;
```

```
ADSDES: .WØRD   0                  ; PØINTER TØ DATASET DESCRIPTØR
CHR:    .WØRD   0                  ; BUFFER FØR TEMPØRARY CHARACTER STØRAGE
;
PRSFNM::
        MØV     2(R5),R0           ; @(FILE NAME)
        CLR     @8.(R5)            ; CLEAR 'ERRØR'.
;   ZERØ DATA SET DESCIPTØR.
        MØV     6(R5),R2           ; @(DATASET DESCRIPTØR)
        MØV     R2,ADSDES
        MØV     #6,R3              ; LØØP CØUNTER
10$:    CLR     (R2)+
        SØB     R3,10$
        MØV     @4(R5),R1          ; # CHARACTERS IN FILE NAME
        BEQ     PRSRET             ; IF 0, RETURN.
;   SEARCH FØR DEVICE NAME FØLLØWED BY ':'.
        MØV     R0,R2              ; PØINT TØ START ØF STRING
        CLR     R3                 ; CLEAR DEVICE NAME CHAR CØUNT
20$:    INC     R3
        CMPB    (R2)+,#':          ; TEST FØR ':'
        BEQ     30$                ; BRANCH IF FØUND.
        SØB     R1,20$             ; NØT ':', CHECK CHAR CØUNT.
;   NØ ':' FØUND, RESET PØINTERS.
        MØV     @4(R5),R1          ; RESET # CHARACTERS
        MØV     R0,R2              ; PØINT TØ START ØF STRING
        ADD     #4,ADSDES          ; BYPASS DEVICE NAME DESCRIPTØR
        BR      DIRSCN
;   ':' FØUND, SET UP DEVICE NAME DESCRIPTØR AND ASSIGN LUN IF REQUESTED.
30$:    CMP     R3,#5              ; IF # CHARS IN DEVICE NAME >5, ERRØR.
        BLE     35$
        JMP ERRØR
35$:    MØV     R3,@ADSDES         ; # CHARACTERS IN DEVICE NAME.
        INC     ADSDES             ; INCREMENT PØINTER TØ ADDRESS
        INC     ADSDES             ;  PØRTIØN ØF DEVICE NAME DESCRIPTØR.
        MØV     R0,@ADSDES         ; @(DEVICE NAME)
        INC     ADSDES             ; PØINT TØ DIRECTØRY DESCRIPTØR.
        INC     ADSDES
        DEC     R1                 ; SKIP ':'
        CMPB    (R5),#5            ; CHECK FØR FIFTH ARGUMENT.
        BNE     DIRSCN
; ASSIGN LUN.
        MØV     R0,R4
        MØVB    (R4)+,ALUN+A.LUNA    ; MØVE PHYSICAL DEVICE NAME
        MØVB    (R4)+,ALUN+A.LUNA+1
;   CØNVERT PHYSICAL UNIT NØ. TØ BINARY.
        CLR     R3                 ; CØNVERT CHARACTERS UP TØ ':'
40$:    CMPB    (R4),#':           ; TØ UNIT NØ.
        BEQ     50$
        MØVB    (R4)+,CHR          ; NØT ':', MØVE TØ TEMP STØRAGE
        SUB     #'0,CHR            ; CØNVERT TØ DECIMAL DIGIT
        MUL     #10.,R3            ; MULTIPLY PREVIØUS RESULT
        ADD     CHR,R3             ; ADD NEW CHARACTER
        BR      40$
;   SET UP DPB TØ ASSIGN UNIT.
50$:    MØV     R3,ALUN+A.LUNU
        MØV     @10.(R5),ALUN+A.LULU
        DIR$    #ALUN,DIRERR
;
;   AT THIS PØINT, R0 PØINTS TØ THE FIRST CHARACTER ØF THE FILENAME STRING.
;                  R1 CØNTAINS THE NUMBER ØF CHARACTERS REMAINING TØ BE SCANNED.
;                  R2 PØINTS TØ THE FIRST CHARACTER FØLLØWING THE ':' SYMBØL.
;
;   SCAN FØR DIRECTØRY IDENTIFICATIØN "[UIC]".
DIRSCN: MØV     R2,R0              ; MAINTAIN PØINTER TØ START ØF PØSSIBLE UIC
        CMPB    (R2),#'[           ; IS NEXT CHARACTER '['?
        BEQ     5$                 ; BRANCH IF YES.
        ADD     #4,ADSDES          ; NØ UIC. SKIP DIRECTØRY DESCRIPTØR.
        BR      FNMØVE
55$:    CLR     R3                  ; CLEAR CHARACTER CØUNT.
10$:    INC     R3                  ; INCREMENT CHARACTER CØUNT.
        CMPB    (R2)+,#']           ; SCAN UNTIL ']' FØUND
        BEQ     20$                 ;  ØR UNTIL ALL CHARACTERS SCANNED.
        SØB     R1,10$
;   NØ CLØSING ']' - ERRØR.
        JMP     ERRØR
20$:    MØV     R3,@ADSDES          ; MØVE # CHARACTERS IN DIRECTØRY NAME
        INC     ADSDES              ;   TØ DIRECTØRY DESCRIPTØR.
        INC     ADSDES
        MØV     R0,@ADSDES          ; MØVE ADDRESS ØF DIRECTØRY NAME
        INC     ADSDES              ;  TØ DIRECTØRY DESCRIPTØR.
        INC     ADSDES
        DEC     R1                  ; SKIP ']'
;
;   AT THIS PØINT, R2 PØINTS TØ START ØF REMAINDER ØF STRING.
;                  R1 CØNTAINS THE # ØF CHARACTERS REMAINING TØ BE PRØCESSED.
;
;   SET UP FILENAME DESRIPTØR.
FNMØVE:
        MØV     R1,@ADSDES
        INC     ADSDES
        INC     ADSDES
        MØV     R2,@ADSDES
PRSRET: RTS     PC
;
;   FILENAME STRING CØNTAINES DEVICE NAME > 5 CHARACTERS ØR '[' WITHØUT
;   CLØSING ']', SET ERRØR FLAG.
;
ERRØR:  DEC     @8.(R5)
        RTS     PC
        .END
```

## 13. [131,140] GETCOO.FTN

The subroutine GETCOO lets the user accept or change the default coordinates of the cluster map file and theme display that are passed in blobal common.

- Calling sequence

  CALL GETCOO (IC,TC,ISET)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| IC | I | 4 | Out | Coordinates for theme display |
| TC | I | 4 | Out | Coordinates for cluster map file |
| ISET | I | | Out | Flag set to 1 when user wants to exit |

```
MFØRTRAN IV-PLUS V02-04                    14:25:48    31-AUG-77            PAGE 1
GETCØØ.FTN        /TR:BLØCKS/WR
0001              SUBRØUTINE GETCØØ(IC,TC,ISET)
           C
           C                        THIS PRØGRAM GETS CØØRDINATES FRØM THE USER
           C                        FØR THE CLUSTER MAP FILE AND THEME DISPLAY
           C                        IF CØØRDINATES ØTHER THEN THE DEFAULTS ARE
           C                        DESIRED
           C
0002              IMPLICIT INTEGER(A-Z)
0003              INCLUDE '[300,3]CAMSCØMØN.INC'
0004 *           INCLUDE 'SY:[300,3]CAMSPARAM.INC'
0005 *           PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
     *          1,MAXV=11,NDØTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
     *          2NØSPWD=6,NØDTWD=10
0006 *           EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
0007 *           INTEGER C1(469),C2(256),C3(71),C4(348),C5(629)
     *     C*
0008 *           INTEGER ACDATE,SUBCAT,SUBPØP,CATKNT,CATTH
0009 *           BYTE CHNVEC,NØCHAN,NØSUB,DØTCAT,DØTCLU
0010 *           CØMMØN/CØM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NØCHAN,NØSUB,
     *          1SUBCAT(MAXSUB),SUBPØP(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NØDØ,
     *          2NØDU,NØTH,DØTCAT(NDØTS),DØTCLU(NDØTS)
     *     C*
0011 *           INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DØTDAY,PDATE1,TDATE1
0012 *           INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDØM,GRID
0013 *           BYTE DELFLG,NØACD,SØILGR,SUNEL,NSTART,NTYPE1,ALP,ALPØ
0014 *           BYTE PCTCT,PCTCTØ,VAR,VARØ,DLABEL,TYPE
0015 *           CØMMØN/CØM2/ISEG,DELFLG,NØACD,ADATES(2,MAXACD),SØILGR(MAXACD),
     *          1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
     *          2DØTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
     *          3PDATE3(2),TDATE3(2),NØCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPØ,
     *          4        PCTCT(MAXCAT),PCTCTØ,VAR(MAXCAT),VARØ
     *     C*
0016 *           INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
     *          1UFLAG4
0017 *           INTEGER PFLAG,DSKNNT
0018 *           CØMMØN/CØM3/PFLAG,DSKNNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
     *          1,UFLAG2,UFLAG3,UFLAG4,NEWLAB(MAXSUB)
     *     C*
0019 *           INTEGER TX1,TY1,TX2,TY2,ACDISP,G,B,DTWIND,DØTARY,GMIN,GMAX,FUL
0020 *           INTEGER SPWIND,CLAWND,CLUWND
0021 *           CØMMØN/CØM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
     *          1B(4),DTWIND(5,NØDTWD),SPWIND(5,NØSPWD),IMWIND(4),NUMDØT,
     *          2DØTARY(NDØTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
0022 *           CØMMØN/CØM5/DISKID,RANDØM(NDØTS),GRID(NDØTS),DLABEL(NDØTS),
     *          1TYPE(NDØTS),RECLNC
0023              DIMENSIØN IC(4),TC(4)
0024              DIMENSIØN CØØRD(4)
0025              BYTE W(74)
0026              ISET=0
0027              IC(1)=IX1
0028              IC(2)=IY1
0029              IC(3)=IX2
0030              IC(4)=IY2
0031              TC(1)=TX1
0032              TC(2)=TY1
0033              TC(3)=TX2
```

13-2

```
               TC(4)=TY2
         10    CØNTINUE
               WRITE(6,820) TC
               WRITE(6,830)
               CALL ØUTPUT(7)
               READ(6,800) W
               CALL FRØNT(W,74)
               IF(W(1).EQ.'R') GØ TØ 10
               IF(W(1).EQ.' ') GØ TØ 40
               IF(W(1).EQ.'X') GØ TØ 777
               IPT = 0
               DØ 15 I=1,4,2
                  CALL INTFF(IPT,W,74,CØØRD(I))
                  IF(CØØRD(I).GT.NPIX.ØR.CØØRD(I).LT.1) GØ TØ 20
                  CALL INTFF(IPT,W,74,CØØRD(I))
                  IF(CØØRD(I+1).GT.NLIN.ØR.CØØRD(I+1).LT.1) GØ TØ 20
         15       CØNTINUE
               GØ TØ 30
         20    CØNTINUE
               WRITE(6,850)
               GØ TØ 10
         30    CØNTINUE
               DØ 35 I=1,4
                  TC(I)=CØØRD(I)
                  CØØRD(I)=0
         35       CØNTINUE
         40    WRITE(6,840) IC
               WRITE(6,830)
               CALL ØUTPUT(7)
               READ(6,800)W
               CALL FRØNT(W,74)
               IF(W(1).EQ.'R') GØ TØ 10
               IF(W(1).EQ.' ') GØ TØ 70
               IF(W(1).EQ.'X') GØ TØ 777
               IPT=0
               DØ 45 I=1,4
                  CALL INTFF(IPT,W,74,CØØRD(I))
         45       CØNTINUE
               IF(CØØRD(3).EQ.0) CØØRD(4)=CØØRD(2)+NLIN
               IF(CØØRD(3).EQ.0) CØØRD(3)=CØØRD(1)+NPIX
               DØ 50 I=1,4
                  IF(CØØRD(I).LE.512.AND.CØØRD(I).GE.1) GØ TØ 50
                  WRITE(6,850)
                  GØ TØ 40
         50       CØNTINUE
               DØ 60 I=1,4
                  IC(I)=CØØRD(I)
                  CØØRD(I)=0
         60       CØNTINUE
         70    ISET=1
         777   CØNTINUE
               RETURN
         800   FØRMAT(74A1)
         820   FØRMAT(/' SEGMENT CØØRDINATES ARE  >',I3,',',I3,4X,I3,',',I3)
         830   FØRMAT('$ NEW CØØRDINATES IF DESIRED  >')
         840   FØRMAT(/' DISPLAY CØØRDINATES ARE  >',I3,',',I3,4X,I3,',',I3)
```

```
FØRTRAN IV-PLUS V02-04          11125148    31-AUG-77          PAGE 3
GETCØD.FTN          /TR:BLOCKS/WR
0090      850     FØRMAT('S ***   ERRØR IN CØØRDINATES - TRY AGAIN   ***')
0091      860     FØRMAT(//' ***   INPUT ERRØR   ***')
0092              END
```

## 14. HPROS

This program is documented in reference 1.

```
HFØRTRAN IV-PLUS V02-04                14:26:17    31-AUG-77            PAGE 1
HPRØS.FTN        /TR:BLØCKS/WR
            SUBRØUTINE HPRØS(LUN,F11,BUF,FMT,EØF,PRTY,SHFG)
            IMPLICITINTEGER(A-Z)
            CØMMØN/HCØM/SS,SE,LS,LE,NRPDS,NDSPR,NCPR,NRPC,ANCL,NC,NS,
           1 NBIT,DØI,NCAR,SVD,RSIZ,PSKIP,HSIZ,CALP,CERR
              CØMMØN/IØLP/IØ,LP
            BYTE BUF(1)
              BBUF(I)=FFUNC(BUF(I))*256+FFUNC(BUF(I+1))
              IF(CERR.NE.0)RETURN
            IF(FMT.EQ.1) GØ TØ 1
            IF(FMT.EQ.2) GØ TØ 2
            IF(FMT.EQ.3) GØ TØ 3
            WRITE(IØ,1000) FMT
      1000  FØRMAT(' FLAKEY FØRMAT SPECIFICATIØN = ',I5,' HPRØS TERMINATES.'//
           1 /)
              CERR = 9
              RETURN
      1     SS = BBUF(108)
            SE=BBUF(110)
            LE = 0
            NRPDS=BUF(104)
            NDSPR=BUF(1776)
            NCPR=BUF(102)
            NRPC=BUF(103)
            ANCL=BBUF(105)
            NC=BUF(90)
C
              NS=BBUF(1787)
C
            NBIT=BUF(91)
            DØI = BUF(107)
            NCAR=BBUF(1785)
            SVD = BBUF(92)
            RSIZ = BBUF(100)
            PSKIP = 0
            HSIZ = 1530
            CALP = 0
            GØ TØ 4
      2     SS = 1
            SE = BBUF(23)-6
            LE = BBUF(39)
            NRPDS = 1
            NDSPR = 1
            NCPR = BBUF(19)
            NRPC = 1
            ANCL = 4
            NC = NCPR
            NS = SE
            NBIT = 8
            DØI = 0
            NCAR = NCPR
            SVD = 1
            RSIZ = NC*(NS+6)+4
            PSKIP = 0
            HSIZ = 400
            CALP = 6
            GØ TØ 4
```

~~14~~-2

115

```
      3     NS = BBUF(39)/4
              I=BUF(14)
              SS = NS*(IAND(I,15)-1)+1
            SE = SS+NS-1
            LS = 1
            LE = 2340
            NRPDS = 1
            NDSPR = 1
            NCPR = 4
            NRPC = 1
            ANCL = 0
            NC = 4
            NBIT = 8
            DØI = 2
            NCAR = 4
            SVD = 1
            HSIZ = BBUF(17)
            PSKIP = 9
            HSIZ = 338
            CALP = 0
            CALL RREAD(LUN,F11,BUF(41),624,NCT,EØF,PRTY)
            IF(EØF.NE.0) GØ TØ 4
            WRITE(IØ,1001)
      1001  FØRMAT(' EØF ENCØUNTERED IN HPRØS,  PRØGRAM TERMINATES.'//)
              CERR = 10
              RETURN
      4     IF(SHFG.NE.0) RETURN
            ØPEN(UNIT = 3,TYPE='UNKNØWN',NAME='[300,1]HEADER.DAT'
           1 ,FØRM='UNFØRMATTED',ACCESS ='SEQUENTIAL',DISPØSE='SAVE')
              H2=HSIZ*2
            WRITE(3)(BUF(I),I=1,H2)
            CLØSE(UNIT=3)
            RETURN
            END
```

## 15. HVFY

This program is documented in reference 1.

```
HFØRTRAN IV-PLUS V02-04                  14126143     31-AUG-77               PAGE 1
HVFY.FTN            /TRIBLOCKS/WR
                    SUBROUTINE HVFY(WR,FM)
                    IMPLICIT INTEGER (A-Z)
                    BYTE WR(1)
                    COMMON /ICOM/SS,SE,LS,LE,NRPDS,NDSPR,NCPR,NRPC,ANCL,NC,NS,
                   1 NBIT,DKI,NCAR,SVD,RSIZ,PSKIP,HSIZ,CALP,CERR
                    COMMON/IOLP/IO,LP
          D         WRITE(LP,1000)SS,SE,LS,LE,NRPDS,NDSPR,NCPR,NRPC,ANCL,NC
          D1000     FORMAT(' HCOM'/(' ',10I10))
          D         WRITE(LP,1007)NS,NBIT,DKI,NCAR,SVD,RSIZ,PSKIP,HSIZ,CALP
          D1007     FORMAT(' ',10I10)
                    IF(DKI.EQ.0.OR.(FM.EQ.3.AND.DKI.EQ.2)) GØ TØ 1
                    WRITE(IO,1001)DKI
          1001  FØRMAT(' DKI = ',I5,' UNACCEPTABLE')
                    GØ TØ 99
          1     IF(NBIT.EQ.8) GØ TØ 2
                    WRITE(IO,1002)NBIT
          1002  FØRMAT(' NBIT = ',I5,' UNACCEPTABLE')
                    GØ TØ 99
          2     IF(NRPC.EQ.1) GØ TØ 3
          D         WRITE(IO,1003)NRPC
          D1003  FØRMAT(' NRPC = ',I5,' UNACCEPTABLE')
          C
          C
          C         FUDGE FØR CHUMMY UMR TAPES.
          C
          C         NRPC SET TØ 1
          C
          C         NCPR SET TØ NC (MAYBE)
          C
          C
                    NRPC=1
                    NCPR=0
                    IF(NRPDS.GT.1) NCPR=(NC-NCAR)/(NRPDS-1)
          D         WRITE(IO,1006)NCAR,NCPR,NRPC
          D1006     FØRMAT(' NEW NCAR = ',I5,' NCPR = ',I5,' NRPC = ',I5/)
                    GØ TØ 3
          3         IF(NDSPR.GE.1)GØ TØ 4
                    WRITE(IO,1005)NDSPR
          1005      FØRMAT(' NDSPR = ',I5,' UNACCEPTABLE'/
                    1 ' RESET TO 1'/)
                    NDSPR=1
                    GØ TØ 4
          4         RETURN
          99    WRITE(IO,999)
          999   FØRMAT(' HVFY TERMINATES.'///)
                    CERR=10
                    RETURN
                    END
```

ORIGINAL PAGE IS
OF POOR QUALITY

## 16. LECTAP

See appendix B for a description of this program.

```
            .TITLE  LECTAP
            .IDENT  /010377/
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;           ;
;           PRØGRAM  ; FØRTRAN CØMPATIBLE MAG TAPE SUBRØUTINE       ;
;           TITLE    ; LECTAP ;
;           AUTHØR   ; G. CRIDLAND           ;
;           CØMPANY  ; G.E.(MØDIFIED BY AERØNUTRØNIC-FØRD)          ;
;                  ; RESTØRED TØ GE STANDARD (MØRE ØR LESS) BY LEC
;           FUNCTIØN ; TØ PERFØRM THE NECESSARY TAPE ØPERATIØNS FØR       ;
;           FØRTRAN PRØGRAMS. ;
;           ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
            .MCALL  MØUT$,MØWAS$,DIR$,CLEF$S,WTSE$S,QIØ$
            .MCALL  SETF$S
            .MCALL  ALUN$
            .MCALL  MØUT$S
            .MCALL  EXIT$S
            .MCALL FSRSZ$,FINIT$
;
            .GLØBL  TINIT,TRLSE,TREAD,TWRIT,TUNLD,TEØF
            .GLØBL  TRWD,TSKIP,TSTAT,TSET,TWAIT
;
            .GLØBL  TFILE,TATCH
;
;             INITIALIZE REQUESTED MAG TAPE UNIT
;
            FSRSZ$  1
TINIT:      ;
            FINIT$
            CLR     MTLSFT  ;INITIALIZE LAST FUNCTIØN ISSUED.
            MØV     R3,-(SP)         ; SAVE REGISTER USED
            TST     @4(R5)
            BEQ     1$
            MØV     #ASGXT,R3
            BR      2$
1$:         MØV     #ASGMT,R3
2$:
            MØV     @6(R5),A.LUNU(R3);
            MØV     @2(R5),A.LULU(R3)
            TST     @4(R5)
            BEQ     3$
            DIR$    #ASGXT
            BR      4$
3$:         DIR$    #ASGMT
4$:
            MØV     (SP)+,R3         ; RESTØRE REGISTER USED.
            RTS     PC
ASGMT: ALUN$        0,MT,0           ; ASSIGN LUN DIRECTIVE
ASGXT:  ALUN$       0,XT,0  ;ASSIGN LUN FØR XT
            ;
            ;
            ; MAG TAPE WAIT - FAKE ØUT WITH QUICK EXIT
            ;
            ;
TWAIT:                              ;
            RTS     PC               ; QUICK EXIT.
;
;             RELEASE MAG TAPE UNIT
TRLSE:      ;
```

~~16-2~~

120

```
        MØV     #8.,MTFCSV          ; SET TO DETACH MAG TAPE UNIT.
        MØV @2(R5),QIØDPB+Q.IØLU
        JMP     MTQIØ  ;
;
;       ATTACH UNIT.   CALL MTINIT FIRST...
;
TATCH:
        MØV @2(R5),QIØDPB+Q.IØLU
        CLR     MTFCSV
        JMP     MTQIØ
;
;       READ MAGNETIC TAPE UNIT
;
TREAD:  ;
        MØV @2(R5),QIØDPB+Q.IØLU
        MØV     #1,MTFCSV           ; SET FØR READ REQUEST.
MTRW:;
        MØV     R0,-(SP)          ; SAVE R0
        MØV     R1,-(SP)          ;
        MØV     #QIØDPB,R0          ; SET R0 WITH ADR ØF QUEUE I/Ø DPB
        MØV     4(R5),Q.IØPL(R0); SET QIØ DPB PARM LIST TØ START ADR
        MØV     @6(R5),R1         ; GET WØRD CØUNT
        ASH     #1,R1             ; DØUBLE WØRD CØUNT
        MØV     R1,Q.IØPL+2(R0)  ; STØRE WØRD CØUNT IN PARAMETER LIST WØRD 2
        JSR     PC,MTQIØ          ;
MTEXIT: ;
        MØV     (SP)+,R1          ; RESTØRE R1.
        MØV     (SP)+,R0          ; RESTØRE R0.
        RTS     PC       ; EXIT.
;
;       WRITE
;
TWRIT:  ;
        MØV @2(R5),QIØDPB+Q.IØLU
        MØV     #2,MTFCSV           ;
        JMP     MTRW   ;
;       UNLØAD MAGNETIC TAPE UNIT
;
TUNLD:  ;
        MØV @2(R5),QIØDPB+Q.IØLU
        MØV     #7,MTFCSV           ;
MTU2:;
        JSR     PC,MTQIØ          ;
        RTS     PC                ; EXIT.
;
;       WRITE *** END-ØF-FILE ***
;
TEØF:   ;
        MØV @2(R5),QIØDPB+Q.IØLU
        MØV     #3,MTFCSV           ;
        BR      MTU2   ;
;
;       REWIND
;
TRWD:   ;
        MØV @2(R5),QIØDPB+Q.IØLU
        MØV     #4,MTFCSV           ;
        BR      MTU2   ;
;
;       MAGNETIC TAPE *** QUEUE I/Ø ***
;
MTQIØ:  ;
        MØV     R0,-(SP)          ;
        MØV     #QIØDPB,R0          ;
MTIØ:;
        MØV     R1,-(SP)          ;
```

```
             MØV      R2,-(SP)          ;
             MØV      #MTFCTB,R1         ;
             MØV      MTFCSV,R2           ;
             ASH      #1,R2  ;
             ADD      R2,R1  ;
             MØV      (R1),Q.IØFN(R0) ;
             MØV      (SP)+,R2          ; RESTØRE REGISTERS USED
             MØV      (SP)+,R1          ;
             DIR$     #QIØDPB; REQUEST MAGNETIC TAPE FUNCTIØN
             WTSE$S   #1
             MØV      (SP)+,R0          ; RESTØRE R0
             RTS      PC         ; EXIT.
             ;
MTFCTB:        ;
             .WØRD    IØ.ATT ;    0 ; ATTACH
             .WØRD    IØ.RLB ;    2 ; READ LØGICAL BLØCK
             .WØRD    IØ.WLB ;    2 ; WRITE LØGICAL BLØCK
             .WØRD    IØ.EØF ;    3 ; WRITE E-Ø-F
             .WØRD    IØ.RWD ;    4 ; REWIND
             .WØRD    IØ.SPB ;    5 ; SKIP RECØRD
             .WØRD    IØ.SPB ;    6 ; BACKSPACE RECØRD
             .WØRD    IØ.RWU ;    7 ; UNLØAD
             .WØRD    IØ.DET ;    8 ; RELEASE
             .WØRD    IØ.STC ;    9 ; SET CHARACTERISTICS
             .WØRD    IØ.SEC ;   10 ; SENSE CHARACTERISTICS
             .WØRD    IØ.SPF  ; 11 ; SKIP FILES.
;
;              BUILD STAUS WØRD FØR RETURN TØ USER
MTFCSV:  .=.+2 ; MAG TAPE FUNCTIØN ISSUED
MTLSFT:  .=.+2 ; LAST MAG TAPE FUNCTIØN ISSUED
MTUNIT:  .WØRD                         ; SAVE TAPE UNIT NØ. HERE
         .CSECT STATUS
MTSWD:   .WØRD    0
         .WØRD    0
         .CSECT
MTSW:    .=.+2 ; BUILD RETURN STATUS HERE
SAVE:    .=.+2  ;LAST CØMMAND FØR MTSTAT
;
;              SKIP (N) RECØRDS
;
TSKIP:     ;
         MØV @2(R5),QIØDPB+Q.IØLU
         MØV      #5,MTFCSV          ;
MTSKIØ:    ;
         MØV      R0,-(SP)          ;
         MØV      #QIØDPB,R0          ;
         MØV      @4(R5),Q.IØPL(R0) ;
         BR        MTIØ    ;
;
;             SET DENSITY AND PARITY (SEVEN) TRACK ØNLY
;
TSET:    ;
         MØV @2(R5),QIØDPB+Q.IØLU
         MØV      #9.,MTFCSV        ;
         MØV      R0,-(SP)          ;
         MØV      #QIØDPB,R0          ;
         CLR      Q.IØPL(R0)          ;
         MØV      R3,-(SP)          ; SET DENSITY REQUESTED
         MØV      @4(R5),R3            ;
         ASH      #1,R3   ;
         ADD      #MTDEN,R3            ;
         BIS      (R3),Q.IØPL(R0) ;
         MØV      (SP)+,R3          ;
         ;
         TST      @6(R5) ; PARITY ØDD ØR EVEN ?
         BEQ      MTØDD  ;
```

```
        BIS     #10,0,IØPL(RO)   ; EVEN
MTØDD:  ;
        JMP     MTIØ    ;
        ;
MTDEN:  .WØRD   2           ;    0 ; 200 BPI
        .WØRD   1           ;    1 ; 556 BPI
        .WØRD   0           ;    2 ; 800 BPI
        .WØRD   40         ;    3 ; 800 BPI DUMP MØDE
;
;       MTFILE(UNIT,NUM)  NUM = + FØRWARD,  NUM = - REVERSE.
;
TFILE:
        MØV @2(R5),CIØDPR+0,IØLU
        MØV     #11,,MTFCSV
        BR      MTSKIØ
;
;         SENSE CHARACTERISTICS
;
TSTAT:    ;
        MØV @2(R5),OIØDPR+0,IØLU
        CMP     MTFCSV,#10.
        BEQ     1$
        MØV     MTFCSV,SAVE
1$:
        MØV     #10.,MTFCSV      ; INDICATE READ CHARACTERISTICS REQUEST
        JSR     PC,MTQIØ         ;
        MØV     SAVE,MTSW
        BIC     #177760,MTSW
        MØV     R1,-(SP)         ;
        MØV     MTSWD+2,R1       ; GET RETURNED SET/SENSE STATUS WØRD
        BIT     #3,R1   ; TAPE DENSITY 800 BPI ?
        BNE     MT556   ;
        BIS     #40000,MTSW      ; YES,
        BR       CHKDMP ;
MT556:  ;
        BIT     #1,R1   ;
        BNE     CHKDMP ;
        BIS     #20000,MTSW      ;
        BR       CHKDMP ;
CHKDMP:  ;
        BIT     #4,R1;
        BEQ     MTEØT   ;
        BIS     #60000,MTSW      ;
MTEØT:  ;
        BIT     #20,R1 ; TAPE PAST E-Ø-T MARKER ?
        BEQ     MTLSTF ;
        BIS     #1000,MTSW        ; YES,
MTLSTF:  ;
        BIT     #40,R1 ; LAST CØMMAND ENCØUNTERED EØF RECØRD ?
        BEQ     WRTLCK ;
        BIS     #200,MTSW          ; INDICATE TAPE PAST EØF ]
WRTLCK:  ;
        BIT     #2000,R1         ; WRITE LØCK ØN ?
        BEQ     MT7CH   ; NØ,
        BIS     #2000,MTSW        ; YES,
MT7CH:  ;
        BIT     #10000,R1          ; UNIT IS 7-CHANNEL ?
        BEQ     MTLDPT ;
        BIS     #10000,MTSW      ;
MTLDPT:  ;
        BIT     #20000,R1          ;
        BEQ     RTNSW   ; NØ, RETURN SET/SENSE WØRD,
        BIS     #400,MTSW          ; YES,
        ;
RTNSW:  ;
        MØV     (SP)+,R1         ; RESTØRE REGISTERS USED
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
        MØV     MTSW,@4(R5)     ;
        CLR     @6(R5) ; ??? RETURN RESIDUE COUNT OF ZERO
        RTS     PC      ;
;
QIØDPB: QIØ$    IØ.RLB,0,1,,MTSW?,0,<0,0,0,0>
;
;
```

# 17. LIN.FTN

<u>ENTRY POINT - LIN</u>

Line INput with character checking. Input a line from device number II and checks for: 1) X, 2) B, 3) R, 4) CR, 5) normal data input, and 6) routine commands to change the value of TI. See CTR documentation.

- Calling sequence

```
CALL LIN (IO,L,TI,I)
Go to (1, 2, 3, 4, 5, 6), I
```

1. X entered - exit immediately
2. B entered - go to previous query
3. R entered - restart program
4. RQ - ask query again
5. CR entered
6. Normal data entry

```
HFØRTRAN IV-PLUS V02-04                14:27:42    31-AUG-77              PAGE 1
LIN.FTN                 /TR:BLØCKS/WR
                        SUBRØUTINE LIN(IØ,LINE,TI,FLAG)
                        IMPLICIT INTEGER (A-Z)
                        BYTE LINE(74),LUTAB(10),LUNN,ATF
                        DATA ATF/1/
            C           FLAG = 1  X
            C                  2  R
            C                  3  RESTART
            C                  4  REQUERY
            C                  5  C/R
            C                  6  NØRMAL INPUT
            9999        IF(TI.EQ.IØ)CALL ØUTPUT(7)
                        READ(TI,9000,END=9910)(LINE(I),I=1,74)
            9000      FØRMAT(74A1)
                      CALL FRØNT(LINE,74)
                      IF(LINE(1).EQ.'=') GØ TØ 9200
                        IF(LINE(1).EQ.'!')GØ TØ 9450
                        IF(LINE(1).EQ.'@') GØ TØ 9500
                        IF(LINE(2).NE.' ') GØ TØ 999
                      IF(LINE(1).EQ.'B') GØ TØ 9100
                      IF(LINE(1).EQ.'R') GØ TØ 9300
                      IF(LINE(1).EQ.'X') GØ TØ 9400
            999         CØNTINUE
                        FLAG = 6
                      IF(LINE(1).EQ.' ')FLAG = 5
                        RETURN
            9910        IF(ATF.EQ.1)WRITE(IØ,9911)TI
            9911        FØRMAT(' END-ØF-FILE ENCØUNTERED ØN LUN ',I2,', =K ASSUMED.'/)
                        GØ TØ 9912
            9400        FLAG = 1
                        RETURN
            9450        WRITE(IØ,9451)LINE
            9451        FØRMAT(' ',74A1)
                        GØ TØ 9999
            9500        CØNTINUE
                        LP = 1
            9501        LP = LP + 1
                        IF(LP.GT.33)GØ TØ 9502
                        IF(LINE(LP).NE.' ') GØ TØ 9501
            9502        LINE(LP)=0
                        IF(ATF.EQ.0)CLØSE(UNIT=3)
                        ATF = 0
                        ØPEN(UNIT=3,NAME=LINE(2),TYPE='ØLD',READØNLY)
                        LUNN = LUNN + 1
                        LUTAB(LUNN)=TI
                        TI = 3
                        GØ TØ 9999
            9200      IF(LINE(2).EQ.'K') GØ TØ 9250
                      IF(LINE(2).NE.'F') GØ TØ 9299
                        LP = 2
                      CALL INTFF(LP,LINE,74,TI)
                      LUNN = LUNN + 1
                      LUTAB(LUNN)=TI
                        LP = LP + 1
                      IF(LINE(LP).EQ.',')LP = LP + 1
                      CALL FRØNT(LINE(LP),75-LP)
                      IF(LINE(LP).EQ.' ') GØ TØ 9240
```

17-2

126

```
FØRTRAN IV-PLUS V02-04          14:27:42    31-AUG-77          PAGE 2
LIN.FTN            /TR:BLOCKS/WR
              LPP = LP - 1
        9201  LPP = LPP + 1
              IF(LPP.GT.LP+31) GØ TØ 9202
              IF(LINE(LPP).NE.' ') GØ TØ 9201
        9202  LINE(LPP)=0
              ØPEN(UNIT=TI,NAME=LINE(LP),TYPE='ØLD',READØNLY)
              GØ TØ 9999
        9240    IF(TI.EQ.IØ)GØ TØ 9290
                GØ TØ 9999
        9250  IF(LINE(3).NE.' ') GØ TØ 9260
        9912    LUNN = LUNN - 1
              IF(LUNN.LE.0) GØ  TØ 9251
              TI = LUTAB(LUNN)
                IF(TI.NE.IØ)GØ TØ 9999
                IF(ATF.EQ.0)CLØSE(UNIT=3)
                ATF = 1
              GØ TØ 9290
        9251  LUNN = 0
              TI = IØ
              GØ TØ 9290
        9260  LP = 2
              CALL INTFF(LP,LINE,74,LUNC)
              CLØSE(UNIT=LUNC,DISPØSE='SAVE')
              GØ TØ 9999
        9299  TI = IØ
              WRITE(IØ,9298)(LINE(I6),I6=1,73)
        9298  FØRMAT(' INPUT LINE IN ERRØR,  LINE FØLLØWS,'/' ',73A1)
              GØ TØ 9290
        9100    FLAG = 2
                RETURN
        9300    FLAG = 3
                RETURN
        9290    CØNTINUE
                FLAG = 4
                RETURN
              END
```

## 18. RREAD

This program is described in reference 1.

```
HFØRTRAN IV-PLUS V02-04                14128105    31-AUG-77           PAGE 1
RREAD.FTN          /TR:BLØCKS/WR
                   SUBRØUTINE RREAD(LUN,F11,BUF,PRSIZ,BCT,EØF,PRTY)
                     IMPLICIT INTEGER (A-Z)
                     CØMMØN/IØLP/IØ,LP
                   CØMMØN /STATUS/S
                     CØMMØN/FATAL/ZZ,RR
                   INTEGER BUF(1),S(2)
                   INTEGER PRTY,EØF,BCT,PRSIZ,F11,LUN
                     INTEGER CL(2)
                     DATA EØFNXT/0/
                   EØF = 1
          D          WRITE(LP,3000)F11,PRSIZ
          D3000      FØRMAT(' STEP 1 ',2I10)
                   IF(F11.EQ.0) GØ TØ 1
          D          WRITE(LP,3001)
          D3001      FØRMAT(' HUH???'///)
                     PCNT = 20
                     PRS=(PRSIZ+1)/2
          8          CALL TREAD(LUN,BUF(7),PRS)
                     IF(S(1).EQ."374)GØ TØ 7
                   IF(S(1).NE."366) GØ TØ 2
          3        EØF = 0
          2        BCT = S(2)
          D          WRITE(LP,3004)BCT
          D3004      FØRMAT(' BCT = ',I10)
                   RETURN
          7          PCNT = PCNT - 1
                     IF(PCNT.LE.0) GØ TØ 9
                     CALL TSKIP(LUN,-1)
                     GØ TØ 8
          9          PRTY = PRTY + 1
                     GØ TØ 2
          1          CØNTINUE
          D          WRITE(LP,3002)EØFNXT,PRSIZ
          D3002      FØRMAT(' STEP 2 ',2I10)
                     IF(EØFNXT.EQ.1) GØ TØ 4
                     CALL FVREAD(LUN,BUF,PRSIZ)
                     CALL FVWAIT(LUN,S)
          D          WRITE(LP,3003)(BUF(I),I=1,10)
          D3003      FØRMAT(' ',10I10)
                     RR = 0
                     EØFNXT = 0
                     DØ 5 IE=1,2
          5          IF(BUF(IE).NE.0) GØ TØ 6
                     EØFNXT = 1
          6          BCT = BUF(4)
          D          WRITE(LP,5032)(BUF(X),X=1,6),BCT,PRSIZ
          D5032      FØRMAT(' ',8I10/)
                     IF(PRSIZ.EQ.BCT)RETURN
                     RR =BCT
          C          RR .NE. 0 IS ERRØR RECØRD SIZE
                     BCT = IMIN0(BCT,PRSIZ)
                     IF(EØFNXT.EQ.0)CALL FVDSET(LUN,BUF(1),BUF(2))
                     RETURN
          4          EØF = 1
                     EØFNXT=0
                     RETURN
```

```
FORTRAN IV-PLUS V02-04          14:28:05      31-AUG-77             PAGE 2
PREAD.FTN               /TR:BLOCKS/WR
                     ENTRY RSKIP(LUN,F11,ISKIP,BUF)
                     LSKIP=ISKIP
                     IF(F11.EQ.0) GO TO 101
                     CALL TSKIP(LUN,LSKIP)
          102        RETURN
          101          IF(ISKIP.EQ.0)RETURN
                       LSKIP=IABS(ISKIP)
                       QQ=1
                       IF(ISKIP.LT.0)QQ=5
                       QP1=QQ+1
                       FST=0
                       I = 0
          107          DO104J=QQ,QP1
          104          IF(BUF(J).NE.0)GOTO105
                       IF(FST.NE.0)CALLFVDSET(LUN,CL(1),CL(2))
                       RETURN
          105          CALL FVDSET(LUN,BUF(QQ),BUF(QQ+1))
                       IF(I.GE.LSKIP) RETURN
                       CALL FVDGET(LUN,CL(1),CL(2))
                       CALL FVREAD(LUN,BUF,512)
                       CALL FVWAIT(LUN)
          D            WRITE(LP,3003)(BUF(X),X=1,10)
          103          FST=1
                       I = I + 1
                       GO TO 107
                       END
```

## 19. SHELL.FTN

Sorts an array via the method known as Shell sort.

- Calling sequence

CALL SHELL (NR,RS,B,P)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| NR | I | 1 | In | Number of records to be sorted |
| RS | I | 1 | In | Size of records to be sorted |
| B | I | (RS,NR) | In | Array of records |
| P | I | NR | In | For $1 \leq I \leq NR$, P(I) gives pointer to row of item B (.,I) to be considered the start of record 1. |

There exists a version of this called JASSIF in which B is a byte array, which makes a lot of difference on this machine.

```
FORTRAN IV-PLUS V02-04                14:28:25    31-AUG-77        PAGE 1
SHELL.FTN                /TR:BLOCKS/WR
                         SUBROUTINE SHELL(NR,RS,B,P)
                         INTEGER NR,RS,P(NR),M
                         INTEGER A,B(RS,NR)
                         M = NR
              1          M = M/2
                         IF(M,EQ,0)RETURN
                         K=NR-M
                         J = 1
              2          I=J
              3          IM=I+M
                         DO 301 L = 1,RS
                         LL1 = L+P(IM)-1
                         IF(LL1,GT,RS) GO TO 4
                         LL2 = L + P(I)-1
                         IF(LL2,GT,RS) GO TO 303
                         IF(B(LL1,IM),LT,B(LL2,I)) GO TO 303
              301        IF(B(LL1,IM),GT,B(LL2,I)) GO TO 4
                         GO TO 4
              303        DO 302 L = 1,RS
                         A = B(L,I)
                         B(L,I)=B(L,IM)
              302        B(L,IM)=A
                         L=P(I)
                         P(I)=P(IM)
                         P(IM)=L
                         I=I-M
                         IF(I,GT,1) GO TO 3
              4          J = J + 1
                         IF(J,GT,K) GO TO 1
                         GO TO 2
                         END
```

ORIGINAL PAGE IS
OF POOR QUALITY

## 20. [300,6] SUBSTR

See volume 1, section 3.5.2.10.8.1 for description. Listing is presented below.

```
MFØRTRAN IV-PLUS V02-04                  14:28:46    31-AUG-77            PAGE 1
SUBSTR.FTN             /TR:BLOCKS/WR
                       SUBRØUTINE SUBSTR(A,I,N,B,J,M)
                       IMPLICIT INTEGER (A-Z)
                       LØGICAL*1 A(1),B(1)
                       DATA BLANK/2H  /
                       IS=I
                       JS=J
                       L=0
                       IF(N .EQ. 0) GØ TØ 20
                       L=N
                       IF( L .GT. M ) L=M
                       DØ 10 K=1,L
                       B(JS)=A(IS)
                       IS=IS + 1
                       JS=JS + 1
            10         CØNTINUE
                       IF( N .GE. M) RETURN
            20         L=L + 1
                       DØ 30 K=L,M
                       B(JS)=BLANK
                       JS=JS+1
            30         CØNTINUE
                       RETURN
                       END
```

## 21. [300,6] THMLOP.FTN

**The subroutine performs the logical operation between two theme tracks and the results are output to the third theme.**

- **Calling sequence**

  CALL THMLOP

```
THMLØP.FTN      /TR:BLOCKS/WR
                SUBRØUTINE THMLØP
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C     LØAD FUNCTIØN THEME A& B PERFØRM AND, ØR, XØR, SUB, ØUTPUT TØ  C
          C         THEMEC                                                      C
          C         ****THMLØP.FTN  ****************************************   C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                IMPLICIT INTEGER(A-R),(S-Z)
                BYTE W,FUN,THMDEF
                DIMENSIØN W(74)   ,X(256,2),Y(256,2),THMDEF(4)
                THMDEF(1)='A'
                THMDEF(2)=1
                THMDEF(3)=2
                THMDEF(4)=3
             10 CØNTINUE
                WRITE (6,1000)
           1000 FØRMAT(1X,'SELECT LØGICAL ØPERATIØN')
                WRITE(6,1001) THMDEF(1)
           1001 FØRMAT('$','(A)DD,(S)UBTRACT,A(N)D,(E)XCLUSIVE ØR ',A1,1X,'>')
          C     READ ØPTIØN
                READ(6,1010) W
           1010 FØRMAT(74A1)
                CALL FRØNT (W,74)
                IF(W(1).EQ.'X' ) GØ TØ 450
                IF(W(1).EQ.'R' ) GØ TØ 10
                IF(W(1).EQ.' ') GØ TØ 150
                IF(W(1).EQ.'A'.ØR.W(1).EQ.'S'.ØR.W(1).EQ.'N'.ØR.W(1).EQ.'E')
               X GØ TØ 150
                GØ TØ 10
            150 CØNTINUE
                IF(W(1).EQ.'A') THMDEF(1)='A'
                IF(W(1).EQ.'S') THMDEF(1)='S'
                IF(W(1).EQ.'N') THMDEF(1)='N'
                IF(W(1).EQ.'E') THMDEF(1)='E'
                FUN=THMDEF(1)
                IF(FUN.EQ.'A') AFUN='ADD'
                IF(FUN.EQ.'S') AFUN='SUB'
                IF(FUN.EQ.'N') AFUN='AND'
                IF(FUN.EQ.'E') AFUN='EØR'
            100 CØNTINUE
                WRITE(6,1030) THMDEF(2),THMDEF(3)
           1030 FØRMAT('$SELECT INPUT THEMES           ',1X,I1,2X,I1,2X,'>')
          C     READ THEME A AND THEME B
                READ(6,1040) W
           1040 FØRMAT(74A1)
                CALL FRØNT(W,74)
                IF(W(1).EQ.'X' ) GØ TØ 450
                IF(W(1).EQ.'R') GØ TØ 10
                IF(W(1).EQ.' ') GØ TØ 250
                K=0
                CALL INTFF(K,W,74,THEMEA)
                CALL FRØNT(W,74)
                CALL INTFF(K,W,74,THEMEB)
                IF(THEMEA.LE.0.ØR.THEMEA.GT.8.ØR.THEMEB.LE.0.ØR.THEMEB.GT.8)
               X GØ TØ 102
                GØ TØ 104
            102 CØNTINUE
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
          WRITE(6,106)
    106   FØRMAT( 1X,'OUT ØF RANGE !!!')
          GØ TØ 100
    104   CØNTINUE
          THMDEF(2)=THEMEA
          THMDEF(3)=THEMEB
    250   THEMEA=THMDEF(2)
          THEMEB=THMDEF(3)
    300   CØNTINUE
          WRITE(6,1100) THMDEF(4)
   1100   FØRMAT('$SELECT ØUTPUT THEME:        ',1X,I1,1X,'>')
C         READ ØUTPUT THEME C
          READ(6,1120) W
   1120   FØRMAT(74A1)
          K=0
          CALL FRØNT(W,74)
          IF(W(1).EQ.'X' ) GØ TØ 450
          IF(W(1).EQ.'Q') GØ TØ 100
          IF(W(1).EQ.' ') GØ TØ 1121
          CALL INTEF(K,W,74,THEMEC)
          IF(THEMEC.LE.0.ØR.THEMEC.GT.8) GØ TØ 302
          GØ TØ 304
    302   CØNTINUE
          WRITE (6,306)
    306   FØRMAT(1X,'OUT ØF RANGE !!!')
          GØ TØ 300
    304   CØNTINUE
          THMDEF(4)=THEMEC
   1121   CØNTINUE
          THEMEC=THMDEF(4)
          WRITE(6,1122) THEMEA,AFUN,THEMEB,THEMEC
   1122   FØRMAT(1X,'THEME LØGICAL ØPERATIØN',/
         X1X,        'THEME',I2,2X,A3,2X,'THEME',I2,2X,'=','THEME',I2)
   1123   CØNTINUE
          WRITE(6,1124)
   1124   FØRMAT('$PRØCEED (Y)ES/(N)??   >')
          READ(6,1126) W
   1126   FØRMAT(74A1)
          CALL FRØNT(W,74)
          IF(W(1).EQ.'N') GØ TØ 10
          IF(W(1).EQ.'Y') GØ TØ 448
          GØ TØ 1123
    448   CØNTINUE
C         READ THEME A & THEME B INTØ X,Y AND CØMPUTE AND,ØR, XØR
C          SUBSTRACT THEN WRITE RESULT INTØ THEME C
          DØ 450 L1=0,32,32
          DØ 450 L2=0,1
          DØ 450 L3=0,511,64
          LINE=L1+L2+L3
          CALL IRT(THEMEA,LINE,16,X)
          CALL WAIT
          CALL IRT(THEMEB,LINE,16,Y)
          CALL WAIT
          IF(FUN.EQ.'A') GØ TØ 22
          IF(FUN.EQ.'S' )GØ TØ 25
          IF(FUN.EQ.'N') GØ TØ 20
```

```
      IF(FUN.EQ.'E') GØ TØ 24
   20 CØNTINUE
      DØ 28 K1=1,2
      DØ 28 K2=1,256
   28 X(K2,K1)=JAND(X(K2,K1),Y(K2,K1))
      GØ TØ 36
   22 CØNTINUE
      DØ 30 K1=1,2
      DØ 30 K2=1,256
   30 X(K2,K1)=IØR(X(K2,K1),Y(K2,K1))
      GØ TØ 36
   24 CØNTINUE
      DØ 32 K1=1,2
      DØ 32 K2=1,256
   32 X(K2,K1)=IEØR(X(K2,K1),Y(K2,K1))
      GØ TØ 36
   26 CØNTINUE
      DØ 34 K1=1,2
      DØ 34 K2=1,256
      Y(K2,K1)=ICØM(Y(K2,K1))
   34 X(K2,K1)=JAND(X(K2,K1),Y(K2,K1))
   36 CØNTINUE
      CALL INT(THEMEC,LINE,16,X)
      CALL WAIT
  450 CØNTINUE
      WRITE(6,1180)
 1180 FØRMAT(    '$(R)ESTART ØR E(X)IT >')
      READ(6,1200) W
 1200 FØRMAT (74A1)
      CALL FRØNT (W,74)
      IF(W(1).EQ.'R' ) GØ TØ 10
      IF(W(1).EQ.'X') GØ TØ 1220
      GØ TØ 450
 1220 CØNTINUE
      RETURN
      END
```

## 22. [131,140] TWRITE.FTN

### 22.1 ENTRY POINT - TWRITE

The subroutine TWRITE writes the clusters and categories assigned to themes on to the I-100. The assigned clusters and categories are passed to TWRITE by the common statement LOCOM2.

- Calling sequence

```
              SUBRØUTINE TWRITE
              IMPLICIT INTEGER (A-Z)
              DIMENSIØN CMASK(60),IX(512),IY(512),TX(512),TY(512)
              DIMENSIØN PIX(256,8)
              BYTE Y(4096), BUF(196)
              EQUIVALENCE (Y(1),PIX(1,1))
              CØMMØN /ZØØM/IC(4),TC(4),IX,IY,TX,TY,MX,MY
              CØMMØN /LØCOM2/CMASK
              IF(MY.GT.128) GØ TØ 5
              MY1=16
              N=MY/MY1
              IF(MY-N*MY1.NE.0) N=N+1
              GØ TØ 8
      5       CØNTINUE
              N=8
              MY1=MY/N
              IF(MY-N*MY1.NE.0) MY1=MY1+1
      8       CØNTINUE
              DØ 60 LL=1,MY1
                 L=LL
                 DØ 10 I=1,N
      C
      C                          READ DATA FRØM CHANNEL FIVE FØR PRØCESSING
      C
                    CALL IRV(5,IY(L),PIX(1,I))
                    L=L + MY1
                    IF(L.GT.MY) GØ TØ 15
      10            CØNTINUE
      15         IF(N.LE.4) CALL WAIT
                 L=LL
                 P512 = 0
                 DØ 40 I=1,N
      C
      C                          READ CLUSTER MAP FILE
      C
                    READ(6'TY(L)) BUF
      C
      C                          PRØCESS THE DATA FRØM CLUSTER MAP FILE
      C                          MANIPULATE THE THEME DISPLAY ACCØRDINGLY
      C
                    DØ 30 LINE=1,MX
                       IN  = IX(LINE)  + P512 + 1
                       M   = IBYTE(IN-1,Y)
                       K   =TX(LINE) -1
                       CLUSTR = IBYTE(K,BUF)
                       IF(CLUSTR.EQ.0) GØ TØ 30
                       Y(IN) = IØR(M,CMASK(CLUSTR))
      30               CØNTINUE
                    L = L + MY1
                    IF (L.GT.MY) GØ TØ 45
      C
      C                          INCREMENT PØINTER FØR Y(NY)
      C
                    P512 = P512 + 512
      40            CØNTINUE
      45         L=LL
```

```
FØRTRAN IV-PLUS V02-04          14:29:44     31-AUG-77            PAGE 2
TWRITE.FTN        /TR:BLØCKS/WR
        C
        C                               WRITE PRØCESSED DATA BACK TØ CHANNEL FIVE
        C                               FØR THEME DISPLAY ØF CLUSTERS
        C
                      DØ 50 I=1,N
                         CALL IWV(5,IY(L),PIX(1,I))
                         L = L+MY1
                         IF(L.GT.MY) GØ TØ 55
         50              CØNTINUE
         55           IF(N.LE.4) CALL WAIT
         60           CØNTINUE
                   RETURN
                   END
```

~~22-5~~

141

## 23. [300,6] VDALTR.FTN

The subroutine VDALTR erases the full (Channel 1-5) window which is in globe common (CØM 4).

- Calling sequence

CALL VDALTR (ML, MT, MR, MB, CH, IBUF, IND, FLAG)

| Argument | Type | Dimension | In/Out | Description |
|---|---|---|---|---|
| Input | | | | |
| ML | | | | |
| MT | coordinates of window | | | |
| MR | | | | |
| MB | | | | |
| CH; channel number to be erased | | | | |
| IBUF(1) = 0 | | | | |
| IND = 1 | | | | |
| FLAG (unused) | | | | |

```
MFØRTRAN IV-PLUS V02-04              14:30:07    31-AUG-77              PAGE 1
VDALTR.FTN          /TR:BLØCKS/WR
                    SUBRØUTINE VDALTR(XU,YU,YL,YL,CHNL,BUFF,IND,FLAG)
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C
          C
          C         ALTER VIDEØ SCREEN
          C
          C         FRØM CØØRDINATES XU,YU TØ XL,YL
          C
          C         BY THE CØNTENTS ØF THE BUFFER PASSED
          C
          C
          C
          CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
          C
          C
                    IMPLICIT INTEGER (A-Z)
                    BYTE TBUF(512),BUFF(1)
          C
          C
          C
                    IF (XU .LT. 0   .ØR.  XU .GT. 511)  GØ TØ 999
                    IF (XL .LT. 0   .ØR.  XL .GT. 511)  GØ TØ 999
                    IF (YU .LT. 0   .ØR.  YU .GT. 511)  GØ TØ 999
                    IF (YL .LT. 0   .ØR.  YL .GT. 511)  GØ TØ 999
          C
                    LNCT = XL - XU + 1
                    PXCT = YL - YU + 1
                    IF (LNCT .LT. 1   .ØR.  LNCT .GT. 512)  GØ TØ 999
                    IF (PXCT .LT. 1   .ØR.  PXCT .GT. 512)  GØ TØ 999
                    IF (CHNL .LT. 0   .ØR.  CHNL .GT.   5)  GØ TØ 999
          C
                    K = 1
                    CHST = CHNL
                    IF (CHNL .NE. 0)  GØ TØ 50
                    CHNL = 1
                    CHST = 5
          C
           50       DØ 100 I=YU,YL
                    K1 = K
          C
                    DØ 100 C=CHNL,CHST
                    CALL IRV(C,I,TBUF)
                    CALL WAIT
          C
                    K = K1
                    DØ 110 J=XU,XL
                    TBUF(J+1) = BUFF(1)
                    IF (IND .EQ. 1)  GØ TØ 105
                    TBUF(J+1) = BUFF(K)
           105      K = K + 1
           110      CØNTINUE
          C
                    CALL IWV(C,I,TBUF)
```

~~23-2~~
143

ORIGINAL PAGE IS
OF POOR QUALITY

```
FØRTRAN IV-PLUS V02-04          14:30:07    31-AUG-77           PAGE 2
VDALTR.FTN          /TR:BLØCKS/WR
0031                CALL WAIT
0032        100     CONTINUE
          C
          C
0033                FLAG = 0
0034        900     RETURN
          C
          C
0035        999     FLAG = 1
0036                GØ TØ 900
          C
0037                END
```

~~23-5~~
144

## 24. [300,6] WINDER.FTN

**The subroutine WINDER consists of two subroutine BLKTHM and VDALTR which erases the partial or full window.**

- **Calling sequence**

  **CALL WINDER**

```
WINDER,FTN      /TR:BLOCKS/WR
                SUBROUTINE WINDER
        CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        C       WINDOW ERASE                                             C
        C        ***WINDER,FTN*****                                      C
        CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
                IMPLICIT INTEGER(A-X)
                INCLUDE 'SY:[300,3]CAMSCOMON,INC'
       •        INCLUDE 'SY:[300,3]CAMSPARAM,INC'
       •        PARAMETER MAXCAT=60,MAXSUB=60,MAXCHN=4,NPIX=196,NLIN=117,MAXFLD=50
       •       1,MAXV=11,NDOTS=209,DLSKIP=10,DSSKIP=10,MAXACD=6,MAXACC=4,
       •       2NOSPWD=6,NDOTWD=10
       •        EQUIVALENCE (C1,ACDATE),(C2,ISEG),(C3,PFLAG),(C4,TX1),(C5,DISKID)
       •        INTEGER C1(469),C2(256),C3(71),C4(348),C5(629)
       •  C*
       •        INTEGER ACDATE,SUBCAT,SUBPOP,CATKNT,CATTH
       •        BYTE CHNVEC,NOCHAN,NOSUB,DOTCAT,DOTCLU
       •        COMMON/COM1/ACDATE(2,MAXACC),CHNVEC(MAXCHN,MAXACC),NOCHAN,NOSUB,
       •       1SUBCAT(MAXSUB),SUBPOP(MAXSUB),CATKNT(MAXCAT),CATTH(MAXCAT),NODO,
       •       2NODU,NOTH,DOTCAT(NDOTS),DOTCLU(NDOTS)
       •  C*
       •        INTEGER ADATES,SUNAZ,ANALST,FLDDAY,DOTDAY,PDATE1,TDATE1
       •        INTEGER PDATE2,TDATE2,PDATE3,TDATE3,CATNAM,DISKID,RANDOM,GRID
       •        BYTE DELFLG,NOACD,SPILGR,SUNEL,NSTART,NTYPE1,ALP,ALPO
       •        BYTE PCTCT,PCTCTO,VAR,VARO,DLABEL,TYPE
       •        COMMON/COM2/ISEG,DELFLG,NOACD,ADATES(2,MAXACD),SPILGR(MAXACD),
       •       1SUNEL(MAXACD),SUNAZ(MAXACD),IMDATE(2),ANALST(5),FLDDAY(2),
       •       2DOTDAY(2),NSTART,NTYPE1,PDATE1(2),TDATE1(2),PDATE2(2),TDATE2(2),
       •       3PDATE3(2),TDATE3(2),NOCAT,CATNAM(MAXCAT),ALP(MAXCAT),ALPO,
       •       4         PCTCT(MAXCAT),PCTCTO,VAR(MAXCAT),VARO
       •  C*
       •        INTEGER EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1,UFLAG2,UFLAG3,
       •       1UFLAG4
       •        INTEGER PFLAG,DSKMNT
       •        COMMON/COM3/PFLAG,DSKMNT,EFLAG1,EFLAG2,EFLAG3,EFLAG4,EFLAG5,UFLAG1
       •       1,UFLAG2,UFLAG3,UFLAG4,NEWLAB(MAXSUB)
       •  C*
       •        INTEGER TX1,TY1,TX2,TY2,ACDISP,G,B,DTWIND,DOTARY,GMIN,GMAX,FUL
       •        INTEGER SPWIND,CLAWND,CLUWND
       •        COMMON/COM4/TX1,TY1,TX2,TY2,IX1,IY1,IX2,IY2,ACDISP(2),III(4),G(4),
       •       1B(4),DTWIND(5,NDOTWD),SPWIND(5,NOSPWD),IMWIND(4),NUMDOT,
       •       2DOTARY(NDOTS),GMIN,GMAX,FUL(2,7),CLAWND(8),CLUWND(8)
       •        COMMON/COM5/DISKID,RANDOM(NDOTS),GRID(NDOTS),DLABEL(NDOTS),
       •       1TYPE(NDOTS),RECLOC
                BYTE IBUF
                BYTE W
                DIMENSION IXY(2,209)
                DIMENSION W(74),CURSOR(5)
                DIMENSION IBUF(1)
                CURSOR(1)=0
                CURSOR(2)=256
                CURSOR(3)=1
                CURSOR(4)=256
                CURSOR(5)=1
                CALL IAK(CURSOR)
          10    CONTINUE
                FLAGS=0
```

ORIGINAL PAGE IS
OF POOR QUALITY

~~24-2~~
146

```
              FLAGD=0
              WRITE(6,20)
           20 FØRMAT(1X,'SELECT WINDØW BY CURSØR AND ENTER "CR"',/
             X '$WHEN READY     >')
              READ(6,30) W
           30 FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1).EQ.'R') GØ TØ 10
              IF(W(1).EQ.'X') GØ TØ 60
              CALL TRK(CURSØR)
              DØ 31 I=1,NØSPWD
              IF(SPWIND(1,I).GT.0) GØ TØ 32
              GØ TØ 31
C
C             SEARCH FØR SPECTRAL PLØT WINDØW
C
           32 CØNTINUE
              IF(CURSØR(2).LT.SPWIND(2,I).ØR.CURSØR(2).GT.SPWIND(4,I)) GØ TØ 31
              IF(CURSØR(4).LT.SPWIND(3,I).ØR.CURSØR(4).GT.SPWIND(5,I)) GØ TØ 31
              FLAGS=1
              KREC=I
              ML=SPWIND(2,I)
              MT=SPWIND(3,I)
              MR=SPWIND(4,I)
              MB=SPWIND(5,I)
              GØ TØ 33
           31 CØNTINUE
              DØ 34 K=1,NØDTWD
              IF(DTWIND(1,K).EQ.1) GØ TØ 36
              GØ TØ 34
           36 CØNTINUE
C
C             SEARCH FØR DØT PLØT WINDØW
C
              IF(CURSØR(2).LT.DTWIND(2,K).ØR.CURSØR(2).GT.DTWIND(4,K)) GØ TØ 34
              IF(CURSØR(4).LT.DTWIND(3,K).ØR.CURSØR(4).GT.DTWIND(5,K)) GØ TØ 34
              FLAGD=1
              ML=DTWIND(2,K)
              MT=DTWIND(3,K)
              MR=DTWIND(4,K)
              MB=DTWIND(5,K)
              GØ TØ 33
          34  CØNTINUE
              WRITE(6,9400)
         9400 FØRMAT(1X,'CURSØR NØT IN THE WINDØW')
              GØ TØ 10
           33 CØNTINUE
           35 CØNTINUE
              WRITE(6,40)
          40  FØRMAT(1X, 'SELECT ØPTIØN FØR WINDØW ERASE.',/
             X '$(P)ARTIAL ØR (F)ULL. >')
              READ(6,50) W
           50 FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1).EQ.'R') GØ TØ 10
              IF(W(1).EQ.'X') GØ TØ 60
```

```
FØRTRAN IV-PLUS V02-04          14:30:26    31-AUG-77          PAGE 3
WINDER.FTN      /TR:BLØCKS/WR
              IF(W(1).EQ.'F') GØ TØ 70
              IF(W(1).EQ.'P') GØ TØ 80
              GØ TØ 35
         70 CØNTINUE
      C
      C       ERASE FULL WINDØW CHANNEL 1-5
      C
              IND=1
              IBUF(1)=0
              DØ 52 CH=1,5
              CALL VDALTR(ML,MT,MR,MB,CH,IBUF,IND,FLAG)
         52 CØNTINUE
              IF(FLAGS.EQ.1) SPWIN(   ,KREC)=0
              IF(FLAGD.EQ.1) DTWIND(.,K)=0
              ØPEN(UNIT=8,NAME='[300,1]SCATXY.TMP',TYPE='ØLD',ACCESS='DIRECT')
              DØ 54 K1=1,209
              DØ 54 K2=1,2
         54 IXY(K2,K1)=0
              IF(FLAGS.EQ.1) WRITE(8'KREC,ERR=56) IXY
              CLØSE(UNIT=8)
              GØ TØ 60
         80 CØNTINUE
              WRITE(6,100)
        100 FØRMAT( '$SELECT THEME TRACK NUMBER TØ BE ERASED, >')
              READ(6,110) W
        110 FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1).EQ.'B') GØ TØ 35
              IF(W(1).EQ.'X') GØ TØ 60
              K=0
              CALL INTFF(K,W,74,THEME)
              IF(THEME.LE.0.ØR.THEME.GT.8) GØ TØ 112
              GØ TØ 114
        112 CØNTINUE
              WRITE(6,116)
        116 FØRMAT(1X,'ØUT ØF RANGE !!!')
              GØ TØ 80
        114 CØNTINUE
      C
      C       PARTIAL ERASE WINDØW
      C
              IØP=1
              IBUF(1)=0
              CALL BLKTHM(ML,MT,MR,MB,THEME,IBUF,IØP)
         91 CØNTINUE
              WRITE(6,90)
         90 FØRMAT(1X,'MØRE THEME TRACKS TØ BE ERASED',/
             X'$PRØCEED (Y)ES/(N)Ø? >')
              READ(6,92) W
         92 FØRMAT(74A1)
              CALL FRØNT(W,74)
              IF(W(1).EQ.'N') GØ TØ 10
              IF(W(1).EQ.'Y') GØ TØ 80
              GØ TØ 91
         56 CØNTINUE
              WRITE(6,58) I
```

```
FØRTRAN IV-PLUS V02-04          14:30:26    31-AUG-77            PAGE 4
WINDER,FTN        /TR:BLØCKS/WR
      58 FØRMAT(1X,'DISK I/Ø ERRØR ENCØUNTERED WHEN ERASING RECØRD NØ ',I2,
        X / 1X,'ØF THE SPECTRAL PLØT SCREEN CØØRDINATE FILE !!!')
      60 CØNTINUE
         WRITE(6,180)
     180 FØRMAT('$',    'E(R)ASE ANØTHER WINDØW ØR E(X)IT >')
         READ(6,190) W
     190 FØRMAT(74A1)
         CALL FRØNT(W,74)
         IF(W(1).EQ.'R') GØ TØ 10
         IF(W(1).EQ.'X') GØ TØ 200
         GØ TØ 60
     200 CØNTINUE
         RETURN
         END
```

## 25. ZOOOM

This program is described in reference 1.

```
HFØRTRAN IV-PLUS V02-04                    14:31:07    31-AUG-77              PAGE 1
Z0000M.FTN          /TR:BLOCKS/WR
              SUBROUTINE Z000ØM(IX1,IY1,IX2,IY2,TX1,TY1,TX2,TY2,IX,IY,TX,TY,XZ,Y
           1 Z,MX,MY,NX)
              IMPLICIT INTEGER (A-Z)
              CØMMØN /FATAL/Z0,RR
                CØMMØN /IØLP/IO,LP
              REAL XZ,YZ
              INTEGER IX(512),IY(512),TX(512),TY(512)
              Z0 = 0
              IF(IX1-IX2)1,2,3
      1       Q1=1
              MX=IX2-IX1+1
              GØ TØ 4
      3       Q1 = -1
              MX = IX1-IX2+1
      4       IF(TX1-TX2)5,2,6
      5       Q2=1
              TB = TX1
              NX = TX2-TX1+1
              GØ TØ 7
      6       Q2=-1
              TB = TX2
              NX=TX1-TX2+1
      7       XZ=Q1*NX/FLØAT(IX2-IX1+Q2)
              DØ 8 I = 1,MX
              IX(I)=IX1+Q1*(I-1)
      8       TX(I)=TX1+Q1*(I-1)/XZ
      9       IF(TY1-TY2)10,2,11
     10       TB=TY1
              IB=IY1
              Q4=1
              TYD=TY2-TY1+1
                GØ TØ 12
     11       TB = TY2
              IB = IY2
              Q4=-1
              TYD= TY1-TY2+1
     12         CØNTINUE
                IF(IY1-IY2)13,2,14
     13         CØNTINUE
                J=IY2-IY1+1
                MY = J
              GØ TØ 15
     14         CØNTINUE
                MY = IY1-IY2+1
              Q4=-Q4
     15       YZ = FLØAT(MY)/FLØAT(TYD)
              DØ 16 I = 1,MY
              IY(I)=IB+Q4*(I-1)
     16       TY(I)=TB+(I-1)/YZ
              RETURN
      2         CØNTINUE
                WRITE(IO,1000)IX1,IY1,IX2,IY2,TX1,TY1,TX2,TY2
   1000       FØRMAT(' INPUT CØØRDINATE ERRØR IN Z000ØM','/' ',8I10)
              Z0 = 1
              RETURN
              END
```

ORIGINAL PAGE IS
OF POOR QUALITY

## 26. REFERENCE

1. Kell, T.: "As-Built" Design Specification for the I-100 Tape Read Consolidation Program (FULOI), LEC-9925, JSC-11848, December 1976.

## APPENDIX A

## DOCUMENTATION ON FSTVID, ERRMES, AND MODEF

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND 20771

REPLY TO
ATTN OF: Code 933

[illegible]
DEC 1 3 1976

Mr. Jesse L. Kersh
NASA/Johnson Space Center
Code TF121
Houston, Texas 77058

Dear Mr. Kersh:

The enclosed magnetic tape contains the FSTVID disk and tape I/O package that was requested by Ted Kell. The tape is in RSX-11D PIP format at 1600 bpi and is labeled 'FSTVID'. Three files are contained on the tape:

| | |
|---|---|
| FSTVID.MAC | Contains the FSTVID package. |
| ERRMES.MAC | Contains error message subroutines required by FSTVID. |
| MODEF.MAC | Is a global symbol definition subroutine which must be included in tasks having no Fortran modules. |

Documentation on the use of this package is also enclosed.

Sincerely,

John T Dalton

John T. Dalton
Computer Systems Branch

Enclosures

```
.IRECTORY M00:
.-DEC-76 13:25

.IVID.MAC;74        66.       10-DEC-76 00:00
:RMES.MAC;33        19.       10-DEC-76 00:00
.DEF.MAC;1          1.        10-DEC-76 00:00

      TOTAL OF 86. BLOCKS IN 3. FILES
```

ORIGINAL PAGE IS
OF POOR QUALITY

155

# FSTVID

## A Disk and Tape I/O Support Package

## for RSX-11D

John T. Dalton
Code 933

February 26, 1976

~~A-5~~ 156

## ABSTRACT

This document describes FSTVID, a package of subroutines, for performing efficient disk and tape I/O under the RSX-11D Version 6a operating system* on PDP-11 computers. While the File Control Services (FCS) of RSX-11D provide many capabilities, they also impose severe restrictions due to block size limitations and overhead. FSTVID was developed to bypass the actual reading and writing functions of FCS and thus remove some of these limitations. FSTVID interfaces with FCS to allocate, delete, and extend files, thus maintaining compatibility with RSX-11D file structures.

(The name FSTVID denotes "Fast Video". The original motivation for this package was a need for rapid transfer of image data between a video display and peripheral devices.)

* While this package was developed under Version 6a of RSX-11D, it should be useable under other versions with little or no modification.

**Basic Functions:**

The following entry points are provided by FSTVID:

FVOPEN - opens a disk or tape file

FVREAD - reads a record from disk or tape

FVWRIT - writes a record to disk or tape

FVWAIT - waits for completion of the previous read or write operation

FVCLOS - closes a disk or tape file

FVDLTE - deletes an open disk file

FVDSET - sets the start block number for the next read or write

FVRWND - rewinds a tape or sets the start block to 1 for the next read or write operation on disk

PRSFNM - parses a file name and sets up a Data Set Descriptor (called by FVOPEN)

Files are referenced by Logical Unit Number (LUN). Assignment of a LUN to a tape drive or to a disk file is determined by the device/file name specified in the FVOPEN argument list. LUN values 1 through 16 are supported.

Each open disk file requires a File Descriptor Block (FDB). In order to minimize core requirements, four FDB's have been coded into FSTVID. (If more than 16 LUN's or more than 4 open disk files are required, these limits may be increased by editing the values of the symbols "NLUNS" and "NFDBS" in FSTVID and reassembling the package).

Data records in disk files under RSX-11D are accessed by Virtual Block Number (VBN), with the first virtual block in a file corresponding to VBN 1. Each virtual block corresponds to a logical (physical) block on the disk and consists of 256 words (512 bytes). In order to allow multiple block transfers in a single disk access, the following approach was taken in the implementation of this package:

FSTVID maintains a pointer (NXTREC) for each LUN assigned to a disk device. This pointer indicates the start VBN for the next access and is initialized to 1 when the file is opened. When FVREAD or FVWRIT is called for a disk file, the pointer is updated after the data transfer by the number of disk blocks required to complete the transfer. (For example, writing 514 bytes would update the pointer by two blocks.) The effect is that a sequential access method is supported. (Note that generation of records that are not a multiple of 512 bytes results in unused disk space.)

Data may be accessed in a non-sequential manner by calling FVDSET to alter the NXTREC pointer, thus causing the next FVREAD or FVWRIT operation to begin at the specified Virtual Block Number.

For example, assume that a group of 512 byte data records exists on disk file. These records may be accessed in several ways:

(1) sequentially by successive FVREAD's of 512 bytes;

(2) sequentially in pairs by successive FVREAD's of 1024 bytes;

(3) sequentially in groups of N records by successive FVREAD's of N X 512 bytes;

(4) record M by calling FVDSET with an argument of M followed by calling FVREAD for 512 bytes;

(5) N sequential records beginning with record M by calling FVDSET with an argument of M followed by calling FVREAD for N X 512 bytes.

The following section describes the calling sequences for each of the entry points.

<u>Subroutine Decriptions and Calling Sequences</u>:

<u>FVOPEN</u>

This subroutine initializes the package to access a tape or disk file:

```
    CALL FVOPEN(ITYPE,LUN,FILE,NC,ISTAT,IEVFLG[,NBLKS])
where
    ITYPE = Type of access required
          = 1 to read an existing file,
          = 2 to write (create) a new file,
```

4.

= 3 to modify an existing disk file (no extension of the file permitted)

= 4 to update an existing disk file (the file is extended if necessary by FVWRIT)

= 5 if the file specified in FILE exists, it is opened for update as in ITYPE = 4. If the file specified in FILE does not exist, it is created as in ITYPE = 2.

| | |
|---|---|
| LUN = | Logical Unit Number to be assigned to the disk file or tape drive. |
| FILE = | File name string or array containing a file name string (RSX-11D convention). |
| NC = | Number of Characters in the file name string. |
| ISTAT = | Word in which a status code is returned. |
| IEVFLG = | Event flag number to use for I/O synchronization. |
| NBLKS = | An argument specifying the number of physical disk blocks to allocate for the file (used only if ITYPE = 2 and FILE specifies a disk device). |

File name strings are of the form ddn:[uic]file.type;ver where

| | |
|---|---|
| dd | is a two character device designator |
| n | is a 1 or 2 digital octal unit number (must be present if dd is present) |
| [uic] | is the User Identification Code specifying the directory on the specified device in which the file is located |

| | |
|---|---|
| file | is the file name (up to 9 characters) |
| typ | is a file type designator (up to 3 characters) |
| ver | is an octal version number |

If dd = MT or MM, the LUN is assigned to the specified magnetic tape device; and the [uic],file,typ, and ver portions of the file name are ignored; otherwise the device is assumed to be disk. If "ddn:" is omitted, the device used is that to which the specified LUN is currently assigned. If the LUN is not assigned, SYØ: is used by default.

If [uic] is omitted, the UIC used is that under which the task is running.

If file is omitted, FSTVID is used by default.

If typ is omitted, IMA is used by default.

If ver is omitted and a file is being read, modified, or updated, the latest version of the file is opened. If ver is omitted and a file is being created, the version number of the new file is one greater than the highest current version number. If ver is specified for a file being created and the file already exists with that version number, the old file is superseded by the one being created.

If a file is being created on disk and NBLKS is specified, an attempt is made to allocate NBLKS contiguous blocks. If this fails, a non-contiguous file allocation is attempted. If this fails, an error is returned from FCS.

The most efficient disk transfer is a multiple block transfer to or from a contiguous file. If a file must be extended (due to NBLKS not being specified or not being large enough when the file is created), the file will probably not be contiguous. Therefore, it is recommended that NBLKS be specified and be large enough to contain the file. Of course, if NBLKS is larger than the number of blocks actually written in the file, the difference will be wasted disk space.

When FILE specifies a magnetic tape device, the LUN is assigned to the specified device. The tape is not positioned. File positioning may be performed using FVRWND and FVCLOS.

## FVOPEN Error Processing

The following values are returned in ISTAT:

0 = Successful open

-1 = File name syntax error returned from PRSFNM

-2 = File is already open for specified LUN

-3 = All FDB's are in use

-4 = Specified LUN is too large

-5 = An error was returned from FCS during the OPEN process

Errors -1, -2, -3, and -4 cause error messages to be printed on the tasks "TI" using the Message Output (MO) device handler. Operation of the task continues.

In the case of an error return from FCS, the FCSERR subroutine is called. A message describing the error is displayed on the tasks "TI" and the task is suspended. The task may be continued by typing:

```
CON "task name"
```

to MCR. The task may also be aborted at this time. If the task is continued, control then returns to the calling program.

## FVREAD/FVWRIT

These subroutines read/write a specified number of bytes from/to the file or device assigned to the specified LUN:

```
CALL FVREAD (LUN,BUFFER,NBYTES)
CALL FVWRIT (LUN,BUFFER,NBYTES)
```

where

| | |
|---|---|
| LUN = | the Logical Unit Number assigned to the input/output file or device by a previous FVOPEN call, |
| BUFFER = | array into/from which the record is to be read/written, |
| NBYTES = | the number of bytes to be read/written (must be even and non-zero) |

This subroutine is asynchronous in that control is returned to the calling program immediately after the I/O request is queued. Before the input record is processed (or output buffer area is changed), FVWAIT must be called to insure that the transfer has completed.

## I/O Error Processing

If an error results from a call to FVREAD or FVWRIT, subroutine IOERR is called to generate an error message on the task's "TI". If an end-of-file is detected on the file or device, task execution continues. Other errors cause task suspension. The task may be continued by typing

```
CON "task name"
```

to MCR.

Status of the I/O may be tested by the calling program (see FVWAIT).

## FVWAIT

This subroutine waits for completion of the previous I/O request on the specified LUN:

```
CALL FVWAIT (LUN[,IOST])
```

where

LUN = Logical Unit Number to wait for,

IOST = (optional) two-word array into which the contents of the I/O status block are returned:

```
Word 1 - Byte 0 = I/O Status Code
         Byte 1 = Unused
Word 2 = Total bytes transferred
```

I/O status codes are defined in Appendix A of "RSX-11D Executive Reference Manual".

FVCLOS

This subroutine closes a file opened by FVOPEN:

CALL FVCLOS(LUN[,IRWFLG])

where

LUN = Logical Unit Number of file to be closed,

IRWFLG = (optional) rewind flag

If the LUN is assigned to a disk file, the FCS CLOSE macro (CLOSE$) is issued for the LUN and the FDB is freed. If an error is returned from FCS, a message is printed on the task's "TI" and the task is suspended. It may be resumed by typing

CON "task name"

to MCR.

If the LUN is assigned to a magnetic tape device, the following occurs:

(1) If the tape is opened for input and the last operation did not encounter an end-of-file and the tape is not rewinding or at the load point, the tape is spaced past the next file mark.

(2) If the tape is opened for output, an end-of-file is written.

(3) If the second argument (IRWFLG) is present and greater than 0, the tape is rewound.

(4) The LUN is deassigned. After a LUN is used for magnetic tape, the default LUN assignment specified at Task Build no longer applies.

FVDLTE

This subroutine deletes an open disk file:

CALL FVDLTE(LUN,ISTAT)

where

LUN = Logical Unit Number of the open disk file to be deleted.

ISTAT = Word into which a status code is returned.

= 0 for successful deletion

= -1 if the LUN did not specify an open disk file

= -2 if an error was returned from the delete request to FCS

If an error is returned from FCS, FCSERR is called to display a message on the task's "TI" and the task is suspended. It may be continued by typing

CON "task name"

to MCR.

11 .

<u>FVDSET</u> (Disk files only)

This subroutine sets the number of the first virtual block to be read or written by the next call to FVREAD or FVWRIT:

```
CALL FVDSET(LUN,IVBN)
```

where

LUN = the Logical Unit Number of the disk file for which the "next Virtual Block Number" is to be changed

IVBN = the start Virtual Block Number for the next access (first VBN in a file = 1).

Since Virtual Blocks are 256 words (512 bytes), a program writing or reading logical records larger than 256 words must translate its logical record numbers into virtual block numbers as follows:

$$VBN = (LBN-1) \times N + 1$$

where

VBN = Virtual Block Number in file,

LRN = Number of the Logical Record to be accessed,

N = Number of 256 word Virtual Blocks required to contain one logical record (as defined by the calling program)

FVRWND

This subroutine rewinds a disk file or tape:

```
CALL FVRWND(LUN)
```

where

LUN = Logical Unit Number of the tape or disk file to be rewound.

If LUN is assigned to a magnetic tape device, the tape is rewound.

If LUN is assigned to a disk file, the Virtual Block Number for the next access is set to 1 (equivalent to CALL FVDSET(LUN,1)).

PRSFNM

This subroutine parses a file name and sets up a Data Set Descriptor:

```
CALL PRSFNM(FILE,NCHAR,DSDESC,ERROR[,LUN])
```

where

FILE = File name string or an array containing a file name string (RSX-11D convention),

NCHAR = The number of characters in the file name string.

DSDESC = A 6-word array which will contain a Data Set Descriptor for the file upon return. Data Set Descriptors are described in "RSX-11D I/O Operations Reference Manual".

ERROR = Status word. This is set to -1 if a syntax error is detected, 0 otherwise.

LUN = (optional) Logical Unit Number. If present and FILE specifies a device name, LUN is assigned to that device.

PRSFNM performs the minimal amount of error checking required to set up a Data Set Descriptor. An error is returned if the device name/number is more than 5 characters (including ":") or if a "[" is found wihtout a "]" in the UIC. Other errors will be detected by FCS when the file is opened.

## Magnetic Tape Positioning

FVCLOS positions a tape past the next end-of-file mark when necessary to ensure that a subsequent FVOPEN does not cause reading or writing in the middle of a file. While this feature may be used to effect positioning of a tape to a desired file, it is generally easier to use the Fortran special subroutines for issuing "Queue I/O" directives.

For example, a tape may be positioned to the start of file N by the following sequence:

```
DIMENSION IBUF(...),IOST(2)
CALL FVOPEN(1,1,'MTO:',4,ISTAT,1)
CALL FVRWND(1)
IF (N.EQ.1) GO TO 20
```

```
      N1 = N-1
      DO 10 I=1,N1
      CALL FVREAD (1,IBUF,NBYTES)
      CALL FVWAIT(1,IOST)
      CALL FVCLOS(1)     ! CAUSES SPACE PAST EOF
      CALL FVOPEN(1,1,'MTO:',4,ISTAT,1)
   10 CONTINUE
   20 CONTINUE
```

or by the following sequence:

```
      DIMENSION IOST(2),IPRM(6)
      DATA ISPR/"2440/    !I/O FUNCTION CODE IO.SPF
      CALL FVOPEN (1,1,'MTO:',4,ISTAT,1)
      CALL FVRWND(1)
      IPRM(1)=N-1
      CALL WTQIO(ISPF,1,1,,IOST,IPRM)
```

## Linking FSTVID into a Task

### AOIPS PDP-11/70

FSTVID is contained in object file SY0:[1,300]FSTVID.OBJ on the AOIPS PDP-11/70.

The FCS, I/O and directive error message subroutines FCSERR, IOERR, and DIRERR are also required (see AOIPS System Manager's Bulletin, Number 19). These are found in

```
SYO:[1,300]ERRMES.OBJ
```

on the 11/70. If the task using these routines has no Fortran modules, file

```
SYO:[1,300]MODEF.OBJ
```

must also be included as input to the Task Builder.

<u>AOIPS PDP-11/45 (IMAGE 100)</u>

FSTVID is not available on the 11/45 due to limited space on the system disk. The above object files should be obtained from the 11/70 and transferred to the user's disk on the 11/45.

# ERROR MESSAGE SUBROUTINES FOR RSX-11D DIRECTIVE, I/O, AND FILE CONTROL SERVICES (FCS) ERRORS

JOHN T. DALTON
NASA/GSFC CODE 933
NOVEMBER 10, 1975

THROUGH THE MESSAGE OUTPUT (MO) HANDLER, RSX-11D PROVIDES A CONVENIENT MEANS OF GENERATING FORMATTED MESSAGES FROM MACRO-11 PROGRAMS. IN ADDITION, FILE [1,2]QIOSYM.MSG CONTAINS MESSAGES FOR EACH OF THE DIRECTIVE AND I/O ERROR CODES RETURNED BY THE SYSTEM.

IN ORDER TO PROVIDE A SIMPLE MECHANISM FOR THE GENERATION OF ERROR MESSAGES, A PACKAGE OF SUBROUTINES HAS BEEN CREATED IN [1,300]ERRMES WHICH UTILIZE THIS FACILITY. THE FOLLOWING ENTRY POINTS ARE PROVIDED:

## DIRERR

THIS SUBROUTINE PRINTS MESSAGES FOR RSX-11D DIRECTIVE ERRORS THE MESSAGES ARE OF THE FORM:

```
*** <TASK NAME> - SUSPENDED
DIRECTIVE ERROR -- PC = XXXXXX
<ERROR MESSAGE TEXT>
```

WHERE XXXXXX = THE VALUE OF THE PROGRAM COUNTER (PC) AT THE POINT WHERE DIRERR WAS CALLED.

NO ARGUMENTS ARE REQUIRED BY THIS SUBROUTINE. ERROR MESSAGES WILL BE GENERATED AT THE TASK'S "TI" BY SIMPLY PROVIDING THE SUBROUTINE NAME "DIRERR" AS THE OPTIONAL FINAL ARGUMENT IN DIRECTIVE CALLS AND INCLUDING [1,300]ERRMES AS INPUT TO THE TASK BUILD.

EXAMPLE 1. USE OF DIRERR

```
        WTSE$S  EFN,DIRERR      ; WAIT FOR EVENT FLAG NUMBER SPECIFIED
                                ; AT LOCATION "EFN". CALL DIRERR IN CASE
                                ; OF ERROR.
```

## IOERR

THIS SUBROUTINE PRINTS MESSAGES FOR I/O ERRORS ON THE TASK'S "TI". THE ADDRESS OF THE I/O STATUS BLOCK MUST BE PROVIDED AS THE ONLY ARGUM (FORTRAN LINKAGE CONVENTIONS ARE USED). MESSAGES ARE OF THE FORM:

```
        **** <TASK NAME> - SUSPENDED   ("CONTINUED" IF THE ERROR IS END OF F

        <ERROR MESSAGE TEXT>
        PC=AAAAAA
        I/O STATUS BLOCK: A,B C (D)8

WHERE A = BYTE 1 OF THE FIRST WORD OF THE I/O STATUS BLOCK IN OCTAL,
      B = BYTE 0 OF THE FIRST WORD (ERROR CODE) IN DECIMAL,
      C = WORD 2 OF THE I/O STATUS BLOCK IN DECIMAL,
      D = WORD 2 OF THE I/O STATUS BLOCK IN OCTAL.
```

EXAMPLE 2. USE OF DIRERR AND IOERR

ORIGINAL PAGE IS
OF POOR QUALITY

```
ERRARG: .BYTE   1,0             ; ARGUMENT LIST FOR IOERR
        .WORD   0
IOST:   .WORD   0,0             ; I/O STATUS BLOCK
          .
          .
          .
        QIOW$S  #IO.ATT,LUN,EFN,PRI,#IOST,#AST,<0>,DIRERR
;    DIRERR WILL BE CALLED IF AN ERROR OCCURS DURING ISSUANCE  OF THE QIOW$S.
        TSTB    IOST            ; I/O COMPLETED - TEST ERROR CODE.
        BGE     NOERR           ; BRANCH IF NO ERROR.
        MOV     R5,-(SP)        ;SAVE R5 ON STACK.
        MOV     #IOST,ERRARG+2  ;MOVE @(I/O STATUS BLOCK) TO ARG LIST.
        MOV     #ERRARG,R5      ;R5 POINTS TO ARG LIST FOR IOERR
        JSR     PC,IOERR
        MOV     (SP)+,R5        ;RESTORE R5 FROM STACK.
          .
          .
          .
        <CODE TO BE EXECUTED IF TASK CONTINUES.>
          .
          .
          .
NOERR:  -----   ------          ; BRANCH HERE IF NO ERROR.
```

## FCSERR

THIS SUBROUTINE PRINTS MESSAGES ON THE TASK'S "TI" FOR FILE CONTROL SERVICES (FCS) ERRORS WHICH ARE, IN FACT, EITHER DIRECTIVE OR I/O ERRORS. THE ADDRESS OF THE FILE DESCRIPTOR BLOCK (FDB) FOR THE FILE ON WHICH THE ERROR OCCURRED IS OBTAINED FROM R0.

MESSAGES ARE OF THE FORM:

```
        **** <TASK NAME> - SUSPENDED
        <ERROR MESSAGE TEXT>
        FCS ERROR: PC = XXXXXX
        <F.ERR> <F.ERR+1> <FILENAME>, LUN=<LUN>
        AAAAAA BBBBBB
```

WHERE F.ERR AND F.ERR+1 ARE ERROR CODES OBTAINED FROM THE FDB (SEE THE RSX-11D I/O OPERATIONS REFERENCE MANUAL).

AAAAAA AND BBBBBB ARE THE TWO WORDS OF THE I/O STATUS BLOCK IN OCTAL AND ARE ONLY PRINTED IF THE I/O STATUS BLOCK ADDRESS IS PROVIDED IN THE FDB.

EXAMPLE 3. USE OF FCSERR

```
        OPEN$W  #FDB,LUN,DSPT,RACC,URBA,URBS,FCSERR

                        OR

        OPEN$W  #FDB                 ; FDB HAS BEEN COMPLETELY INITIALIZED.
        BCC     10$                  ; BRANCH IF NO ERROR.
        JSR     PC,FCSERR            ; ERROR - CALL FCSERR. R0 CONTAINS
                                     ; ADDRESS OF FDB.
10$:
```

NOTE:
----

1. ANY REGISTERS USED BY THE ABOVE SUBROUTINES ARE SAVED ON ENTRY
   AND RESTORED BEFORE RETURNING WITH THE FOLLOWING EXCEPTIONS:
   IOERR REQUIRES THE ADDRESS OF THE I/O STATUS BLOCK IN A
   FORTRAN-TYPE ARGUMENT LIST POINTED TO BY REGISTER R5.
   FCSERR REQUIRES THE ADDRESS OF THE FDB IN R0 (FCS CONVENTION)

2. DEFINITION OF THE GLOBAL SYMBOL ".MOLUN" BY A TASK CAUSES THE TASK
   BUILDER TO INITIALIZE AN ADDITIONAL LOGICAL UNIT NUMBER (LUN)
   BY STORING THE FIRST UNUSED LUN IN .MOLUN AND ASSIGNING IT TO THE
   MO PSEUDO-DEVICE. SINCE FORTRAN USES MO, .MOLUN MUST NOT
   BE GLOBALLY DEFINED BY A MACRO-11 SUBROUTINE INCORPORATED IN A
   TASK CONTAINING FORTRAN-GENERATED CODE. THEREFORE, WHILE .MOLUN
   IS ASSUMED TO CONTAIN THE MO LUN IN THIS PACKAGE, .MOLUN IS NOT
   GLOBALLY DEFINED TO PREVENT INTERFERENCE WITH FORTRAN. IF THE
   ERRMES SUBROUTINES ARE BUILT INTO A TASK WITH NO FORTRAN MODULES,
   THE SYMBOL .MOLUN MUST BE GLOBALLY DEFINED ELSEWHERE. THIS
   MAY BE ACCOMPLISHED BY INCLUDING [1,300]MODEF AS INPUT TO THE
   TASK BUILDER.

EXAMPLE 4. BUILDING A TASK TO USE ERRMES SUBROUTINES.

IF FORTRAN-GENERATED MODULES ARE PRESENT:

```
MCR>TKB USER1,LP:/SH=USER1,USER2,...,USERN,[1,300]ERRMES
```

IF NO FORTRAN-GENERATED MODULES ARE PRESENT:

```
MCR>TKB USER1,LP:/SH=USER1,USER2,...,USERN,[1,300]ERRMES,[1,300]MODEF
```

3. THE MO HANDLER IS DOCUMENTED IN CHAPTER 11 OF THE "RSX-11D
   DEVICE HANDLERS REFERENCE MANUAL".

4. WHEN SENDING MESSAGES TO THE TASK'S "TI", THE ERRMES SUBROUTINES
   REQUEST SUSPENSION OF THE TASK THROUGH THE MO HANDLER. (THE
   ONE EXCEPTION IS IN THE CASE OF AN END-OF-FILE MESSAGE FROM
   IOERR, WHEN THE TASK IS CONTINUED.) WHETHER THE TASK IS
   SUSPENDED OR CONTINUES IS INDICATED BY THE MESSAGE. THE
   TASK MAY THEN EITHER BE ABORTED (VIA THE "ABO" MCR COMMAND)
   OR CONTINUED. SINCE THE TASK IS NOT IN A STATE OF TRUE
   SUSPENSION, THE MCR "RESUME" COMMAND WILL NOT EFFECT
   CONTINUATION OF EXECUTION. RATHER, THE TASK MUST BE RESUMED
   BY TYPING:

```
CON "TASK NAME"
```

   TO MCR.

## APPENDIX B

## DOCUMENTATION ON LECTAP

## LECTAP.MAC

LECTAP Fortran Compatible Mag Tape Subroutines

CALL TINIT (UNIT,MTXTFG,DRIVE)

where: UNIT - LOGICAL unit number to be assigned.
MTXTFG - 0=MT, 1=XT
DRIVE - PHYSICAL unit number

Initializes the linkage between the tape driver and dataset.

CALL TATCH (UNIT)

where: UNIT - LOGICAL unit number.

Attaches LUN

CALL TRLSE (UNIT)

where: UNIT - LOGICAL unit number.

Removes the linkage between the tape driver and a dataset.

CALL TWAIT (UNIT)

where: UNIT - LOGICAL unit number.

Waits for completion of process on dataset.

CALL TUNLD (UNIT)

where: UNIT - LOGICAL unit number.

Causes requested mag tape to be rewound to UNLOAD and Select Remote status to go off.

CALL TEOF (UNIT)

where: UNIT - LOGICAL unit number.

Writes an end-of-file record on the mag tape.

CALL TRWD (UNIT)

where: UNIT - LOGICAL unit number.

Causes requested mag tape to be rewound to the beginning-of-tape-marker.

CALL TREAD (UNIT,BUFF,BWC)

where: UNIT - LOGICAL unit number.
BUFF - buffer to read data into.
BWC - size of buffer in words.

Reads the next record in the dataset into the buffer. Returns actual number of bytes read in common status in word 2.

CALL TWRIT (UNIT, BUFF, BWC)

where: arguments are described under MTREAD.

Writes the next record in the dataset from the designated buffer.

CALL TFILE (UNIT, FILES)

where: UNIT - LOGICAL unit number.
FILES - number of files to skip.
+ = forward, - = backward

Skips forward/backward over requested number of EOF marks.

CALL TSET (UNIT, DENSITY, PARITY)

where: UNIT - LOGICAL unit number.
DENSITY - desired density.
PARITY - desired parity.

This request is ignored for 9-track tapes; it sets density and parity as follows for 7-track tapes.

| Density | Parity |
|---|---|
| 0 = 200 bpi | 0 = Odd |
| 1 = 556 bpi | 1 - Even |
| 2 = 800 bpi | |
| 3 = 800 bpi Dump Mode | |

CALL TSTAT (UNIT, FNCT, RESIDU)

where: UNIT - LOGICAL unit number.
FNCT - status returned.
RESIDU - residue count returned.

Returns the current status of the tape unit and a residue count.

Format of the status word is:

| Bit | Content |
|---|---|
| 0-2 | Last command was: |
| | 0 = offline |
| | 1 = read |
| | 2 = write |
| | 3 = write EOF |
| | 4 = rewind |
| | 5 = skip record |
| | 6 = backspace record |
| | 7 = unload |
| 3-6 | Unused |
| 7 | 1 = tape after EOF |
| 8 | 1 = tape at BOT (100,000,000) |
| 9 | 1 = tape after EOT |
| 1 | 1 = write lock on |
| 11 | Parity |
| | 0 = Odd |
| | 1 = Even |
| 12 | 0 = 9-track, 1 = 7-track |

| Bit | Content |
|---|---|
| 13-14 | Density:<br>0 = 200 bpi<br>1 = 556 bpi<br>2 = 800 bpi |
| 15 | 1 = Command Caused Error |

COMMON BLOCK

Name - Status

Word 1 and 2 - Drive status as returned by I/O handler. See RSX-11D device handlers reference manual (#DEC-11-OXDHA-B-D).

## INDEX TO ALL VOLUMES

The following index lists all computer programs and subroutines found in the text and printouts, and the variables listed in the text. The first number of each description is the volume number, and the remaining number refers to the section in that volume. A preceding L indicates the position of a listing. Therefore, 1-3.3.1 indicates a reference in section 3.3.1 of volume 1, and L2-14.6 indicates that a listing can be found in section 14.6 of volume 2. The list of programs and subroutines is definitive. The list of variables is not complete since those not mentioned in text are not included.

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| A | | VARIABLE | 1 | | 1-3.5.1.2 | 1-3.5.1.6 | 1-3.5.2.6 | 1-3.5.2.X | 3-1.1 | |
| | | | 2 | | 3-9. | | | | | |
| ACAT | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| ACDAT | COM1 | VARIABLE | 1 | | 1-4...9 | | | | | |
| ACDATE | | VARIABLE | 1 | | 1-3.5.1.4 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.9 | | |
| ACDISP | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | | | |
| ACLLAP | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 1-3.5.2.6 | 2-14. | | L2-14. |
| ADATES | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.9 | |
| | | | 2 | | 1-3.5.2.X | | | | | |
| ADSK | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| AID | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| AJUL | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| ALABEL | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| ALLUPD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.3 | | | | | L2-3.7 |
| ALP | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| ALP0 | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| ALPTAB | PRIVT | SUBROUTINE | 1 | | | | | | | L2-18.4 |
| ALSORT | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.6 | | | | | L2-14.4 |
| ANACAT | | VARIABLE | 1 | | 3-4.4 | | | | | |
| ANALYST | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| ANCL | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| AQLFST | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-21.1 |
| ARAND | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| ARIND | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| ARRY | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| ASEG | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| ASNLUN | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | | | | | |
| ASSIGN | F4PLIB | SUBROUTINE | 1 | | 1-3.5.2.6 | 1-3.5.2.8 | | | | |
| ATTACH | F4PLIB | SUBROUTINE | 1 | | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | | | |
| AYR | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| B | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.2 | 1-3.5.1.6 | 1-3.5.2.3 | 1-3.5.2.5 | |
| | | | 2 | | 1-3.5.2.X | 3-10. | 3-19. | | | |
| BIASCR | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 1-3.5.2.9 | 2-18. | | L2-18. |
| BITSET | SHARE | SUBROUTINE | 1 | | 3-1. | | | | | L3-1 |
| BLKTHM | SHARE | SUBROUTINE | 1 | | 1-3.5.2.5 | 1-3.5.2.7 | 1-3.5.2.8 | 3-2. | | L3-2 |
| BLOCK | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| BLOWUP | PRVT | SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | L2-13.6 |
| BRFCLJ | PRVT | SUBROUTINE | 1 | | 1-3.5.2.9 | | | | | L2-14.6 |
| BSTAT | OFFICE | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.3.1 | 1-3.3.5 | 1-3.5.1.4 | 2-4. | L2-4. |
| BUFCLM | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| BUFDOT | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| C | | VARIABLE | 1 | | 1-3.5.1.2 | 1-3.5.2.3 | | | | |
| C1 | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| C2 | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| C3 | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| C4 | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| CALP | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| CAMSAVE.INC | | FILE | 1 | | 1-3.4.9 | | | | | |
| CAMSCOMUN.INC | | FILE | 1 | | 1-3.4.9 | | | | | |
| CAMSEX | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 2-7. | | | L2-7. |
| CAMSPARAM.INC | | FILE | 1 | | 1-3.4.6 | 1-3.4.9 | 1-3.5.1.3 | 1-3.5.1.5 | | |
| CARD | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |

~~I-1~~

63

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| CARD.DAT | | FILE | 1 | | 1-3.5.1.2 | | | | | |
| CARDIN | PRIVT | FUNCTION | 1 | | 1-3.5.1.3 | | | | | L2-3.10 |
| CAT | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| CATKNT | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.9 | | | |
| CATLOG | PRIVT | FUNCTION | 1 | | 1-3.5.1.3 | | | | | L2-3.2 |
| CATNAM | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.3 | 1-3.5.1.4 | 1-3.5.2.5 | 1-3.5.2.6 | |
| | | | 2 | | 1-3.5.2.7 | 1-3.5.2.9 | | | | |
| CATTH | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.5 | | | |
| CATTHM | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.4 | | | | L2-15.4 |
| CBRR | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| CDRED | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.4 | | | | | L2-4.1 |
| CHAN | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| CHANJC | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| CHL | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| CHNVBC | | VARIABLE | 1 | | 1-3.5.2.5 | | | | | |
| CHNVEC | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.6 | 1-3.5.2.9 | | |
| CHP | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| CLABEL | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.6 | 1-3.5.2.7 | 2-15.7 | | | L2-14.3 |
| | | | 2 | | | | | | | L2-15.7 |
| CLACAT | | VARIABLE | 1 | | 3-4.4 | | | | | |
| CLADIS | | PROGRAM | 1 | | | | | | | L2-15. |
| CLADIS | PRIVT | SUBROUTINE | 1 | | | | | | | L2-16.3 |
| CLASAV | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-21.4 |
| CLASS | | SUBROUTINE | 1 | TASK | 1-CONTNTS | 1-3.5.2.8 | | | | L2-5.3 |
| CLASSMAP.TMP | | FILE | 1 | | 1-3.4.3 | 1-3.5.2.2 | 1-3.5.2.8 | 1-3.5.2.X | | |
| CLATHM | | SUBROUTINE | 1 | | 1-3.5.2.8 | | | | | L2-16.5 |
| CLAWND | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.5 | | | | |
| CLLAB | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| CLOSE | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | | |
| CLREF | F4PLIB | SUBROUTINE | 1 | | 1-3.5.2.1 | | | | | |
| CLSREP | PRIVT | SUBROUTINE | 1 | | 2-19. | | | | | L2-19. |
| CLUARY | | VARIABLE | 1 | | 3-4.2 | | | | | |
| CLUDIS | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 1-3.5.2.7 | 2-15. | | L2-15. |
| CLURPT | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 1-3.5.2.9 | 2-19. | | L2-19. |
| CLUSEL | | SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | L3-4.2 |
| CLUSNN | PRVT | SUBROUTINE | 1 | | 1-3.5.2.9 | | | | | L2-14.8 |
| CLUSTATS.TMP | | FILE | 1 | | 1-3.4.5 | 1-3.5.2.2 | 1-3.5.2.6 | 1-3.5.2.9 | 1-3.5.2.X | |
| CLUSTATP.TMP | | FILE | 1 | | 1-3.5.2.5 | | | | | |
| CLUSTERMP.TMP | | FILE | [illegible] | | 1-3.4.3 | 1-3.5.2.2 | 1-3.5.2.7 | 1-3.5.2.8 | 1-3.5.2.X | |
| CLUSTR | PRVT | SUBROU | | | | | | | | L2-5.4 |
| CLUTHM | PRIVT | SUBROU. | | | 1-3.5.2.7 | 2-15.5 | | | | L2-15.5 |
| CLUWND | COM4 | VARIABL | [illegible] | | 1-3.4.9 | 1-3.5.2.5 | 1-3.5.2.7 | | | |
| CMASK | | VARIABLE | 1 | | 1-3.5.2.7 | | | | | |
| CNTRL | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.2 | | | | L2-2.2 |
| COM1 | GLOBAL | COMMON | 1 | | 1-CONTNTS | 1-3.4.5 | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.2 | 1-F3-3B |
| | | | 2 | | 1-3.5.2.7 | 1-3.5.2.9 | 1-3.5.2.X | | | |
| COM2 | GLOBAL | COMMON | 1 | | 1-CONTNTS | 1-3.3.1 | 1-3.4.4 | 1-3.4.9 | 1-3.5.2.2 | 1-F3-3D |
| COM3 | GLOBAL | COMMON | 1 | | 1-CONTNTS | 1-3.4.9 | 1-3.5.2.2 | 1-3.5.2.X | | 1-F3-3E |
| COM4 | GLOBAL | COMMON | 1 | | 1-CONTNTS | 1-3.4.9 | 1-3.5.2.2 | 1-3.5.2.9 | | 1-F3-3F |
| | | | 2 | | 1-3.5.2.7 | 1-3.5.2.9 | 1-3.5.2.X | | | |
| COM5 | GLOBAL | COMMON | 1 | | 1-CONTNTS | 1-3.4.9 | 1-3.5.2.2 | 1-3.5.2.9 | 1-3.5.2.X | 1-F3-3G |
| COMLUT | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.3 | | | | | L2-9.2 |
| COMPAR | PRIVT | FUNCTIONNE | 1 | | 1-3.5.1.3 | | | | | L2-3.11 |
| CONDIS | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.12 | | | | L215.12 |

184

I-2

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| CONDIT | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.13 | | | | L215.13 |
| CRUNCH | PRVT | SUBROUTINE | 1 | | | | | | | L2-9 |
| CSGDPH | SHARE | SUBROUTINE | 1 | | 1-3.5.2.2 | 1-3.5.2.3 | 1-3.5.2.4 | 1-3.5.2.5 | 1-3.5.2.9 | L3-3. |
| | | | 2 | | 3-3. | | | | | |
| CURDEF | PRVT | SUBROUTINE | 1 | | | | | | | L2-10.4 |
| CURDEF | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| D | | VARIABLE | 1 | | 1-3.5.2.5 | | | | | |
| DATARD | PRVT | SUBROUTINE | 1 | | | | | | | L2-19.3 |
| DATE | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | 1-3.5.2.5 | | | | |
| DAY | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| DCOORD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.6 | | | | L2-2.6 |
| DDDOT | PRIVT | SUBROUTINE | 1 | | | | | | | L2-1.12 |
| DDIESF | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| DEFALT | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.11 | | | | L215.11 |
| DEFLT2 | PRVT | SUBROUTINE | 1 | | | | | | | L2-16.6 |
| DELEAT | PRIVT | SUBROUTINE | 1 | | 1-3.5.1 | | | | | L2-1. |
| DELFL1 | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.4 | | | | | L2-10.1 |
| DELFLG | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| DETACH | !MALIB | SUBROUTINE | 1 | | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | | | |
| DFLAG | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| DGSCHL | | SUBROUTINE | 1 | | 1-CONTNTS | 1-3.5.2.5 | 2-12.1 | | | L2-12.1 |
| DIRCAT | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| DIRCRE | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.1 | | | | | L2-1. |
| DIRCRE | | VARIABLE | 1 | | 1-3.5.1.1 | | | | | |
| DIRFIL | | VARIABLE | 1 | | 1-3.5.1.1 | | | | | |
| DIRFILE.DAT | | FILE | 1 | | 1-3.3 | 1-3.3.1 | 1-3.5.1.2 | 1-3.5.1.4 | 1-3.5.1.5 | |
| | | | 2 | | 1-3.5.1.6 | 1-3.5.2.2 | 1-3.5.2.X | | | |
| DIRLOD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.3 | | | | | L2-3.3 |
| DIRUPD | IMAUPD | PROGRAM | 1 | | 1-3.5.1.1 | | | | | L2-1. |
| DIRUPD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | | | | L2-5.1 |
| DIRUPD | | VARIABLE | 1 | | 1-3.5.1.1 | | | | | |
| DISKID | COM5 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | | | | |
| DISKNM | | VARIABLE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | | | | |
| DIST | | VARIABLE | 1 | | 1-3.4.6 | | | | | |
| DK | | VARIABLE | 1 | | 3-4.2 | | | | | |
| DLAB | | VARIABLE | 1 | | 1-3.4.6 | | | | | |
| DLABEL | COM5 | VARIABLE | 1 | | 1-3.4.4 | 1-3.4.9 | 1-3.5.2.4 | 1-3.5.2.5 | 1-3.5.2.6 | |
| | | | 2 | | 1-3.5.2.9 | 1-3.5.2.X | 3-11. | | | |
| DLABEL | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| DLABEL | COM5 | VARIABLE | 1 | | 1-3.5.2.6 | 1-3.5.2.X | | | | |
| DLSKIP | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| DO1 | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| DO2 | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| DO3 | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| DODU | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| DOI | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| DOLABEL | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| DOPCT | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| DOTARY | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.5 | | | | |
| DOTCAT | COM1 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| DOTCLU | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.9 | | | | |
| DOTDAT | | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| DOTDAY | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |

```
                                                                                          LOCATION OF
NAME        DESCRIPTION  LINE COMMENT                     OCCURRENCES                         LISTING

DOTFLG             VARIABLE   1           1-3.5._.3
DOTGXY.TMP         FILE       1           1-3.4.7    1-3.5.2.2  1-3.5.2.5
DOTIN              SUBROUTINE 1           1-3.5.2.5  1-3.5.2.9                                 L3-4.1
DOTLAB             SUBROUTINE 1           1-3.5.2.5                                            L2-13.4
DOTN               VARIABLE   1           1-3.4.6

DOTOFF   PRIVT SUBROUTINE 1   *UNLDOT
DOTOUR         PROGRAM    1   TASK        1-CONTNTS  1-3.5.2.1  1-3.5.2.5  2-11.               L2-11.
DOTPRO         PROGRAM    1   TASK        1-CONTNTS  1-3.5.2.1  1-3.5.2.5                      L2-13.
DOTRPT         PROGRAM    1   TASK        1-CONTNTS  1-3.5.2.1  2-17.                          L2-17.
DOTSAU   PRIVT SUBROUTINE 1               1-3.5.2.X                                            L2-21.2

DOTSEL         SUBROUTINE 1               1-3.5.2.9  3-4.4                                     L3-4.4
DOTS.TMP       FILE       1               1-3.3      1-3.4.4    1-3.5.2.2  1-3.5.2.5  1-3.5.2.6
                          2               1-3.5.2.X
DOTUPD  OFFICE PROGRAM    1   TASK        1-CONTNTS  1-3.3.1    1-3.3.3    1-3.5.1.3  2-3.     L2-3.
DOTYPE         VARIABLE   1               3-4.4
DSET     SHARE SUBROUTINE 1               1-3.5.1.3  1-3.5.1.5  1-3.5.2.3  3-5.                L3-5.

DSKCHK   SHARE SUBROUTINE 1               1-3.3.1    1-3.6.1.2  1-3.5.1.3  1-3.5.1.4  1-3.5.1.5  L3-6.
                          2               1-3.5.1.6  1-3.5.2.2  3-6.
DSKID          VARIABLE   1               3-6.
DSKMNT   COM3  VARIABLE   1               1-3.4.9
DSKTBL         VARIABLE   1               1-3.5.1.1
DSKTBL.DAT     FILE       1               1-3.3      1-3.3.1    1-3.5.1.6  1-3.5.1.2  1-3.5.2.X

DSSKIP         VARIABLE   1               1-3.5.1.3
DTCL10   SHARE SUBROUTINE 1               3-4.                                                 L3-4.0
DTERM   OFFICE PROGRAM    1   TASK        1-CONTNTS  1-3.3.1    1-3.3.5                        L2-5.
DTRM1          PROGRAM    1   *DTERM      1-3.5.1.5  2-5
DTWIND   COM4  VARIABLE   1               1-3.4.9    1-3.5.2.4

DUPCT          VARIABLE   1               1-3.5.2.9
DUSET          SUBROUTINE 1   *DSET       1-3.5.2.3
DY             VARIABLE   1               1-3.5.1.2  1-3.5.2.X
E              VARIABLE   1               3-4.3
EFLAG1   COM3  VARIABLE   1               1-3.4.3    1-3.4.9    1-3.5.2.7  1-3.5.2.X

EFLAG2   COM3  VARIABLE   1               1-3.4.9    1-3.5.2.X
EFLAG3   COM3  VARIABLE   1               1-3.4.9    1-3.5.2.4  1-3.5.2.X
EFLAG4         VARIABLE   1               1-3.4.6    1-3.4.9    1-3.5.2.6  1-3.5.2.7  1-3.5.2.X
EFLAG5   COM3  VARIABLE   1               1-3.4.5    1-3.4.9    1-3.5.2.X
EFWARN   PRIVT SUBROUTINE 1               1-3.5.2.7  2-15.1                                    L2-15.1

ELAPSE   SHARE SUBROUTINE 1               1-3.5.1.2  1-3.5.1.3  1-3.1.5.5  1-3.5.1.6  1-3.5.2.1  L3-6.1+
                          2               1-3.5.2.2  1-3.5.2.5  1-3.5.2.6  1-3.5.2.8  1-3.5.2.9  L3-7.
                          3               1-3.5.2.X  3-7.
EOF            VARIABLE   1               1-3.5.1.5
ERRMES   SHARE SUBROUTINE 1               1-3.5.1.3  1-3.5.1.5  3-9.                           L3-8
ERROR          VARIABLE   1               1-3.5.1.5
EXFIL          VARIABLE   1               1-3.5.2.5

EXIT    F4PLIBSUBROUTINE 1                1-3.5.2.5
EXPTD    PRIVT SUBROUTINE 1               1-3.5.1.2  2-2.3                                     L2-2.3
F              VARIABLE   1               3-4.3
F11            VARIABLE   1               1-3.5.2.3  3-5.
FADE     PRIVT SUBROUTINE 1   *FMAINT     1-3.5.1.5

FAKCU3   PRIVT SUBROUTINE 1                                                                    L2-10
FDLINT   PRIVT SUBROUTINE 1                                                                    L2-16.1
FFFPI    SHARE SUBROUTINE 1               1-3.5.2.3  3-9.                                      L3-9
FFUNC    SHARE SUBROUTINE 1               1-3.5.2.5  3-10.                                     L3-10.
FIELD    PRIVT SUBROUTINE 1               1-3.5.1.2  2-2.5                                     L2-2.5
```

ORIGINAL PAGE IS
OF POOR QUALITY

186

187

| NAME | DESCRIPTION | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|
| FIELD | VARIABLE | 1 | | 1-3.5.2.4 | 1-3.5.2.8 | | | | |
| FIELDS.TMP | FILE | 1 | | 1-3.3.4 | 1-3.5.2.2 | 1-3.5.2.4 | 1-3.5.2.8 | 1-3.5.2.9 | |
| | | 2 | | 1-3.5.2.X | | | | | |
| FILE | VARIABLE | 1 | | 1-3.5.1.4 | 3-5. | 1-3.5.2.3 | | | |
| FILE | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| FILEIN | VARIABLE | 1 | | 1-3.5.2.2 | | | | | |
| FILEOUT | VARIABLE | 1 | | 1-3.5.2.2 | | | | | |
| FILEST | PRIVT SUBROUTINE | 1 | | 1-3.5.1.1 | | | | | L2-1. |
| FILNUM | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| FILTYP | VARIABLE | 1 | | 1-3.5.2.X | | | | | |
| FINDOT | PRVT SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | L2-13.2 |
| FL | VARIABLE | 1 | | 1-3.5.2.8 | 3-11.2 | | | | |
| FLABEL | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| FLAG | VARIABLE | 1 | | 1-3.5.2.2 | 3-4.4 | 3-6. | | | |
| FLDDAY | COM2 VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| FLDDEF | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 2-10. | | | L2-10. |
| FLDEND | PRIVT SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.7 | | | | L2-2.7 |
| FLDINT | SUBROUTINE | 1 | | 1-3.5.1.2 | 1-3.5.2.8 | 2-2.17 | 3-11. | | L3-11.+ |
| | | 2 | | | | | | | L2-2.17 |
| FLDLAB | VARIABLE | 1 | | 3-11.1 | | | | | |
| FLDNAM | PRIVT SUBROUTINE | 1 | | | | | | | L2-10. |
| FLDNAM | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| FLDOFF | PRIVT SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-21.8 |
| FLDPPT | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 2-20. | | | L2-20. |
| FLDRPT | PRIVT SUBROUTINE | 1 | | 1-3.5.2.4 | 1-3.5.2.9 | | | | L2-10.2 |
| FLDSAV | PRIVT SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-21.3 |
| FLDST | PRIVT SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.4 | | | | L2-2.4 |
| FLDUPD | OFFICE PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.3.1 | 1-3.3.3 | 1-3.3.4 | 1-3.5.1.2 | L2-2.1 |
| | | 2 | | 2-2.1 | | | | | |
| FLGDOT | SHARE SUBROUTINE | 1 | | 1-3.5.1.2 | 1-3.5.2.4 | 2-2.18 | 3-11.1 | | L3-11.1 |
| | | 2 | | | | | | | +L22.18 |
| FLL | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| FLN | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| FMAINT | PRIVT SUBROUTINE | 1 | | 1-3.5.1.5 | | | | | L2-5.5 |
| FN | VARIABLE | 1 | | 3-5. | | | | | |
| FORM | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| FOR001.DAT | FILE | 1 | SYSTEM | 1-3.5.1.3 | | | | | |
| FPTR | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| FRONT | I-100 SUBROUTINE | 1 | | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | 1-3.5.2.9 | | |
| FSTVID | SHARE SUBROUTINE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | 3-12. | | | L3-12. |
| FTRNFR | PRIVT SUBROUTINE | 1 | | 1-3.5.2.2 | | | | | L2-8.1 |
| FUL | COM4 VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.5 | | | | |
| FULL | VARIABLE | 1 | | 1-3.5.2.5 | | | | | |
| FUL012 | PRIVT SUBROUTINE | 1 | =FUL014 | | | | | | L2-9.5 |
| FUL013 | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | 1-3.5.2.3 | 2-9. | | L2-9. |
| FUL014 | PRIVT SUBROUTINE | 1 | TASK | 1-3.5.2.3 | | | | | L2-9.4 |
| FUCLOS | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUDLTE | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUDSET | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUOPEN | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUREAD | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FURWND | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUWAIT | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |
| FUWRIT | I-100 SUBROUTINE | 1 | | 3-12. | | | | | |

```
                                                                                         LOCATION OF
NAME      DESCRIPTION  LINE COMMENT                OCCURRENCES                             LISTING

G         COM4  VARIABLE   1             1-3.4.9    1-3.5.2.3  1-3.5.2.5
GABI      PRIVT SUBROUTINE 1             1-3.5.2.3                                         L2-9.1
GB              VARIABLE   1             1-3.5.2.3
GBCALC    PRIVT SUBROUTINE 1                                                               L2-9.6
GETADR    F4PLIBSUBROUTINE 1             1-3.5.1.4

GETC00    SHARE SUBROUTINE 1             1-3.5.2.7  2-15.8     3-13.                       L3-11.+
                           2                                                               L2-15.8
GLOBAL.TMP      FILE       1             1-3.4.9    1-3.5.2.2  1-3.5.2.5  1-3.5.2.9
GMAX      COM4  VARIABLE   1             1-3.4.9    1-3.5.2.5
GMIN      COM4  VARIABLE   1             1-3.4.9    1-3.5.2.5
GRENO           VARIABLE   1             1-3.5.2.9

GRENOB          VARIABLE   1             1-3.5.2.5
GRENSS          VARIABLE   1             1-3.5.2.9
GRID      COM5  VARIABLE   1             1-3.4.9
GTYPE     PRVT  SUBROUTINE 1             1-3.5.2.5                                         L2-13.7
HORREL.DAT      FILE       1             1-3.2.2

HEXD      PRIVT SUBROUTINE 1                                                               L2-4.3
HEADIN    PRIVT SUBROUTINE 1             1-3.5.1.5                                         L2-5.9
HFG             VARIABLE   1             1-3.5.2.3
HFL             VARIABLE   1             1-3.5.2.3
HOCUTT    PRIVT SUBROUTINE 1                                                               L2-9.4

HPROS     SHARE SUBROUTINE 1             1-3.5.1.5  3-14.                                  L3-14.
HREAD           SUBROUTINE 1   *DSET     1-3.5.2.3
HSEKPG    PRIVT SUBROUTINE 1             1-3.5.2.5                                         L2-13.5
                           2                                                               L2-19.4
HSIZ            VARIABLE   1             1-3.5.2.3
HVFY      SHARE SUBROUTINE 1             1-3.5.1.5  3-15.                                  L3-15.

I               VARIABLE   1             1-3.5.1.2  1-3.5.1.6  1-3.5.2.5  1-3.5.2.X  3-1.1
                           2             3-9.1      3-17.
IBUF            VARIABLE   1             1-3.5.1.2  1-3.5.1.4  1-3.5.2.5  3-2.
IBYTE     IMALIBSUBROUTINE 1             1-3.5.2.5
IC              VARIABLE   1             1-3.5.2.7  3-13.
ICAKNT          VARIABLE   1             1-3.5.2.8

ID              VARIABLE   1             1-3.5.1.2  1-3.5.1.5
IDATE     F4PLIBSUBROUTINE 1             1-3.5.2.5  1-3.5.2.9
IDENT1    PRIVT SUBROUTINE 1             1-3.5.2.8
IER             VARIABLE   1             1-3.4.6
IERR            VARIABLE   1             1-3.5.1.2

IFST            VARIABLE   1             1-3.5.2.4
II              VARIABLE   1             1-3.5.2.5  1-3.5.2.9  3-7.
II1       COM4  VARIABLE   1             1-3.4.9    1-3.5.2.3  1-3.5.2.5
IIX1            VARIABLE   1             1-3.5.2.3
IIX2            VARIABLE   1             1-3.5.2.3

IIY1            VARIABLE   1             1-3.5.2.3
IIY2            VARIABLE   1             1-3.5.2.3
ILABEL          VARIABLE   1             1-3.5.2.6
IMAUPD  OFFICE  PROGRAM    1   TASK      1-3.3      1-3.3.1    1-3.3.2    1-3.3.3    1-3.5.1      L2-1.
IMDATE    COM2  VARIABLE   1             1-3.4.9    1-3.5.2.X

IMWIND    COM4  VARIABLE   1             1-3.4.9    1-3.5.2.5
INCLDS    PRIVT SUBROUTINE 1             1-3.5.2.9                                         L2-19.2
INDEPT    PRIVT SUBROUTINE 1             1-3.5.1.2                                         L2-2.2
INIT     INTRAC PROGRAM    1   TASK      1-CONTNS   1-3.3.1    1-3.3.3    1-3.3.4    1-3.3.5      L2-8.
                           2             1-3.5.2.1  1-3.5.2.2  2-8
INIT            VARIABLE   1             1-3.5.1.3
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
                                                                                                     LOCATION OF
NAME      DESCRIPTION  LINE COMMENT                          OCCURRENCES                               LISTING

INTFF     F4PLIBSUBROUTINE 1            1-3.5.2.5  1-3.5.2.6  1-3.5.2.8  1-3.5.2.9
INTLZE    PRIVT SUBROUTINE 1            1-3.5.2.2                                                       L2-8.2
INX             VARIABLE   1            1-3.5.1.2
IO              VARIABLE   1            1-3.5.2.4  1-3.5.2.5  1-3.5.2.9  3-3.       3-4.1
IOP             VARIABLE   1            1-3.5.2.5  3-2.

IOPT            VARIABLE   1            1-3.5.1.2
IP              VARIABLE   1            1-3.5.2.3
IPUDEF.TMP      FILE       1            1-3.5.2.4
IRESUB          VARIABLE   1            1-3.5.2.8
IPREG3          SUBROUTINE 1   TASK     1-3.5.2.4

IRT       IMALIBSUBROUTINE 1            1-3.5.2.5
IRU      IMALIB SUBROUTINE 1            1-3.5.2.5
IOPRNT    PRIVT SUBROUTINE 1                                                                            L2-4.2
                           2            3-4.2
ISEG      COM2  VARIABLE   1            1-3.4.9    1-3.5.2.3  1-3.5.2.6  1-3.5.2.9
ISET            VARIABLE   1            3-13.

ITJCHN          VARIABLE   1            1-3.5.2.6
IU              VARIABLE   1            3-17.
IWL      IMALIB SUBROUTINE 1            1-3.5.2.5
IWT      IMALIB SUBROUTINE 1            1-3.5.2.5
IX              VARIABLE   1            1-3.5.2.3  1-3.5.2.7  3-4.2

IX1       COM4  VARIABLE   1            1-3.4.9    1-3.5.2.3  1-3.5.2.7
IX2       COM4  VARIABLE   1            1-3.4.9    1-3.5.2.3  1-3.5.2.7
IXY             VARIABLF   1            1-3.5.1.2
IY              VARIABLE   1            1-3.5.2.3  1-3.5.2.7
IY1       COM4  VARIABLE   1            1-3.4.9    1-3.5.2.3  1-3.5.2.7

IY2       COM4  VARIABLE   1            1-3.4.9    1-3.5.2.3  1-3.5.2.7
J               VARIABLE   1            1-3.5.1.2  1-3.5.1.6  1-3.5.2.X
JDIR            PROGRAM    1   =IMAUPD                                                                  L2-1.
JJ              VARIABLE   1            3-11.2
JJ              VARIABLE   1            1-3.5.2.8

JJL             VARIABLE   1            1-3.5.2.8
JUL10           VARIABLE   1            1-3.5.1.2  1-3.5.2.X
JULDAT          VARIABLE   1            1-3.5.1.5
JULIAN    PRIVT FUNCTION   1                                                                            L2-3. +
                           2                                                                            L2-5.8
JULIAN    PRIVT SUBROUTINE 1            1-3.5.1.2  1-3.5.1.3  1-3.5.2.X  2-2.16                         L2-1 +
                           2                                                                            L2-2.16

K               VARIABLE   1            1-3.5.2.5  1-3.5.2.X
KAUTH           SUBROUTINE 1            1-3.5.1.4                                                       L2-1.
KK              VARIABLE   1            3-4.1
KNN             VARIABLE   1            1-3.4.6    1-3.5.2.6
KNNPRN    PRIVT SUBROUTINE 1            1-3.5.2.6                                                       L2-14.5

KOMBRT    PRIVT SUBROUTINE 1            1-3.5.1.2                                                       L2-2.2
KRW             VARIABLE   1            1-3.5.2.2
L               VARIABLE   1            3-17.
LABEL           VARIABLE   1            1-3.5.1.3  1-3.5.2.8
LABNUM          VARIABLE   1            1-3.5.1.3

LE              VARIABLE   1            1-3.5.2.3
LECTAP    SHARE SUBROUTINE 1            1-3.5.1.3  1-3.5.1.5  3-16.                                     L3-16.
LGBC            VARIABLE   1            1-3.5.2.3
LGF             VARIABLE   1            1-3.5.2.3
LGL             VARIABLE   1            1-3.5.2.3
```

ORIGINAL PAGE IS
OF POOR QUALITY

| NAME | DESCRIPTION | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|
| LIN | SHARE SUBROUTINE | 1 | | 1-3.5.2.3 | 1-3.5.2.4 | 1-3.5.2.5 | 3-17. | | L3-17. |
| LIST1 | PRIVT SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.3 | | | | L2-15.3 |
| LIST2 | PRIVT SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.15 | | | | L2-15.5 |
| LIST3 | PRIVT SUBROUTINE | 1 | | | | | | | L3-16.4 |
| LOOKUP | VARIABLE | 1 | | | | | | | L2-5.11 |
| LOWHL | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| LREED | SUBROUTINE | 1 | =DSET | 1-3.5.1.5 | | | | | |
| LRJUNK | PRIVT SUBROUTINE | 1 | | | | | | | L2-12.2 |
| LS | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| LUN | VARIABLE | 1 | | 1-3.4.6 | 1-3.5.2.3 | 3-5. | | | |
| LUN | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| LUT | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| M | VARIABLE | 1 | | 1-3.5.1.2 | 1-3.5.1.6 | 1-3.5.2.X | | | |
| MAPUPD | PRIVT SUBROUTINE | 1 | | 1-3.5.1.5 | | | | | L2-5.2 |
| MAXACC | VARIABLE | 1 | | 1-3.4.5 | 1-3.4.9 | 1-3.5.2.9 | | | |
| MAXACD | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| MAXCAT | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| MAXCHN | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.3 | 1-3.5.1.5 | | | |
| MAXSUB | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| MB | VARIABLE | 1 | | 1-3.5.2.5 | 3-2. | | | | |
| MDODU | PROGRAM | 1 | =FLDUPD | | | | | | L2-2.1 |
| MENSTD | PRIVT SUBROUTINE | 1 | | 1-3.5.2.9 | | | | | L2-19.1 |
| MFLDS | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| MGL | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| MIXDIS | PRIVT SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.14 | | | | L215.14 |
| MIXED | PRIVT SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.16 | | | | L215.16 |
| ML | VARIABLE | 1 | | 1-3.5.2.5 | 3-2. | | | | |
| MO | VARIABLE | 1 | | 1-3.5.1.2 | 1-3.5.2.X | | | | |
| MR | VARIABLE | 1 | | 1-3.5.2.2 | 1-3.5.2.5 | 3-2. | | | |
| MTXTFG | VARIABLE | 1 | | 1-3.5.2.3 | 3-5. | | | | |
| MU | VARIABLE | 1 | | 1-3.5.2.5 | 3-2. | | | | |
| MX | VARIABLE | 1 | | 1-3.5.2.3 | 1-3.5.2.7 | | | | |
| MXC | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| MY | VARIABLE | 1 | | 1-3.5.2.3 | 1-3.5.2.7 | | | | |
| N | VARIABLE | 1 | | 1-3.5.1.2 | 1-3.5.1.6 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.X | |
| | | 2 | | 3-3. | 3-9. | | | | |
| N | VARIABLE | 1 | | 1-3.5.2.6 | 3-3. | | | | |
| NACQ | VARIABLE | 1 | | 1-3.5.2.6 | | | | | |
| NBIT | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NC | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NCAR | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NCMAX | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NCNTRL | PRIVT SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.20 | | | | L2-2.20 |
| NCPR | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NCR | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| ND | VARIABLE | 1 | | 1-3.4.6 | | | | | |
| NDOPIX | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| NDOT | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| NDOTS | VARIABLE | 1 | | 1-3.4.4 | 1-3.4.9 | 1-3.5.1.3 | | | |
| NDSPR | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NDUPIX | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |

191

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| NEWLAB | COM3 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.6 | 1-3.5.2.7 | | | |
| NF | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NFIELD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.21 | | | | L2-2.21 |
| NFL | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| NFLD | | VARIABLE | 1 | | 1-3.5.2.4 | | | | | |
| NFLDST | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.19 | | | | L2-2.19 |
| NLIN | | VARIABLE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | 1-3.5.2.8 | | | |
| NLP | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| NN.TMP | | FILE | 1 | | 1-CONTNTS | 1-3.4.6 | 1-3.5.2.6 | 1-3.5.2.7 | | |
| NNNNYYDDD.DAT | | FILE | 1 | | 1-3.5.1.1 | | | | | |
| NOACQ | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.9 | 1-3.5.2.X | | | |
| NOCAT | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.5 | 1-3.5.2.9 | 1-3.5.2.X | |
| NOCHAN | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | | | | |
| NODO | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.9 | | | |
| NODOT | | VARIABLE | 1 | | 3-4.4 | | | | | |
| NODTND | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| NODTIE | | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| NODU | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.9 | | | |
| NOFLD | | VARIABLE | 1 | | 3-11.1 | | | | | |
| NOLIN | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| NOSPWD | | VARIABLE | 1 | | 1-3.4.8 | 1-3.4.9 | | | | |
| NOSUB | COM1 | VARIABLE | 1 | | 1-3.4.5 | 1-3.4.6 | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.7 | |
| | | | 2 | | 1-3.5.2.9 | | | | | |
| NOTH | COM1 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.1.4 | 1-3.5.2.9 | | | |
| NPIX | | VARIABLE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | 1-3.5.2.8 | | | |
| NPIX4 | | VARIABLE | 1 | | 1-3.5.1.5 | | | | | |
| NPTS | | VARIABLE | 1 | | 3-11.2 | | | | | |
| NR | | VARIABLE | 1 | | 1-3.5.2.2 | 3-19. | | | | |
| NRPDS | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NS | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| NSAMP | | VARIABLE | 1 | | 1-3.5.2.8 | 3-11.2 | | | | |
| NSEGND | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.22 | | | | L2-2.22 |
| NSTART | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| NT | | VARIABLE | 1 | | 1-3.5.2.5 | 3-2. | | | | |
| NTH | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| NTYPE1 | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.5 | 1-3.5.2.6 | | | |
| NUCAT | | VARIABLE | 1 | | 1-3.5.2.7 | | | | | |
| NUMDOT | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.5 | | | | |
| NU | | VARIABLE | 1 | | 1-3.5.2.4 | 1-3.5.2.8 | 3-11.1 | | | |
| NX | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| OPEN | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | 1-3.5.2.9 | | | | |
| OPMESS | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | | | | L2-3.13 |
| | | | 2 | | | | | | | L2-5.7 |
| OUTFILE.DAT | | FILE | 1 | | 1-3.5.1.2 | | | | | |
| OUTPUT | IMALIB | SUBROUTINE | 1 | | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | 1-3.5.2.9 | | |
| P | | VARIABLE | 1 | | 3-3. | 3-19. | | | | |
| PAINT | PRIVT | SUBROUTINE | 1 | | | | | | | L2-10. |
| PCTCT | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| PCTCT0 | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| PDATE1 | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| PDATE2 | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| PDATE3 | COM2 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| PERDO | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| PERDU | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| PERTH | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| PERUND | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| PFLAG | COM3 | VARIABLE | 1 | | 1-3.4.3 | 1-3.4.5 | 1-3.4.9 | 1-3.5.2.X | | |
| PLOT | | SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | L2-12.3 |
| POLYCR | | PROGRAM | 1 | | | | | | | L2-10. |
| POLYCR | | SUBROUTINE | 1 | | | | | | | L2-10.3 |
| PRESET | PRIVT | SUBROUTINE | 1 | | | | | | | L2-1. |
| PRMUPD | INTRAC | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.3.1 | 1-3.3.3 | 1-.3.34 | 1-3.3.5 | L2-21. |
| | | | 2 | | 1-3.3.2.1 | 1-3.5.2.X | | | | |
| PROCED | PRIVT | SUBROUTINE | 1 | | | | | | | L2-13.3 |
| PTR | | VARIABLE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | 3-6. | | | |
| Q10 | | SUBROUTINE | 1 | | 1-3.5.1.4 | | | | | L2-1. |
| R | | VARIABLE | 1 | | 1-3.5.1.4 | | | | | |
| RANDOM | COM5 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.6 | | | | |
| RANDOT | | VARIABLE | 1 | | | | | | | L2-3.5 |
| RDCARD | PRIVT | FUNCTION | 1 | | 1-3.5.1.3 | | | | | L2-3.8 |
| RDCLMN | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.6 | | | | | L2-14.2 |
| RDDIR | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.9 | | | | L2-2.9. |
| RDDISK | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.1 | | | | | L2-1. |
| RDDGDU | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.11 | | | | L2-2.10 |
| RDDOT | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.10 | | | | L2-2.11 |
| RDFLD | PRIVT | SUBROUTINE | 1 | | | | | | | L2-21.9 |
| RDHEAD | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.1 | | | | | L2-1. |
| RDDDAT | | SUBROUTINE | 1 | | 1-3.5.2.6 | | | | | L2-14.1 |
| RDXYD3 | PRIVT | SUBROUTINE | 1 | | | | | | | L2-10.6 |
| RECPCT | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| RECPRN | | SUBROUTINE | 1 | | 1-3.5.2.8 | | | | | L2-16.2 |
| RECPRO | | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.8 | 2-16. | | | L2-16. |
| REPORT | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.1 | 1-3.5.1.5 | 2-15.10 | | | L2-1 + |
| | | | 2 | | | | | | | L2-5.10 |
| | | | 3 | | | | | | | L2-14.7 |
| | | | 4 | | | | | | | L215.10 |
| REPROP | PRIVT | SUBROUTINE | 1 | TASK | 1-CONTNTS | 1-3.5.2.8 | | | | L2-16. |
| REPRTN | PRIVT | SUBROUTINE | 1 | | | | | | | L2-14.9 |
| REQUES | F4PLIB | SUBROUTINE | 1 | | 1-3.5.2.1 | | | | | |
| RFAC | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| RIELD | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| RNUM | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| ROFF | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.9 | | | | | L2-18.1 |
| ROFNO | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| RPTGEN | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-21.7 |
| RREAD | SHARE | SUBROUTINE | 1 | | 1-3.5.1.3 | 1-3.5.1.5 | 3-18. | | | L3-18. |
| RS | | VARIABLE | 1 | | 3-19. | | | | | |
| RSIZ | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| RSKIP | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| RWD | | VARIABLE | 1 | | 3-5. | | | | | |
| S | | VARIABLE | 1 | | 3-3. | | | | | |
| SCATXY.TMP | | FILE | 1 | | 1-3.4.8 | 1-3.5.2.2 | 1-3.5.2.5 | | | |
| SCPLOT | DRIVER | PROGRAM | 1 | TASK | 1-CONTNTS | 1-3.5.2.1 | | | | L2-12. |
| SDLINE | | | 1 | | | | | | | L2-10.7 |
| SDPNT | PRIVT | SUBROUTINE | 1 | | | | | | | L2-10.8 |
| SE | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |

~~I-10~~

(192)

ORIGINAL PAGE IS
OF POOR QUALITY

```
                                                                                               LOCATION OF
NAME      DESCRIPTION  LINE COMMENT                     OCCURRENCES                              LISTING

SECNDS   F4PLIBSUBROUTINE 1            1-3.5 1.4  .-3.5.2.5
SEGDEL OFFICE PROGRAM     1   TASK     1-CONTNTS  1-3.3.3    1-3.3.4    1-3.3.5    1-3.5.1.6       L2-6.
                          2            2-6.
SEGEND   PRIVT SUBROUTINE 1            1-3.5.1.2  2-2.8                                          L2-2.8
SEGNO          VARIABLE   1            1-3.5.2.X  3-6.
SEGNUM         VARIABLE   1            1-3.5.1.5

SELECT   PRIVT SUBROUTINE 1            1-3.5.2.9                                                 L2-18.2
SETBIT   IMALIBSUBROUTINE 1            1-3.5.2.5
SETEF    F4PLIBSUBROUTINE 1            1-3.5.2.5  1-3.5.2.9
SETVID         SUBROUTINE 1            1-3.5.2.5                                                 L2-12.4
SETWIN         SUBROUTINE 1            1-3.5.2.6                                                 L2-12.5

SHELL    SHARE SUBROUTINE 1            1-3.5.2.5  3-19.                                          L3-19.
SKIP     PRIVT FUNCTION   1            1-3.5.1.3                                                 L2-3.9
SOILGR   COM2  VARIABLE   1            1-3.4.9    1-3.5.2.5
SORT     PRIVT SUBROUTINE 1                                                                      L2-10.X
SORTRC   PRIVT SUBROUTINE 1                                                                      L2-5.6

SPWIND   COM4  VARIABLE   1            1-3.4.9    1-3.5.2.5
SRDISK   PRIVT SUBROUTINE 1            1-3.5.1.1                                                 L2-1.
SRDISK         VARIABLE   1            1-3.5.1.1
SS             VARIABLE   1            1-3.5.2.3
SSSSYYDDD.DAT FILE        1            1-3.5.1.1

START          VARIABLE   1            1-3.5.1.3
STASAV   PRIVT SUBROUTINE 1            1-3.5.2.X                                                 L2-21.5
STATFIL.TMP    FILE       1            1-3.5.2.2
STRAYS   PRIVT SUBROUTINE 1            1-3.5.1.1                                                 L2-1.
STYPE    PRIVT SUBROUTINE 1            1-3.5.2.5                                                 L2-13.1

SUBCAT   COM1  VARIABLE   1            1-3.4.9    1-3.5.1.4   -13.5.2.  9-
SUBPOP   COM1  VARIABLE   1            1-3.4.9    1-3.5.1.4
SUBSTR   SHARE SUBROUTINE 1            1-3.5.1.3  1-3.5.1.5  1-3.5.1.6  1-3.5.2.X  3-20.         L2-3.12
                          2                                                                      L2-6. +
                          3                                                                      L2-13.+
                          4                                                                      L2-21.6
                          5                                                                      L3-20.
SUD           .VARIABLE   1            1-3.5.2.3'
SUNAZ    COM2  VARIABLE   1            1-3.4.9

SUNEL    COM2  VARIABLE   1            1-3.4.9    1-3.5.2.6
SWCLR    PRIVT SUBROUTINE 1                                                                      L2-10.9
T2       PRIVT SUBROUTINE 1   T2DRMOD
T2DR     PRIVT SUBROUTINE 1                                                                      L2-1.
TABLE    SHARE SUBROUTINE 1            3-4.                                                      L3-4.

TAPSCN   PRIVT SUBROUTINE 1            1-3.5.1.1                                                 L2-1.
TC             VARIABLE   1            1-3.5.2.7  3-13.
TCHLST         SUBROUTINE 1            1-3.5.2.3                                                 L2-9.3
TCLANM.MAP     FILE       1            1-3.5.1.5
TCLUNM.MAP     FILE       1            1-3.5.1.5

TDATE1   COM2  VARIABLE   1            1-3.4.9    1-3.5.2.X
TDATE2   COM2  VARIABLE   1            1-3.4.9    1-3.5.2.X
TDATE3   COM2  VARIABLE   1            1-3.4.9    1-3.5.1.3  1-3.5.1.4  1-3.5.2.X
TDIS           VARIABLE   1            1-3.5.2.6
THLOPM         SUBROUTINE 1   TASK     1-CONTNTS  1-3.5.2.5                                      L2-12.8
```

| NAME | DESCRIPTION | | LINE | COMMENT | OCCURRENCES | | | | | LOCATION OF LISTING |
|---|---|---|---|---|---|---|---|---|---|---|
| THMLOP | SHARE | SUBROUTINE | 1 | | 1-CONTNTS | 3-21. | | | | L3-21. |
| TI | | VARIABLE | 1 | | 3-17. | | | | | |
| TIME | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.8 | 1-3.5.2.9 | |
| TOBDON | | VARIABLE | 1 | | 1-3.5.1.1 | | | | | |
| TP | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TP1XL | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| TPCT | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| TRAJPL | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | L2-13.8 |
| TSTAT.DAT | | FILE | 1 | | 1-3.5.1.5 | | | | | |
| TTX2 | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TTXI | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TTY1 | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TTY2 | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TUN | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TUNIT | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| TWRITE | SHARE | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.6 | 3-22. | | | L2-15.6 |
| | | | 2 | | | | | | | L3-22. |
| TX | | VARIABLE | 1 | | 1-3.5.2.3 | 1-3.5.2.7 | | | | |
| TX1 | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | 1-3.5.2.7 | | |
| TX2 | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | 1-3.5.2.7 | | |
| TY | | VARIABLE | 1 | | 1-3.5.2.3 | 1-3.5.2.7 | | | | |
| TY1 | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | 1-3.5.2.7 | | |
| TY2 | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.3 | 1-3.5.2.5 | 1-3.5.2.7 | | |
| TYPE | COM6 | VARIABLE | 1 | | 1-3.4.4 | 1-3.4.9 | 1-3.5.2.5 | 1-3.5.2.6 | 1-3.5.2.9 | |
| | | | 2 | | 1-3.5.2.X | | | | | |
| TYPE | COM5 | VARIABLE | 1 | | 1-3.5.2.6 | 1-3.5.2.9 | 1-3.5.2.X | | | |
| TYPEI | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| TYPI | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| UFLAG1 | COM3 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| UFLAG2 | COM3 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.4 | 1-3.5.2.X | | | |
| UFLAG3 | COM3 | VARIABLE | 1 | | 1-3.4.4 | 1-3.4.9 | 1-3.5.2.X | | | |
| UFLAG4 | COM4 | VARIABLE | 1 | | 1-3.4.9 | 1-3.5.2.X | | | | |
| UNBIAS | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.9 | | | | | L2-18.3 |
| UNCDIS | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.7 | 2-15.2 | | | | L2-15.2 |
| UNID | | VARIABLE | 1 | | 1-3.5.2.8 | | | | | |
| UNIT | | VARIABLE | 1 | | 3-5. | | | | | |
| UNLDOT | PRIVT | SUBROUTINE | 1 | | 1-3.5.2.X | | | | | L2-3 |
| UNPAK | PRIVT | FUNCTION | 1 | | 1-3.5.1.3 | | | | | L2-3.1 |
| UNPAK | | VARIABLE | 1 | | 1-3.5.1.3 | | | | | |
| UPCT | | VARIABLE | 1 | | 1-3.5.2.9 | | | | | |
| UPDATE | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.3 | 1-3.5.2.6 | | | | L2-3.6 |
| UPDOT | PRIVT | SUBROUTINE | 1 | | 1-3.5.1.2 | 2-2.15 | | | | L2-2.15 |
| V | | VARIABLE | 1 | | 1-3.5.2.3 | 3-9. | | | | |
| VALCK | PRIVT | SUBROUTINE | 1 | | | | | | | L2-12.6 |
| VAR | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| VAR0 | COM2 | VARIABLE | 1 | | 1-3.4.9 | | | | | |
| VDALTR | SHARE | SUBROUTINE | 1 | | 1-3.5.2.5 | 3-23. | | | | L3-23. |
| VERTEX | | VARIABLE | 1 | | 1-3.5.2.8 | 3-11.1 | | | | |
| WAIT | IMALIB | SUBROUTINE | 1 | | 1-3.5.2.5 | | | | | |
| WAITFR | F4PLIB | SUBROUTINE | 1 | | 1-3.5.1.4 | 1-3.5.2.1 | | | | |
| WC | | VARIABLE | 1 | | 1-3.5.2.3 | | | | | |
| WINDER | SHARE | SUBROUTINE | 1 | | 1-CONTNTS | 3-24. | | | | L3-24. |

```
                                                                                   LOCATION OF
NAME         DESCRIPTION  LINE COMMENT              OCCURRENCES                      LISTING

WINDRM              PROGRAM     1                                                  L2-12.7
WINDRM              SUBROUTINE  1  TASK   1-CONTNTS 1-3.5.2.5                      L2-12.
WORK                VARIABLE    1         1-3.5.1.3
WRDIR        PRIVT  SUBROUTINE  1         1-3.5.1.2 2-2.14                         L2-2.14
WRDODU       PRIVT. SUBROUTINE  1         1-3.5.1.2 2-2.12                         L2-2.12

WRDOT        PRIVT  SUBROUTINE  1         1-3.5.1.2 2-2.13                         L2-2.13
X                   VARIABLE    1         1-3.5.2.3 1-3.5.2.8
XPCT                VARIABLE    1         1-3.5.2.9
XXXXDOOTS.DAT FILE              1         1-3.5.2.X
XXXXDOTS.DAT  FILE              1         1-3.3     1-3.3.3   1-3.4.4   1-3.5.1.3 1-3.5.2.2

XXXXFIELD.DAT FILE              1         1-3.3     1-3.3.4   1-3.5.1.2 1-3.5.2.2 1-3.5.2.X
XXXXPCLAS.MAP FILE              1         1-3.3     1-3.3.5   1-3.5.2.2 1-3.5.2.X
XXXXPCLUS.MAP FILE              1         1-3.3     1-3.3.5   1-3.4.3   1-3.5.2.2 1-3.5.2.X
XXXXPSTAT.DAT FILE              1         1-3.3     1-3.3.5   1-3.4.5   1-3.5.2.2 1-3.5.2.X
XXXXTCLAS.MAP FILE              1         1-3.3     1-3.3.5   1-3.5.2.2 1-3.5.2.X

XXXXTCLUS.MAP FILE              1         1-3.3     1-3.3.5   1-3.4.3   1-3.5.2.2 1-3.5.2.X
XXXXTSTAT.DAT FILE              1         1-3.3     1-3.3.5   1-3.4.5   1-3.5.1.4 1-3.5.2.2
                                2         1-3.5.2.X
XXXXYYDDD.DAT FILE              1         1-3.3
XZ                  VARIABLE    1         1-3.5.2.3
Y                   VARIABLE    1         1-3.5.2.3 1-3.5.2.5

YLINE               VARIABLE    1         3-11.2
YR                  VARIABLE    1         1-3.5.1.2 1-3.5.1.3 1-3.5.1.5 1-3.5.2.X
YZ                  VARIABLE    1         1-3.5.2.3
ZNAME        PRIVT  SUBROUTINE  1                                                  L2-5.12
ZO                  VARIABLE    1         1-3.5.2.7

ZOOOM        PRIVT  SUBROUTINE  1         1-3.5.2.3 1-3.5.2.7 1-3.5.2.8 2-15.9    3-25.     L2-15.9
                                2                                                  +L3-25.
```

I-12

/95

ORIGINAL
OF POOR